International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Novel Serendipitous Recommender System using Relevance Scores for Long Tail Items

Saurabh Tandel*[1], Keyur Rana[2]

Submitted: 28/08/2023 **Revised**: 21/10/2023 **Accepted**: 01/11/2023

Abstract*:* Assisting users to aid in decision making while doing e-commerce purchases is a primary task of a traditional Recommender System. But often there is a necessity to give equitable importance to the products somehow ignored by the traditional Recommender Systems. Drawing less number of ratings from the users of the system should not be the reason to make the item fall into the long tail list of non-popular Items. To overcome such Recommender System issues of 'Long Tail' and 'Popularity Bias', we are proposing a new Serendipitous Recommender System using k nearest neighbor approach coupled with the Bhattacharyya coefficient to recommend not only novel but also at the same time, the relevant set of Items out of the long tail list of non-popular Items**.**

Keywords: *Recommender System, Serendipity, Long Tail items, Non Popular items, Novelty Score, Bhattacharyya coefficient, k Nearest Neighbor*

1. Introduction

Recommender Systems can be defined as programs which attempt to recommend the most suitable items (products or services) to particular users (individuals or businesses) by predicting a user's interest in an item based on related information about the items, the users and the interactions between items and users [1-4]. To provide genuine recommendations to a user so that the suggested items or products are offering the utmost satisfaction should be given the priority while designing any Recommender System [5-6]. There are plenty of Recommender Systems available in the literature. But the items offered as recommendations by the majority of the Recommender Systems do have the tendency to recommend popular or easily identifiable or routine items [7-8].

Because these offerings by the majority of the Recommender Systems lack the components of novelty and serendipity, such Recommender Systems end up facing the issues of 'Popularity Bias', ignorance of the 'Long Tail' items and 'Matthew Effect' etc [5-12]. Because of such shortfalls of the traditional Recommender Systems, the products which are popular in the catalog have the tendency to gain even more popularity and contribute to the ever expanding lengthy list of the 'Long Tail' of 'Non-Popular' Items, waiting to be recommended for the endless time, leading towards the starvation [5-10].

It has been noticed many times that some Items get exaggerated popularity, while the long list of multiple Items keep striving to attain user awareness and user preferences. This issue is better known as 'Popularity Bias' or 'Matthew Effect' in literature [5-8], [11-12] as also mentioned in the following Fig. 1.

[1]*Assistant Professor, Computer Engineering Department,*
C K Pithawala College of Engg. & Tech. Surat, Gujarat, India &
Research Scholar, Gujarat Technological University, Ahmedabad
https://orcid.org/0000-0001-5094-8586
[2]*Professor, Computer Engineering Department,*
Sarvajanik College of Engg. & Tech., Surat, Gujarat, India
**Corresponding Author Email: saurabh.tandel@ckpcet.ac.in*

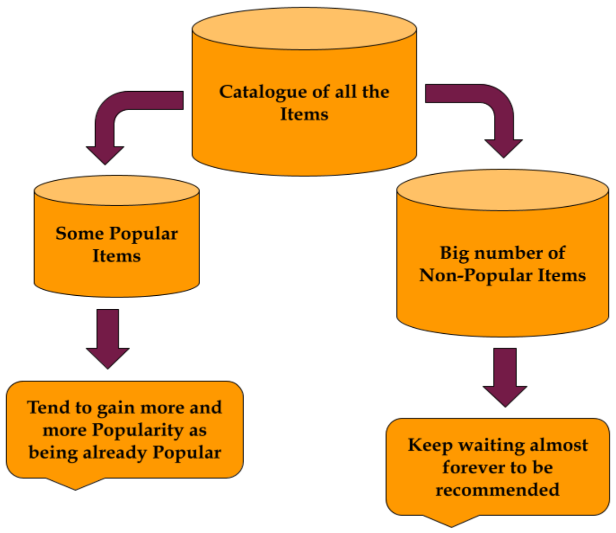
Catalogue of all the Items
Some Popular Items
Big number of Non-Popular Items
Tend to gain more and more Popularity as being already Popular
Keep waiting almost forever to be recommended

Fig. 1. Popularity Bias Explanation

Moreover, there is always a possibility of introducing new or niche items better known as 'Cold Start Items' in the overall catalog of products [13-14]. Now, to make such new or niche items forming the 'Long Tail' catch the attention of the users of the system, there is always a need to improvise the system so that every product in the entire catalog gets the equitable attention and identification [9-10].

The remainder of the paper is organized as follows: Section II reviews the relevant work; the proposed work is presented in Section III; Section IV is all about the experiments; we present results and analysis in Section V and we conclude the paper in Section VI.

2. Related Work

We mostly find the Recommender Systems providing suggestions/recommendations striving for accuracy and closeness with the user profiles [3]. But, as it is found in many past research papers that in order to fulfill the criterion of accuracy and closeness with the user profiles, the system keep recommending the items which may not always be ensuring the usefulness for the user and hence, being unable to justify the more important parameter of user satisfaction and ultimately, contributing little in terms of helping users while decision making [5-7].

To put good weightage on the aspect of user satisfaction, a matrix of novelty has been coined many times in the literature for enhancing the usefulness and effectiveness of the recommendations [15-16].

As per [13-14], most of the items in the recommendation list are already familiar to users and therefore the performance would seriously degenerate in finding cold items, i.e., new items and niche items. To address this issue of recommending cold items, they introduce the concept of Innovators, who are a special subset of users who can discover cold items without the help of a recommender system for achieving the balance between serendipity and accuracy [20]. Another approach, when applied to 'Long Tail' items, enables the system designer to tune the application to achieve a particular tradeoff between ranking accuracy and the inclusion of long-tail items [9-10].

To increase the participation of 'Long Tail' items, diversification of recommendations in a personalized manner is suggested and the recommendation list is optimized based on three objectives of increasing the accuracy, personalizing the diversity, and reducing the popularity of the recommended items [10]. In [6], the modified xQuAD method is proposed to produce a new re-ranked list that manages popularity bias while still being accurate. In [19], Zhang et. al. have proposed a serendipitous music recommendation framework named auralist, a method of successfully injecting serendipity, novelty and diversity into recommendations whilst limiting the impact on accuracy.

Onuma et. al. [17] in their work have proposed TANGENT, a recommendation algorithm that takes into account the connectivity to other groups in order to broaden users' horizons. It is based on the graph mining technique of computing similarity between nodes and can be applied to any dataset that can be represented as a graph. Adamopoulos et. al. [18] have proposed a novel method for generating unexpected recommendations to take into account the actual expectations of the users and suggested some metrics to measure the multifaceted concept of unexpectedness of recommendation lists.

Introduction of novelty brings into horizon the vast set of products that are usually unseen, unexplored or ignored because of being non-popular while generating the recommendations. There is always a need for the components of novelty and relevance contributing to the serendipity so that the 'Long Tail' products get noticed and user interaction with the system leads to more and more user satisfaction [8-9].

This research paper introduces a novel approach to address the above mentioned issues of 'Long Tail', 'Popularity Bias', 'Matthew Effect' and 'Cold Start Items' by employing the superior Bhattacharyya coefficient coupled with the knn(k Nearest Neighbour) approach for finding the set of unexplored yet relevant and novel Recommendations for all the regular set of users. Out of the Top N recommendations delivered by the algorithm, a novelty metric is used as a yardstick to judge the effectiveness of the suggested recommendations.

For the occasional users having the rating count less than the system defined threshold, any of the traditional recommendation algorithms such as Collaborative, Content Based, Hybrid, etc. [21] can be applied to generate the recommendations.

3. Proposed Method

In this section, we are going to explain various terms relevant to the proposed algorithm framework for recommendation as we keep mentioning different phases of the algorithm. For the implementation purpose, we have considered MovieLens Dataset [30]. The dataset aims to collect different users' reviews on a scale of 1-5, for multiple movies watched by them.

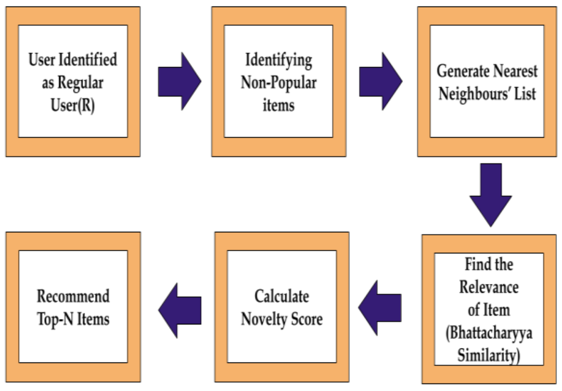
User Identified as Regular User(R)
Identifying Non-Popular items
Generate Nearest Neighbours' List
Find the Relevance of Item (Bhattacharyya Similarity)
Calculate Novelty Score
Recommend Top-N Items

Fig. 2. Proposed System Flow

3.1. Identifying Regular (R) or Occasional (O) user

Based on how regularly or frequently the users are watching the movies and hence, subsequently rating the movies, the users can be categorized into two broad categories of (i) Regular (R) or frequent users and (ii) Occasional users (O).

To categorize the user as a Regular or frequent User, we have considered the user threshold (USER_THRESHOLD), i.e., minimum number of movies the user has rated, as a measure to qualify for the category. Out of the above mentioned 2 categories, we are concentrating only for the Regular or frequent users as more interactions, i.e., more movie reviews by a user are paving the way to provide justifiable analysis.

Whereas, for the Occasional users, any of the traditional Recommender Systems, such as Content Based Filtering(CBF), Collaborative Filtering(CF), Context Aware Filtering(CAF), etc. can be applied and recommendations can be suggested.

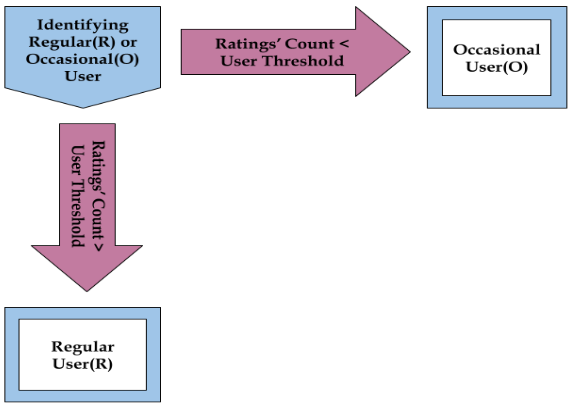
Identifying Regular(R) or Occasional(O) User
Ratings' Count < User Threshold
Occasional User(O)
Ratings' Count > User Threshold
Regular User(R)

Fig. 3. User Identification as Regular (R) or Occasional (O)

3.2. Identifying Non-Popular Items for target user 'u'

After the user has been appropriately categorized as a Regular or frequent user in phase - 'A', this phase ensures that the user is only made available with the Items suffering from the issues of 'Long Tail' or 'Non-Popularity Bias'. Movies falling under the 'Long

Tail' are those movies which are unintentionally ignored by the Regular users (R) of the MovieLens or any such system.

To do a proper justification to the categorization of Popular and Non-Popular Items, we are finding the ratings' count, i.e., number of times a particular Item (Movie) has been rated by all the users involved in the movie rating system. So, the movies with the ratings' count less than the system parameter named as movie threshold (MOVIE_THRESHOLD) are considered to be the non-popular movies responsible for framing the Long Tail of such movies.

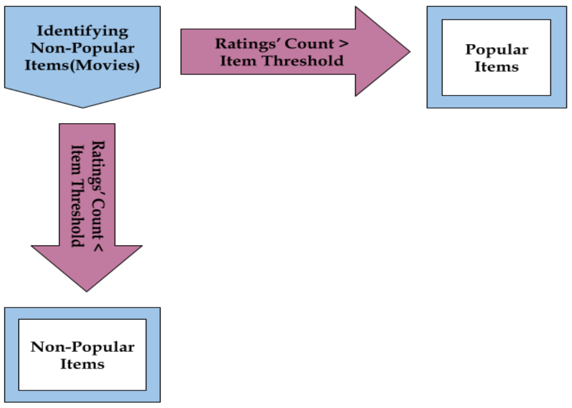
Identifying Non-Popular Items(Movies)
Ratings' Count > Item Threshold
Popular Items
Ratings' Count < Item Threshold
Non-Popular Items

Fig. 4. Item Identification as Popular or Non-Popular

The movies with the ratings' count more than the item threshold are considered to be the popular movies. And for the movies which are already popular and hence not suffering from the 'Popularity Bias' are not taken into the consideration by the proposed system.

3.3. Generate Nearest Neighbours' List for the target user 'u'

We generate the set of k Nearest Neighbours for the target user out of the Regular users of the system identified in phase - 'A'. The value of k (NEAR_USER_COUNT) can range from 10 to 50. For our implementation, we have considered the default value of k = 20, i.e., 20 Nearest Neighbors.

3.4. Find the Relevance of Item 'i' for the target user 'u' using Bhattacharyya coefficient

It is very important to suggest recommendations which are having definite Relevance with the target user. So, in this phase, we find the relevance of item(movie) 'i' for the target user 'u' by a modified linear combination of target user's neighbors ratings [22-23], i.e.,

$$RS(u,i) = \underline{r}(u) + Z * \sum_{n \epsilon N_k} \quad BC(u,n)(r(n,i) - \underline{r}(n)) \quad (1)$$

Here,

RS(u,i) = Relevance of non-popular item 'i' for user 'u'

$\underline{r}(u)$ = Average rating by target user 'u'

Z = Normalized Constant

BC(u,n) = Bhattacharyya coefficient between two users, 'u' and 'n'; explained after the next paragraph

N_k = Set of 'k' neighboring users for the target user 'u'

r(n,i) = Rating of item 'i' by user 'n'

$\underline{r}(n)$ = Average rating by user 'n'

Finding the relevance of a movie for the user is very vital as any movie without being relevant to the user is of no practical usage and can not draw any satisfaction from the user. So, technically this component is very much an inevitable part of any effective Recommender System.

3.4.1. How to compute BC (Bhattacharyya coefficient)?

Traditional similarity measures such as Cosine Distance, Jaccard Distance, Pearson Distance, etc. consider ratings of only co-rated items. Many times the datasets contain sparse data, which can't be properly utilized by traditional measures. Hence, the Bhattacharyya coefficient [24-27] has been used to calculate the similarity of the target user 'u' with the neighboring user 'v' to find proper similarity even if the items(movies) have not been co-rated.

3.4.2. Example: Finding Bhattacharyya coefficient

In order to find the Bhattacharyya coefficient, we must compute the Bhattacharyya coefficient (BC) first.

Calculation of BC:

I = (1,0,2,0,1,0,2,0,3,0) & **J** = (0,1,0,2,0,1,0,2,0,3)

$$BC(I, J) = \sum_{r=1}^{n} \quad \sqrt{IrJr}$$

$$= \sqrt{(2/5) * (2/5)} + \sqrt{(2/5) * (2/5)} + \sqrt{(2/5) * (2/5)}$$

$$= 1$$

After finding the relevance scores (RS) as per Eq. (1) above of all the items(movies), we consider only those movies with the relevance score higher than that of the system defined parameter RELEVANCE_THRESHOLD to consider for the next phase - 'E'.

3.5. Calculate Novelty Score

For all the items (movies) identified in this last phase, generate Novelty Scores(NS). NS is a second component of serendipity after relevance that has been computed in the last phase. Now, in order to compute the Novelty Score, the popular information retrieval measure of TF-IDF (Term Frequency - Inverse Document Frequency) [28-29] has been modified as IF-IUF (Item Frequency - Inverse User Frequency).

$$NS(i) = IF(i) * IUF(i) \quad (2)$$

Here, NS(i) = Novelty Score of item 'i'

Item Frequency:

$$IF(i) = \frac{No\ of\ total\ ratings\ of\ item\ 'i'}{Max\ no\ of\ ratings\ received\ by\ any\ item}$$

Inverse User Frequency:

$$IUF = log(\frac{Total\ no\ of\ users}{No\ of\ users\ who\ have\ rated\ item\ 'i'})$$

3.6. Recommend Top N Items

After arranging the items (movies) from the last phase in the descending order, we consider only Top N movies to recommend for the target user 'u'. N can accept the values of 10 or 20.

4. Experiments

4.1. Dataset

For analysis purposes, we have used the trademark MovieLens Dataset. Following are the statistics for the same.

MovieLens(ml-latest-small) [30]:

Name: MovieLens-Latest-Small

Number of Users: 610

Number of Ratings: 100836

Number of Movies: 9742

Rating Scale: 1 to 5

4.2. Evaluation Metric

The Evaluation Metric [21] for finding the novelty scores of the movies generated from the proposed algorithm to overcome the ‘Popularity Bias’ and falling under the tag of ‘Long Tail’ is as follows:

$$Nov_d(i,u) = \frac{1}{|I_u|} \sum_{j \in I_u} dist(i,j)$$

(3)

Here,

$Nov_d(i,u)$ = Novelty Score of movie ‘i’ for user ‘u’

I_u = List of all movies user ‘u’ has rated

dist(i,j) = Distance between movie ‘i’ and movie ‘j’

= 1 - sim(i,j)

sim(i, j) is any kind of similarity(Cosine/Jaccard/Pearson) between movies i and j & $sim(i, j) \in [0, 1]$

This Novelty Metric equation will give us a measure to evaluate how novel the movies are for the target user, presenting the testimony to judge the effectiveness of the Proposed System.

5. Results and Analysis

5.1. User Recommendation Count Analysis

The following graph is plotted to indicate the number of recommendations generated for the regular users to provide a glimpse of the user recommendation count distribution for MovieLens dataset.

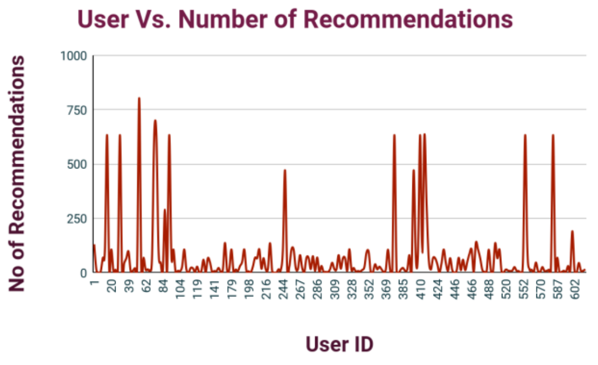

Fig. 5. User Vs Number of Recommendations (Movielens Dataset)

Regarding the user recommendation count graph for the MovieLens Dataset, following Table 1 shows the statistics to better interpret the same.

Table 1. User recommendation Count Statistics (MovieLens Dataset)

Parameter	Value
Number of Regular Users	230
Average Ratings	3.70
Highest Rating Count by any Regular User	2108
Lowest Rating Count by any Regular User	56
Average Rating Count by any Regular User	255
Highest No of Recommendations	803
Lowest No of Recommendations	1
Average No of Recommendations	71

Following Table 2 represents the randomly selected recommendation statistics of the MovieLens dataset generated from the proposed system in terms of user id, average ratings by that user id, no. of movies rated by the user id, total no. of recommendations generated and top 10 movie recommendations in terms of novelty scores.

Table 2. Sample Recommendations (MovieLens Dataset)

User ID	Average Ratings	No of Movies Rated	No of Recommen-dations	Top 10 Recommendations (Movie Ids)
1	4.37	232	131	3622, 3692, 3056, 295, 2037, 3400, 1583, 3662, 3797, 118
29	4.14	81	633	4458, 5422, 5071, 52784, 52668, 5178, 5237, 115828, 5240, 5285
211	3.90	89	108	31909, 3857, 26736, 8241, 8225, 30745, 60291, 44929, 7930, 7646
610	3.69	1302	16	3494, 2202, 3066, 3022, 3334, 2921, 1284, 3030, 3836, 1243

5.2. Novelty Score Analysis

5.2.1. Novelty Score Analysis(Default System parameters)

The following diagram in Fig. 6 depicts the average novelty scores generated for all the regular users, based on the novelty metric Eq. (2), with the default values of the system parameters mentioned in the Table 3 below.

Table 3. Default values of system parameters

System Parameter Name	Cutoff Value
USER_THRESHOLD	50
MOVIE_THRESHOLD	25
NEAR_USER_COUNT	20
RELEVANCE_THRESHOLD	4.5

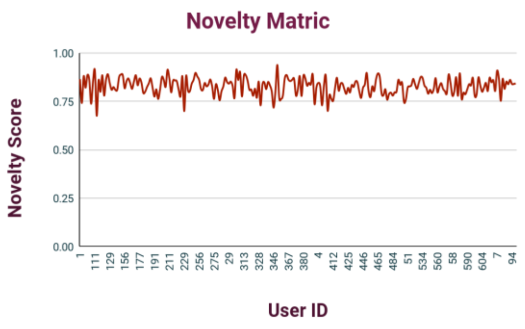

Fig. 6. Novelty Scores for the Regular Users (Movielens Dataset)

For the novelty score graph for the MovieLens dataset in Fig. 6 above, following are the statistics to better understand the overall novelty scores.

Table 4. Novelty Score Statistics (Movielens Dataset)

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
235	1.000	0.475	0.829

Now, to assess the impact of variations in the system parameters, following six different combinations of the system parameters have been attempted and plotted.

5.2.2. Novelty Score Analysis(System parameters - Variant 1)

Table 5. Modified values of the system parameters - Variant 1

System Parameter Name	Cutoff Value
USER_THRESHOLD	20
MOVIE_THRESHOLD	40
NEAR_USER_COUNT	20
RELEVANCE_THRESHOLD	4.5

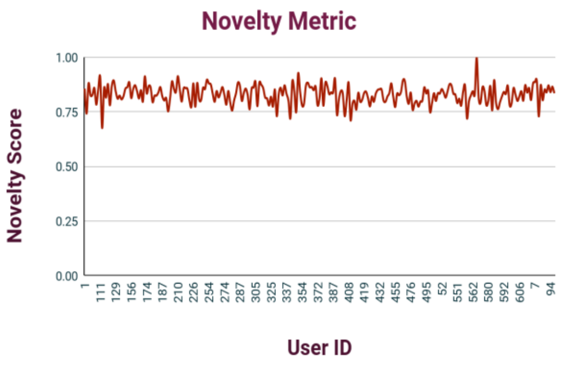

Fig. 7. Novelty Scores for the Regular Users (Movielens Dataset) Variant - 1

Table 6. Novelty Score Statistics (MovieLens Dataset) Variant -1

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
243	1.00	0.470	0.830

5.2.3. Novelty Score Analysis(System parameters - Variant 2)

Table 7. Modified values of the system parameters - Variant 2

System Parameter Name	Cutoff Value
USER_THRESHOLD	40
MOVIE_THRESHOLD	15
NEAR_USER_COUNT	10
RELEVANCE_THRESHOLD	4.5

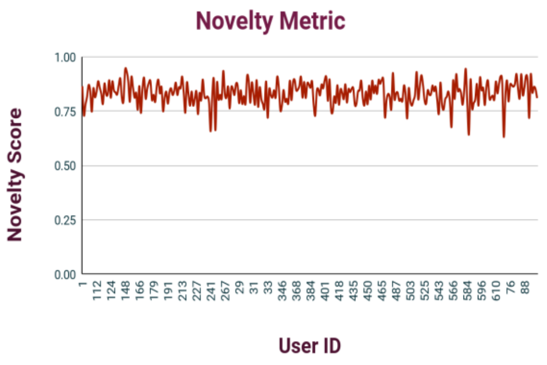

Fig. 8. Novelty Scores for the Regular Users (Movielens Dataset) Variant - 2

Table 8. Novelty Score Statistics (MovieLens Dataset) Variant -2

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
288	1.00	0.474	0.831

5.2.4. Novelty Score Analysis(System parameters - Variant 3)

Table 9. Modified values of the system parameters - Variant 3

System Parameter Name	Cutoff Value
USER_THRESHOLD	50
MOVIE_THRESHOLD	20
NEAR_USER_COUNT	100
RELEVANCE_THRESHOLD	4.5

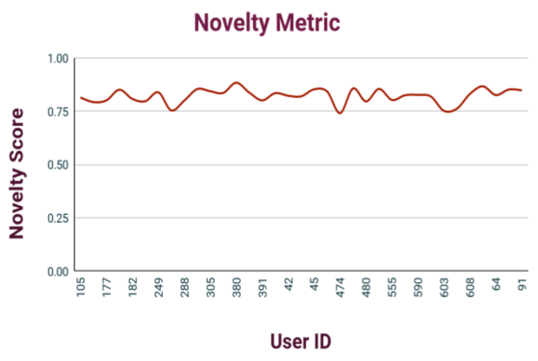

Fig. 9. Novelty Scores for the Regular Users (Movielens Dataset) Variant - 3

Table 10. Novelty Score Statistics (MovieLens Dataset) Variant - 3

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
35	1.00	0.610	0.821

5.2.5. Novelty Score Analysis(System parameters - Variant 4)

Table 11. Modified values of the system parameters - Variant 4

System Parameter Name	Cutoff Value
USER_THRESHOLD	50
MOVIE_THRESHOLD	25
NEAR_USER_COUNT	10
RELEVANCE_THRESHOLD	4.5

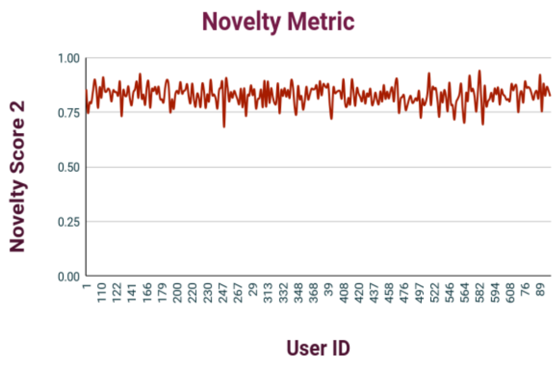

Fig. 10. Novelty Scores for the Regular Users (Movielens Dataset) Variant - 4

Table 12. Novelty Score Statistics (MovieLens Dataset) Variant - 4

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
277	1.00	0.470	0.829

5.2.6. Novelty Score Analysis(System parameters - Variant 5)

Table 13. Modified values of the system parameters - Variant 5

System Parameter Name	Cutoff Value
USER_THRESHOLD	50
MOVIE_THRESHOLD	25
NEAR_USER_COUNT	20
RELEVANCE_THRESHOLD	4.5

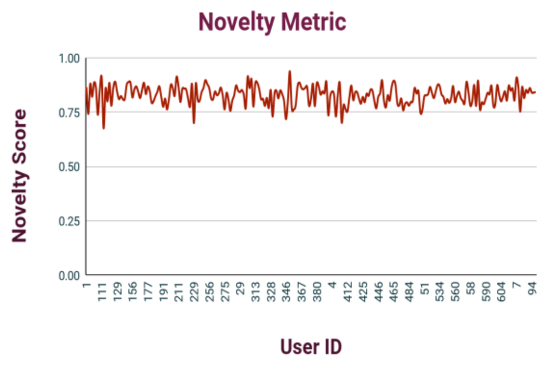

Fig. 11. Novelty Scores for the Regular Users (Movielens Dataset) Variant - 5

Table 14. Novelty Score Statistics (MovieLens Dataset) Variant - 5

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
235	1.00	0.470	0.829

5.2.7. Novelty Score Analysis(System parameters - Variant 6)

Table 15. Modified values of the system parameters - Variant 6

System Parameter Name	Cutoff Value
USER_THRESHOLD	50
MOVIE_THRESHOLD	25
NEAR_USER_COUNT	20
RELEVANCE_THRESHOLD	4.5

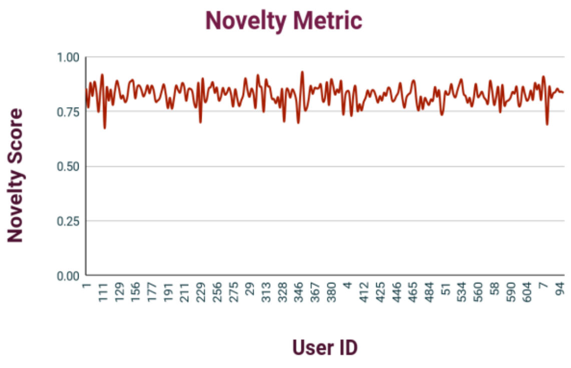

Fig. 12. Novelty Scores for the Regular Users (Movielens Dataset) Variant - 6

Table 16. Novelty Score Statistics (MovieLens Dataset) Variant - 6

No of Users	Highest Novelty Score	Lowest Novelty Score	Average Novelty Score
235	1.00	0.470	0.827

From the observation of all the novelty metric graphs mentioned above i.e., with the default system parameters (Fig. 6) and variations in default system parameters (Fig. 7-12), it is noteworthy that the novelty score evaluation metric generates the average Novelty Scores for the MovieLens dataset with the Average Scores of 0.83. It means that the recommended movies are 83 % novel to the users of the MovieLens dataset.

The lowest Average Novelty Scores for all the scenarios range from 0.470 to 0.610 i.e., the movies recommended because of the algorithm are at least 47 % to 61 % novel. And the highest Average Novelty Score for all the scenarios is 1.000 i.e., the movies recommended because of the algorithm are 100 % novel. So, in a way, this metric provides the testimony for the effectiveness of the proposed system.

6. Conclusions

In this paper, we addressed the challenges of 'Popularity Bias', 'Long Tail', 'Matthew Effect' and 'Cold Start Items' pertaining to the traditional Recommender Systems. For addressing and overcoming these challenges, the component of Serendipity has been introduced to target specifically the movies which are facing the 'Starvation' issue, i.e., movies forming the 'Long Tail' of the non-popular movies are waiting almost for over to be recommended to any user.

In the proposed system, we have considered the Bhattacharyya coefficient, which is quite superior as compared to the other similarity measures, specifically for the scattered values in the datasets. Also, equitable significance was devoted for the relevance scores to confirm that the movies being recommended are of definite interest to the target user. Another Serendipity component of novelty after relevance is given enough emphasis to ensure that the movies Recommended from the proposed algorithm are not only relevant and meaningful, but at the same time, novel also. Hence, both the components of relevance and novelty contribute effectively to make the proposed system highly novel and serendipitous.

To verify the overall effectiveness of the proposed system, the novelty score evaluation metric was introduced and calculated to offer the 83 % average Novelty for the MovieLens dataset.

References

[1] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," Knowledge-Based Systems, vol. 46, pp. 109–132, Jul. 2013, doi: https://doi.org/10.1016/j.knosys.2013.03.012.

[2] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: A survey," Decision Support Systems, vol. 74, pp. 12–32, Jun. 2015, doi: https://doi.org/10.1016/j.dss.2015.03.008.

[3] D. Kotkov,, J. Veijalainen, and S. Wang, "Challenges of serendipity in recommender systems," presented at the *International conference on web information systems and technologies(WEBIST 2016)*, vol. 2, pp. 251-256, Rome, Italy, April 23–25, 2016, doi: https://doi.org/10.5220/0005879802510256.

[4] B. B. Sinha and R. Dhanalakshmi, "Evolution of recommender system over the time," Soft Computing, vol. 23, no. 23, pp. 12169–12188, Jun. 2019, doi: https://doi.org/10.1007/s00500-019-04143-8.

[5] H. Abdollahpouri, R. Burke, and B. Mobasher, "Controlling Popularity Bias in Learning-to-Rank Recommendation," Proceedings of the Eleventh ACM Conference on Recommender Systems, Aug. 2017, doi: https://doi.org/10.1145/3109859.3109912.

[6] H. Abdollahpouri, R. Burke, and B. Mobasher, "Managing popularity bias in recommender systems with personalized re-ranking."presented at *The Thirty-Second International Florida Artificial Intelligence Research Society Conference (FLAIRS-32)*, May 18–22, 2019, Sarasota, Florida, USA, doi: https://doi.org/10.48550/arXiv.1901.07555.

[7] H. Abdollahpouri, "Popularity Bias in Ranking and Recommendation," Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, Jan. 2019, doi: https://doi.org/10.1145/3306618.3314309.

[8] C. Sun and Y. Xu, "Topic Model-Based Recommender System for Longtailed Products Against Popularity Bias," Jun. 2019, doi: https://doi.org/10.1109/dsc.2019.00045.

[9] M. Levy, and K. Bosteels, "Music recommendation and the long tail," presented at *the workshop on music recommendation and discovery (WOMRAD)*, pp. 55-58, Barcelona, SPAIN, 2010, doi:

[10] E. Malekzadeh Hamedani and M. Kaedi, "Recommending the long tail items through personalized diversification," Knowledge-Based Systems, vol. 164, pp. 348–357, Jan. 2019, doi: https://doi.org/10.1016/j.knosys.2018.11.004.

[11] C. Gao et al., "Alleviating Matthew Effect of Offline Reinforcement Learning in Interactive Recommendation," Jul. 2023, doi: https://doi.org/10.1145/3539618.3591636.

[12] M. Perc, "The Matthew effect in empirical data," Journal of The Royal Society Interface, vol. 11, no. 98, p. 20140378, Sep. 2014, doi: https://doi.org/10.1098/rsif.2014.0378.

[13] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations." presented at *the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 253-260, August 11-15, 2002, in Tampere, Finland, doi: https://doi.org/10.1145/564376.564421.

[14] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," Expert Systems with Applications, vol. 41, no. 4, pp. 2065–2073, Mar. 2014, doi: https://doi.org/10.1016/j.eswa.2013.09.005.

[15] L. Zhang, "The Definition of Novelty in Recommendation System," Journal of Engineering Science and Technology Review, vol. 6, no. 3, pp. 141–145, Jun. 2013, doi: https://doi.org/10.25103/jestr.063.25.

[16] S. Vargas, and P. Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," presented at *the fifth ACM conference on Recommender systems*, pp. 109-116, Oct. 2011, Chicago, IL, USA, doi: https://doi.org/10.1145/2043932.2043955.

[17] K. Onuma, H. Tong, and Christos Faloutsos, "TANGENT," Jun. 2009, doi: https://doi.org/10.1145/1557019.1557093.

[18] P. Adamopoulos and A. Tuzhilin, "On Unexpectedness in Recommender Systems: Or How to Expect the Unexpected.," pp. 11–18, Jan. 2011.

[19] Y. C. Zhang, D. Ó. Séaghdha, D. Quercia, and T. Jambor, "Auralist," Proceedings of the fifth ACM international conference on Web search and data mining - WSDM '12, 2012, doi: https://doi.org/10.1145/2124295.2124300.

[20] C.-D. Wang, Z.-H. Deng, J.-H. Lai, and P. S. Yu, "Serendipitous Recommendation in E-Commerce Using Innovator-Based Collaborative Filtering," IEEE Transactions on Cybernetics, vol. 49, no. 7, pp. 2678–2692, Jul. 2019, doi: https://doi.org/10.1109/tcyb.2018.2841924.

[21] D. Kotkov, S. Wang, and J. Veijalainen, "A survey of serendipity in recommender systems," Knowledge-Based Systems, vol. 111, pp. 180–192, Nov. 2016, doi: https://doi.org/10.1016/j.knosys.2016.08.014.

[22] A. Bellogín and P. de, "Understanding Similarity Metrics in Neighbour-based Recommender Systems," Sep. 2013, doi: https://doi.org/10.1145/2499178.2499186.

[23] A. Bellogín and J. Parapar, "Using graph partitioning techniques for neighbour selection in user-based collaborative filtering," Jan. 2012, doi: https://doi.org/10.1145/2365952.2365997.

[24] B. Kr. Patra, R. Launonen, V. Ollikainen, and S. Nandi, "A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data," Knowledge-Based Systems, vol. 82, pp. 163–177, Jul. 2015, doi: https://doi.org/10.1016/j.knosys.2015.03.001.

[25] K. G. Derpanis, "The Bhattacharyya Measure," Jan. 2008.

[26] G. Xuan, P. Chai, and Wu Minhui, "Bhattacharyya distance feature selection," Jan. 1996, doi: https://doi.org/10.1109/icpr.1996.546751.

[27] H. Cao, J. Deng, H. Guo, B. He, and Y. Wang, "An improved recommendation algorithm based on Bhattacharyya Coefficient," Sep. 2016, doi: https://doi.org/10.1109/ickea.2016.7803027.

[28] Sonia Ben Ticha, Azim Roussanaly, A. Boyer, and Khaled Bsaïes, "Feature Frequency Inverse User Frequency for Dependant Attribute to Enhance Recommendations," Nov. 2013.

[29] S.-W. Kim and J.-M. Gil, "Research paper classification systems based on TF-IDF and LDA schemes," Human-centric Computing and Information Sciences, vol. 9, no. 1, Aug. 2019, doi: https://doi.org/10.1186/s13673-019-0192-7.

[30] "MovieLens 100k Dataset | GroupLens," Available: https://grouplens.org/ datasets/movielens/. [Accessed: Jun. 03, 2023]