2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC)

A Game Recommendation Method Based on Machine Learning

Qinyuan Li [1, †]
[1]School of Computer Science
Beijing Institute of Technology
Beijing, China
1120180710@bit.edu.cn

Xiuping Liang [2, *, †]
[2]The Holcombe Department of Electrical and Computer
Clemson University
Clemson, South Carolina, United States
[*]Corresponding author's e-mail:
xiupinl@g.clemson.edu

Chenming Su [3, †]
[3]School of Electrical and Computer Engineering
Xiamen University Malaysia
Xiamen, China
CST1709335@xmu.edu.my

Yankai Wang [4, †]
[4]Department of Mechanical Engineering
McMaster University
Hamilton, Canada
wangy276@mcmaster.ca
[†]These authors contributed equally.

***Abstract*—The emergence and popularity of the Internet have brought users a large amount of information to meet their information needs. Still, the rapid development of the Internet has brought about a significant increase in the amount of online information, making it impossible for users to obtain the part of the information that is useful to them. The efficiency of using information has been reduced instead. This is when a recommender system is needed to recommend the content of interest to users based on their information. Any website or platform has its own independent recommender system, and collaborative filtering is one of the traditional recommender algorithms. This paper uses steam's users' data to cluster and label separately to create four different collaborative filtering game-recommender models based on historical purchases, game time, and the relationship between items. They then compare their advantages and disadvantages to determine which model is more suitable for game recommender systems.**

Keywords-Game Recommender; Apriori; Collaborative Filtering; Recommender Algorithm; User Behavior.

I. INTRODUCTION

In the era of the information explosion, the amount of data is growing all over the world. The huge amount of data gives people more information and leads to many problems for users to choose information. E-commerce websites need to improve their ability to extract information and analyze it to meet the needs of different types of users. With the exponential growth in the volume of information, recommender systems have been an effective strategy to overcome such information overload. It helps users discover the information they are interested in based on their historical behavior and provides personalized services.

Traditional recommender systems include three types of algorithms: content-based systems, collaborative filtering-based systems, and hybrid systems, which basically work in one of two ways: suggesting items similar to the ones a person likes or suggesting items liked by similar people to the user [1].

Due to the development of deep learning and neural networks, most of the research on recommendation algorithms has recently focused on this field. Using auxiliary information, Zheng respectively extracted user behavior features, and item attribute features from user comments to construct two convolutional neural networks to predict user ratings of items [2]. Nie et al. proposed to use deep neural networks to learn non-associative functions between users and items to improve the interaction information between items and users [3]. Based on the model, Salakhutdinov et al. used a two-layer RBM model with scores from training data as input to predict scores by a backpropagation algorithm [4]. Tang et al. constructed a neural autoregressive collaborative filtering model using the NADE model to improve the speed of RBM training [5]. To satisfy the dynamic information, Devooght et al. used LSTM to improve the diversity and coverage of recommendations by considering the timing sequence of users' consumption of items [6]. Hidasi et al. proposed a GRU-RNN training model based on recurrent neural networks to enable the dynamically changing user history information to be updated in training [7]. Using tags, Xu et al. proposed a deep semantic similarity model for a tag-based item recommendation, which uses the labeled user information and item information in two deep neural networks for item recommendation [8].

However, because deep learning itself is not interpretable and has limitations such as poor robustness, such algorithms cannot explore the computational method of the model from the principle [9]. Moreover, the recommender models based on deep learning nowadays still use auxiliary information such as users and items as the input features, which extends traditional recommender algorithms.

In this paper, the experiment process contains the following steps. First, the data set cleaning and data normalization. After finishing the data processing, since the data in this data set are all unlabeled, we used the KMeans algorithm to cluster and label different users. After a simple analysis of users with

978-1-6654-0605-5/21/$31.00 ©2021 IEEE

2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC) | 978-1-6654-0605-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICFTIC54370.2021.9647367

different labels, we used the K- Nearest Neighbor (KNN) algorithm to check the clustering accuracy. Although good results have been obtained, two different algorithm experiments are carried out on this basis. The next part of the experiment was an Apriori algorithm experiment. The purpose is to try to recommend games for users through the frequent itemsets of the game. However, the accuracy of the results obtained is not good. Then we try to use the data after clustering for the Apriori algorithm. Because clustering will classify users once based on user behavior, this will improve the predicted results, but the result did not improve much. One of the main reasons is that the Apriori algorithm is item-based. The clustering is based on user behavior, so the data set after clustering does not improve the accuracy of the prior algorithm very well. Then, we lowered the support of the Apriori algorithm and changed it to another algorithm, the item collaborative filtering algorithm. This greatly improves the accuracy of the results. In addition, as a comparison, Singular Value Decomposition (SVD) was used to recommend the game to users from another aspect.

II. METHODS

A. KMeans and KNN

KMeans is an algorithm for finding k clusters in a given data set. The user gives the number of clusters k and can also be determined by the elbow method and other methods. Each cluster is described by its centroid, which is the center of all points in the cluster. The steps of the K-means algorithm are as follows: First, to ensure that the impact caused by the difference in the magnitude of the data is reduced, the Z-SCORE method is used to normalize the data. The specific calculation formula is as follows:

$$\mathrm{x} = \frac{\mathrm{x}-\mu}{\sigma} \tag{1}$$

Where x is the data, μ is the mean of the original data, and σ is the standard deviation of the original data.

Second, determine k initial points as centroids by the elbow method. The calculation formula of the error sum of squares is as shown in the figure below:

$$SSE = \sum_{i=1}^{k} \sum_{p \in c_i} (pm_i)^2 \tag{2}$$

Where k refers to the number of clusters, c_i is the *i-th* cluster, p is the sample point in c_i, and m_i is the mean of the samples in c_i.

The next step is to assign each point in the data set to a cluster. That is, find the centroid closest to it for each point and assign it to the cluster corresponding to the centroid. Finally, the centroid of each cluster is updated to the average value of all points in the cluster.

KNN. After using the KMeans algorithm to label the data, we use the KNN algorithm to train the model and predict the data, the K nearest neighbor algorithm, given a training data set, for a new input instance, find the nearest neighbor K in the training data set an instance, most of these K instances belong to a certain category. The input instance is classified into this label. The formula used to calculate Euclidean distance is as follows.

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3}$$

Where x and y represent vector points in space. One of the main applications of the KMeans algorithm is to perform cluster analysis on data selected for specific operational purposes and divide the target group into several subdivided groups with distinct characteristics to provide better sales and service for these subdivided groups, therefore improving the efficiency of sales. This experiment aims to divide customer groups into different categories through KMeans clustering to realize the function of personalized recommendation games.

B. Aprior

To measure whether the item sets are frequent or not, we use three variables to describe this problem:

Support is defined as the proportion of transactions in the dataset which contain the item set. It shows the importance of the item set.

$$Support(x,y) = P(XY) = number(XY)/ number(AllSamples) \tag{4}$$

Confidence is equal to conditional probability.

$$Confidence(X \leftarrow Y) = P(X|Y) = P(XY)/P(Y) \tag{5}$$

Lift is equal to the ratio of Confidence to Support, and it is a normalization process. When the value is greater than 1, it is a valid strong association rule.

$$Lift(X \leftarrow Y) = P(X|Y)/P(X) = Confidence(X \leftarrow Y)/P(X) \tag{6}$$

Apriori is an iterative algorithm that uses a hierarchical search, using the data from the kth step to compute the $(k+1)\_th$ step. We use minimum Support as a super-parameter to start the iteration from the items set of size one to the end. Each iteration is divided into two steps. The first step is to form a new set of candidate sets by recombining items in all subsets of the previous step, with the set size increasing from k items to $(k+1)$ items. The second step is to prune all sets that do not satisfy the minimum support to obtain a bigger frequent subset. When the set generated in the $(k+1)\_th$ step is not a frequent subset, it returns to the kth layer, and the iteration ends.

After all the frequent item sets have been obtained by iteration, we select those item sets where the value of lift is greater than one. From this, we have gathered all the item sets that satisfy our requirements, and we recommend items in the same item sets to users.

C. Matrix Decomposition SVD

Matrix factorization is the process of decomposing a single matrix into two or more matrices, for instance: LU decomposition, PLU decomposition, and LDU decomposition, and so on. This method has better interpretability in the mathematical aspect than deep learning and neural networks in the recommender algorithm. Precisely, the deep learning and neural network method are more like a black box, which has better performance in some situations but cannot be explained and understood. However, the shortcomings are apparent in this recommender algorithm. The classical matrix decomposition

requires the number of rows and columns of the matrix to be equal, which means the matrix must be the square matrix. It is almost impossible to be satisfied with this algorithm. It can be used in the matrix but cannot be used in the matrix. In the meantime, the matrix decomposition method faces the sparsity and cold start problem in this algorithm. Therefore, the SVD is introduced to improve these problems. It has been proved to be a better solution in recommender algorithms.

$$R = U\Sigma V^T \tag{7}$$

The Singular Value Decomposition can factorize the $m \times n$ user's rating matrix R into the multiplication of three matrices, U, Σ, and V^T, where U is the $m \times m$ user feature unitary matrix, Σ is the $m \times n$ diagonal matrix of singular values, and V^T is the $m \times n$ game-feature unitary matrix.

SVD is welly applied in math, including computing the pseudoinverse, matrix approximation, and determining the rank, range, and null space of a matrix which is also extremely useful in all areas of science, engineering, and statistics, such as signal processing, least squares fitting of data, and process control.

D. Item-based collaborative filtering

Item-based collaborative filtering is based on the similarity between item information and users' interests. The algorithm assumes that item A and item B are very similar because most users like item A also like item B. There is an assumption that each user's interests are limited to a few areas. Suppose two items belong to the same user's interest list. In that case, both items may belong to a limited number of areas. Suppose two items appear in the interest lists of many users simultaneously. In that case, they may belong to the same domain and thus have a high degree of similarity by calculating the similarity between the user interest model and the item feature vector, then sending items with high similarity to users of that model [10].

Let $N(i)$ denote the number of users who like item i and denote the number of users who like items, at the same time [11]. Then the similarity between item and item is:

$$w_{ij} = \frac{|N(i) \cap N(j)|}{|N(i)|} \tag{8}$$

Since each user is independent and has an independent feature vector, there is no need to consider the interests of other users or how much the ratings are, and it can recommend new items or items that are unpopular to users. These advantages make item-based collaborative filtering recommender systems immune to cold start and data sparsity problems [12].

III. EXPERIMENT

In this paper, we use Kaggle's steam dataset to train our model. This dataset has four features, user ID, purchase history, play history and play time. We processed this dataset differently to fit our model.

A. Analysis of Feature Extraction Results Based on KMeans

After data cleaning, the first step is to find useful features from the data. By selecting and calculating the original data set, data that is more suitable for the KMeans model can be acquired. The data used for the clustering algorithm after selection is shown in Table 1.

TABLE I. DATA USED FOR CLUSTERING

Time	State	Number	Ave_time
This is the total amount of time played by the user.	The percentage of games owned by users that are purchased but not played.	The total number of games owned by the user.	Average time played by users.

The next step is to normalize the data by zscore algorithm. The reason for this is to reduce the magnitude of the impact between the data. The formula of zscore normalization is shown in the figure below.

Then use the elbow method to select the appropriate K value, and the picture after elbow method processing is shown in Figure 1.

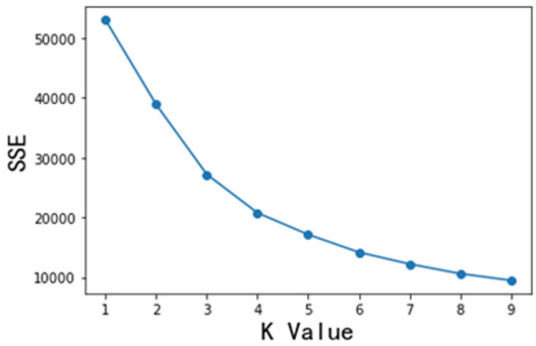

Figure 1. Result of the elbow method

From the figure of the elbow method, the value of K is 4 when it is the best clustering, so use 4 as the value of K for KMeans clustering. Then analyze the data obtained and adopt different recommendation strategies for different types of users.

Customer1. These users do not have many games, but the total game time and the average time per game are very long. This means that these users are more willing to spend a long time on a certain game instead of just having a simple experience. Their purpose of playing the game may not be simply to relax, but more likely to treat the game as a hobby or enjoy exploring every detail of the game. **Customer2.** All the data of these users are relatively weak, so these users are ordinary users. **Customer3.** The status value of these users is very high, which means that a large part of the games purchased by these users is not played. These users are more inclined to consume impulsively. They may not actually play the game after purchase, but more to follow the trend to buy. **Customer4.** These users have a large number of games, so although their total game time is relatively large, their average game time is not long. This means that the purpose of these customers buying games is more likely to be relaxation, so they only want to have a shallow experience of the game and are keen to try and try different games.

After that, the second part of the work is to model the data by KNN and establish labels. The main purpose of this is to check the effect of KMeans clustering. The second is to provide labels that are classified according to user behavior for the

following algorithms. According to the KNN algorithm, when the K value is 5, the precision, recall, and F score results are shown in Table 2.

TABLE II. THE PRECISION, RECALL, AND F1 SCORE OF KNN RESULT

Precision	Recall	F Score
0.96956	0.99201	0.99625

Finally, the results obtained by the KNN algorithm are visualized through TSNE.

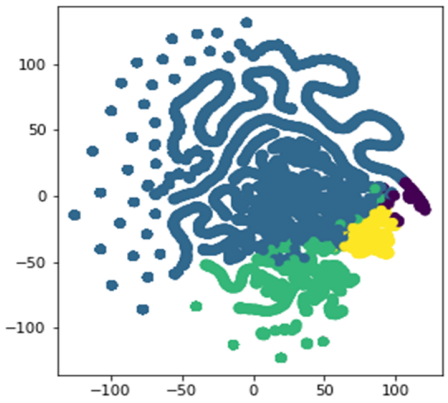

Figure 2. Visualization of KNN results.

According to the above image, the clustering effect is very good, and all the classified images have relatively clear boundaries.

B. *Using Apriori to do recommendations with or without KMeans classification*

In this model, two super-parameters need to be confirmed:

- **Max_size.** the maximum size of the history-play list of user's that we can drop out before starting the iteration;
- **Min_support.** the minimum support to prune the infrequent item sets.

These two super-parameters determine the training efficiency of this model and the accuracy of the prediction results. So we use Coverage and RMES as two dimensions to set our super-parameters and to evaluate Apriori mode. We use the user's time spent playing the game as the true rating of the game for RMES and the number of elements in the union of all frequent items as the Coverage.

To get a good recommended performance, we set Max_size to 9 and Min_support to 0.03 after using the 10-fold crossover method, which results in a low RMES of 0.01977, indicating that the algorithm has good prediction accuracy. However, for the Apriori algorithm, the RMES can only indicate the correctness of the recommended games but not wholly the goodness of the recommender system. This is due to the considerable variation in historical game sets per user, making the frequent itemset generated very small. This limits the number of games that can be recommended when recommending from the frequent itemset. In our experiments, the Coverage is only 86/3600. In that case, many users' favorite games are unlikely to be recommended by the recommendation system, and the recommender system cannot achieve the desired recommendation purpose.

Therefore, we defined a "recommender value" to illustrate the strengths and weaknesses of the Apriori recommender system. We use each game in the historical game set to recommend new games to users. The recommender value is the ratio of the number of games that match the games in the historical game set to the number of games in the historical game set for all new games recommended.

TABLE III. RECOMMENDER VALUE BEFORE AND AFTER CLUSTERING

	recommendation value			
Before clustering	0.08771			
After clustering	***Group 1***	***Group 2***	***Group 3***	***Group 4***
	0.07668	0.12522	0	0
Decrease Min_support to 0.01	0.09470			

After 10-fold cross-validation, as shown in Table 3, we found that the Apriori recommendation algorithm shows lower performance on the original dataset. This is an understandable result since the most important thing in a recommender algorithm is to have the ability to recommend different games to the user, which requires a sufficiently high Coverage. The Apriori algorithm removes a great number of games in the process of pruning the frequent itemset so that it can only cover a small but highly associated set of games in the end.

The process of getting the frequent itemset is for the whole dataset, so the categories of games we get with high association may be very different. For this reason, we use K-means clustering to classify users into different groups and then recommend them separately. However, it turns out that the Coverage is still low, and the dataset after user classification performs even worse than the dataset without user-based classification. This indicates that the Apriori algorithm is good at mining items with a strong association but unsuitable for user recommendation.

C. *Using SVD to verify Kmeans optimization performance*

In this paper, the raw data set is used as a baseline. The matrix is filled with 0 for the NULL elements. Although there are several ways of NULL elements filling, mean filling, or interpolation filling, these filling methods are not used in this case for analysis. In the latter part, the K-means clustering method will be used for the raw data to group the users into different groups, then proceed with the SVD method. The purpose of this step is to study whether the k-means clustering affects the performance of the baseline. The performance improvements include improving the density and lowering the prediction matrix's error to the original matrix so that the accuracy of prediction is also improved.

TABLE IV. RMSE OF PREDICTION BEFORE AND AFTER CLUSTERING

	RMSE			
Before clustering	1.85759			
	Group1	***Group2***	***Group3***	***Group4***
After clustering	0.00612	1.50474	0.86933	5.58810

For the result of Table 4, it is considered meaningful. The following reasons lead to this result, for the user's rating to a game is not given directly, instead use the time of playing. Generally, the factor time has an extensive and diverse range, so the time could not converge to a range with high density. By analyzing the range, the actual matrix is extremely low density (0.17%) because many zombie users and unpopular games are in the dataset. This situation causes the circumstance that: although there exist prevalent games and active players, which time is at a higher level than most of the elements in the matrix, a large proportion of playing time is still clinging to the 0 axes.

Meanwhile, the distribution pattern satisfies the elbow method, which is reasonable to cause large sparsity and diversity. In this paper, there are two solutions proposed to improve these problems. One way is to make popular game recommendations to the active user. Another is to use the k-means clustering method to group the users.

- Popular game recommendation to the active users

In this method, the users process fewer games, and the games with fewer users are cut and dropped from the raw data. By applying the elbow method, the key-value c1 and c2 considered the new boundary of the data set. After this data cleaning, the result of RMSE is 6.995149, which is considered reasonable in the active group. The clustering cut the group 0 and 2 from the original one.

The advantage of this method is evident, including resource-saving, especially for recommendation in the business application. It is unnecessary to pay much attention and resources to unpopular games and inactive users. The targeting is well improved, and the efficiency improved as well. Active users are more willing to buy and play games than inactive ones. The disadvantages are also apparent. After data cleaning, the polarization is worse, which means the popular games will be recommended more and more in the future. However, unpopular games will never be recommended to any users.

This method could be applied to a dataset or situation that has extensive enough data. Some of the data are acceptable to be abandoned, like the commercial or business area. In scientific research, or there are data preservation requirements, this method is not suitable to be applied.

- Use the K-means clustering method to group users.

This method narrows down the range of the time in each group and preserves the data of the inactive users (group 2,3). It is also possible to do clustering to the games. However, game clustering is not as valuable as game clustering. For the game has already been labeled and grouped clearly, this procedure is not meaningful. The RMSE of each group is as Table 4. For different groups, the RMSE also different

For the shortcomings of this method, they are mainly two parts. For the first part, this method cannot make recommendations across the group. Specifically, each group is a close circle. The recommendation can only be made within this circle. At the same time, the more groups are clustered, the smaller and simpler the group complexity, which means the meaning of the recommendation is lost. Therefore, how to choose the number of groups is crucial, and the principle of clustering is essential. The second part is that for some of the users, who cannot be accurately and identified into one group, in other words, one user can belong to groups 1 and 2 at the same time or cannot belong to any of the groups. To avoid the no-group belonging case, the number of groups must increase. Thus, the users can belong to more than one group, which is assumed to solve this problem.

In conclusion, the K-means clustering method before SVD has better performance than the baseline and better than the data cleaning method in this recommender algorithm.

D. Item-based collaborative filtering

First, create a user-to-item inverted index, as shown in Table 5. The following table shows that the rows represent users, the columns represent items, and 1 means the user likes the item.

TABLE V. INVERTED INDEX

Game to User	1	2	3	4	5
1	1	1		1	
2		1	1		1
3			1	1	
4		1	1	1	
5	1			1	

Then, create an item-to-item co-occurrence matrix that indicates the number of users who like two items simultaneously and is calculated from the inverted index, as shown in Table 6.

TABLE VI. ITEM-TO-ITEM CO-OCCURRENCE MATRIX

Game to Game	1	2	3	4	5
1		1		2	
2	1		2	2	1
3		2		2	1
4	2	2	2		
5		1	1		

Then, calculate the similarity from the similarity formula between items and recommend items to users based on their history.

For example, as shown in Table 7, the recommendation for user id ‘5250’ is listed below.

TABLE VII. RECOMMENDATION FOR USER ID '5250'

Game Name	Similarity
Left 4 Dead 2	1.0936
Unturned	0.5329
Borderlands 2	0.4284
Left 4 Dead	0.4182
Garry's Mod	0.3767
Killing Floor	0.2968
Warframe	0.2296

From the result, 'Left 4 Dead 2' has the highest similarity to games user id '5250' likes.

To test the accuracy of item-based collaborative filtering. We deleted one game data per user id randomly and calculated how many deleted games were recommended. As a result, 56% of deleted games were recommended, which means that users would purchase nearly half of the games in the recommender system.

IV. CONCLUSION

This paper has adopted traditional recommender algorithms as the research object and studied making recommendations in different situations. In the first part of this paper, K-means clustering is mainly used as the primary data processing method (by grouping users, the characteristics of users are extracted for grouping and recommendation, and the results before clustering are compared with those before clustering, it is found that in most cases, clustering users first can improve the quality of recommendation). In terms of results, the performance results of Apriori were not satisfactory, mainly because few games did not overlap between players. At the same time, Item-Base got unexpectedly good quality recommendation results. In principle, the Method of Apriori is to take the union of games between different players, that is, to find players who have played the same game. This method goes through multiple iterations. After simplification and abstraction, it will finally approach the item-based method itself, which only carries out one iteration level.

After the SVD, the comparison of the singular value decomposition before and after clustering is made. It is found that clustering can indeed improve the performance of matrix decomposition. The low RMSE of the original and prediction matrices indicates that the prediction accuracy does improve. The U and V matrices obtained from the decomposition of SVD and the method of getting the product prediction matrix to make recommendations can also be derived from the U and V matrices themselves to make game-game and player-player correlations. Game-game correlation is similar to the Apriori and item-based methods, which recommend similar games based on the games purchased by the player, and recommend games with high similarity. Player-player correlation was used where players are recommended to other players with closer preferences based on similarities between them. This paper shows that the traditional recommendation algorithm model does have a broader scope of application and the possibility of collaborative filtering in more game recommendation systems through experiments and analysis.

REFERENCES

[1] A. Singhal, P. Sinha, & R. Pant, (2017). Use of deep learning in modern recommendation system: A summary of recent works. arXiv preprint arXiv:1712.07525.

[2] L. Zheng, (2016). A survey and critique of deep learning on recommender systems. no. September, 31.

[3] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, & T. S. Chua, (2017, April). Neural collaborative filtering. In Proceedings of the 26th international conference on world wide web (pp. 173-182).

[4] R. Salakhutdinov, A. Mnih, & G. Hinton, (2007, June). Restricted Boltzmann machines for collaborative filtering. In Proceedings of the 24th international conference on Machine learning (pp. 791-798).

[5] Y. Zheng, B. Tang, W. Ding, & H. Zhou, (2016, June). A neural autoregressive approach to collaborative filtering. In International Conference on Machine Learning (pp. 764-773). PMLR.

[6] R. Devooght, & H. Bersini, (2016). Collaborative filtering with recurrent neural networks. arXiv preprint arXiv:1608.07400..

[7] B. Hidasi, A. Karatzoglou, L. Baltrunas, & D. Tikk, (2015). Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939.

[8] Z. Xu, C. Chen, T. Lukasiewicz, Y. Miao, & X. Meng, (2016, October). Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (pp. 1921-1924).

[9] L. Rice, E. Wong, & Z. Kolter, (2020, November). Overfitting in adversarially robust deep learning. In International Conference on Machine Learning (pp. 8093-8104). PMLR.

[10] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, & T. S. Chua, (2017, April). Neural collaborative filtering. In Proceedings of the 26th international conference on world wide web (pp. 173-182).

[11] G. Linden, B. Smith, & J. York, (2003). Amazon. com recommendations: Item-to-item collaborative filtering. IEEE Internet computing, 7(1), 76-80.

[12] A. Pujahari, & V. Padmanabhan, (2015, December). Group Recommender Systems: Combining user-user and item-item Collaborative filtering techniques. In 2015 International Conference on Information Technology (ICIT) (pp. 148-152). IEEE.