EasyChair Preprint

№ 9601

Study of Cold-Start Product Recommendations and Its Solutions

Deep Pancholi and C. Selvi

January 20, 2023

Study of Cold-start Product Recommendations and its Solutions

Deep Pancholi[1] and Selvi C.[2]

[1] Amrita Vishwa Vidyapeetham, Coimbatore, India
cb.en.p2cse20011@cb.students.amrita.edu
[2] Indian Institute of Information Technology, Kottayam, Kerala
selvic@iiitkottayam.ac.in

Abstract. In today's digital era, consumers rely more and more on the systems that provide them with a personalized experience. In interacting with the system, these consumers create more and more data of different types (click-through rates, items viewed, time spent, number of purchases, and other metrics.). This extensive collection of data from various users is used only to improve the personalizing experience of the users. These systems that utilize consumers' data to create a more personalized and customized user experience are called Recommender Systems. Recommender Systems play a huge role in helping companies create a more engaging user experience. E-Commerce giants like Amazon and Flipkart employ such Recommender Systems. These can learn from the user-system interaction the likes and dislikes of users and can promote the visibility of items that interest the user. They are also helpful in luring the customer to buy those things he would have to search for manually in the absence of such a system, which can recommend the item to a user based on his previous interactions. Streaming services like YouTube, Amazon Prime Video and Netflix also use Recommender Systems to suggest movies/shows that the user might like based on the watch history. This study proposes a hybrid model with item-item collaborative filtering using a graph, user-user collaborative filtering based on textual reviews and ratings, and demographic data to generate accurate product recommendations that address the cold-start issue.

Keywords: recommender systems, collaborative filtering, cold-start issue, hybrid model, item-item graph, user-user collaborative filtering

1 Introduction

The importance of recommender systems in any market segment where customer interaction happens is increasing daily with the increase in data that people are generating using intelligent devices and systems linked with the web. Thus, any company or firm must utilize the customer-system interaction to maximize the profits as much as possible. Recommendations are generated using many different approaches and algorithms, most of which can be studied as research problems, making this field very

intriguing. This paper further explores one such segment in this area: Product Recommender Systems using Graph Data Structures.

1.1 Non-personalized recommender systems

Non-personalized recommendations refer to suggestions or recommendations which are not meant for one single user or type of user. They are comparatively easy to generate, and any prior information about the users is not required to be known. The recommendations generated, in turn, are more general than specific to users. These include showing the most popular products in each category, the most ordered items in a specific time window in the recent past, and many more methods.

1.2 Personalized recommender systems

Personalized recommendations are those generated using data about one user or a group of similar users. These require data about each user to be known beforehand and are challenging to generate compared to non-personalized recommendations. The recommendations generated are specific to users as the data used was also specific. So, these include showing any user what he would like to purchase next based on his order history or watch history. Personalized recommender systems need user-specific data to generate accurate and meaningful recommendations. As mentioned before, this process is complicated compared to non-personalized systems, and hence we may face more problems. Such problems can be a lack of data about the user to generate recommendations (data sparsity), inability to generate recommendations for many users at a time (scalability issue), and other issues.

1.3 Types of Personalized Recommender Systems

- Collaborative filtering: These systems aggregate ratings of objects, identify similarities between the users based on their ratings, and generate new recommendations based on inter-user comparisons.

- Content-based: Here, the objects are mainly associated with each other through associated features. Unlike collaborative filtering, the system learns the user's interests based on his interaction with the system and not by relating his behaviour to other users.

- Demographic-based: It categorizes users based on demographic classes. It is comparatively easy to implement and does not require history or user ratings. Demographic systems use collaborative correlations between users but use different data.

- Hybrid: These combine any two of the systems mentioned above to suit some specific industry needs. As it can combine two models, it is the most sought-after in the industry.

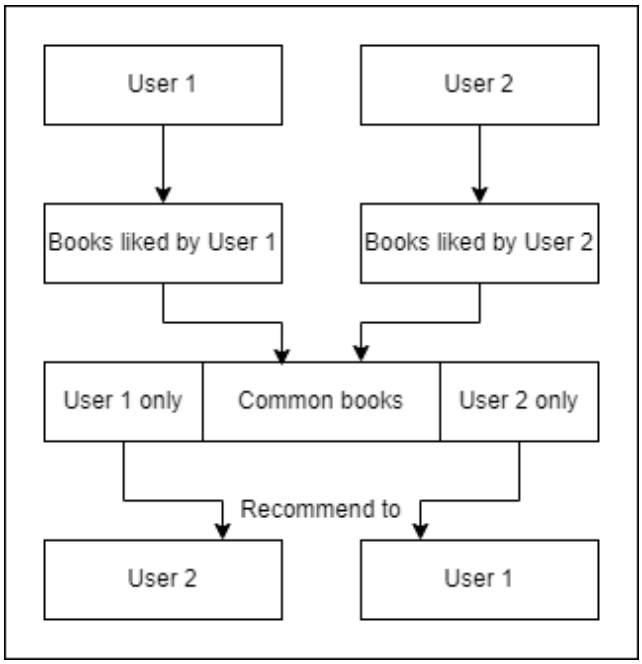

Fig. 1. Concept of Collaborative Filtering

This study focuses on the possible solutions to the cold-start issue in content-based recommender systems. Collaborative filtering recommender systems are the most widely used technologies in the market. They use the data of users of the system in correlation with each other. This interaction of multiple users' data to infer associations and recommendations assumes that the users who have agreed on something in the past will agree in the future and hence be interested in a similar type of object. This collaboration filtering among the user data is very interesting. Hence, this paper will be focusing on this aspect, along with some metadata, to address the issue of cold-start recommendations.

Memory-based approaches to collaborative filtering can be either user-item filtering or item-item filtering. As their name stand, user-item filtering focuses on a particular user and finds similar users based on item ratings. On the other hand, item-item filtering takes an item and, based on some users who liked the item, finds other users who also liked similar items.

On the other hand, model-based approaches are developed using machine learning algorithms. These approaches can be clustering-based, matrix factorization based or deep learning-based models.

The primary difference is that the memory-based approaches do not use parametric machine learning approaches. However, we use similarity metrics like cosine similarity or Pearson correlation coefficients, which are based on arithmetic operations.

Table 1. Types of Collaborative Filtering Approaches: Advantages and Disadvantages

Type of CF	Definition	Advantages	Disadvantages
Memory based CF	Find similar users based on cosine similarity or Pearson correlation and take the weighted average of ratings	Easy creation and explanability of results	Performance reduces when data is sparse, so non scalable
Model based CF	Use machine learning to find user ratings of unrated items e.g., PCA, SVD, Neural networks, matrix factorization	Dimensionality reduction deals with missing/sparse data	Inference is intracable because of hidden/latent factors

2 Literature Survey

The authors of [1] propose a text review-based collaborative filtering recommender system which extends a text embedding technique. This method is used to predict ratings and generate recommendations.

The authors of [2] propose constructing a hybrid model based on studying two probabilistic aspect models combined with user information such as age, gender, and job. This model posits that people with comparable characteristics (age, gender, and occupation) have similar interests. In the event of a large dataset, these three features are ineffective. Other options for improving performance can also be chosen.

The authors in [3] propose a Demographic collaborative recommender system which initially partitions the users based on demographic attributes and then clusters them based on ratings using a k-means clustering algorithm.

The authors in [4] try to base recommendations solely on users' demographic information by conducting k-means clustering experiments. The results did not exhibit any correlation between ratings and demographic features.

The authors in [5] demonstrate two recommender systems: one that uses projections of the bipartite user-item network to generate recommendations and compare the performance and a straightforward approach that uses a probabilistic model without graph structure.

The authors in [6] propose to eliminate fake reviews by performing sentiment analysis on them in order to avoid ambiguous recommendations.

The authors in [7] propose a probabilistic model to offer non-registered users a natural interface based on uncertainty rules. This natural interface allows the new users to infer their recommendations. The model automatically calculates the probabilities of non-registered users liking or disliking an item and the probability that the non-registered user either likes or dislikes similar items.

The authors of [8] present a solution to the cold start problem based on a probabilistic machine learning model that takes advantage of data obtained during acquisition. The model extracts information that can be used to forecast customer behavior in the future. It is called the 'First Impression Model' (FIM) by the authors, and it is based on the idea that the behaviors and choices of newly acquired customers might reveal underlying features that are predictive of their future behavior.

The authors in [9] propose a new recommendation model. This heterogeneous graph neural recommender learns user-to-item embedding using a convolutional graph network based on a heterogeneous graph constructed from user-item interactions, social links, semantic links predicted from the social network and text reviews.

2.1 Inferences

Currently, there are minimal applications of the same in a case where a new user registers with the system, and there is not enough data to generate recommendations.

What is intriguing is that despite the users generating new data constantly and the same being uploaded to the web, the cold-start issue is something that renders the recommender systems useless for new users. Moreover, since the user has never interacted with the system before, there was no data collected regarding user activity. Hence, the model has nothing to work on to generate recommendations for the same user.

This study aims to utilize the user-user collaborative filtering based on textual reviews and item-item graphs based on item similarity along with demographic data to improve the quality of recommendations and try to soothe the cold-start problem. A survey of the existing recommender systems was also done to understand the working of various product recommender systems.

2.2 Objectives

Product recommender systems: As mentioned above, product recommender systems play an essential role in the current era. Therefore, it is crucial to study and examine the working, architecture, and performance of different possible methodologies for generating user product recommendations.

Objective 1: To study and thoroughly understand the working of different product recommender systems.

Item-item recommender: The graph data structure allows us to represent and visualize metadata related to items in a single graph.\newline

Objective 2: Using metadata, generate an item-item graph with all items (including new items).

User-item recommender: Collaborative filtering systems utilize the sparse matrix embeddings of all user-item interactions in the system and use similarity measures to find the most similar users based on their behavior.

Objective 3: To apply a collaborative filtering method on textual reviews and star ratings given by users on the items.

Cold-start issue in Product Recommendation: Cold-start issue, called data-sparsity, is a scenario where the recommendation engine does not have enough data about the users and items in the dataset to generate accurate recommendations. This problem can be alleviated by including more data about users and items, like demographics, social networks, and other types of information.

Objective 4: To utilize demographic data of users to find similar users to new users in the system.

3 Proposed System

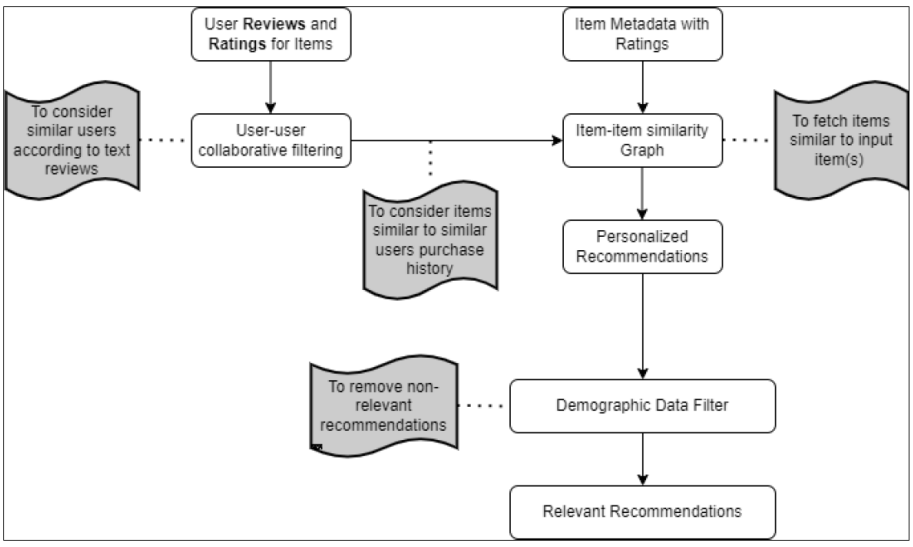

Fig. 2. Hybrid Architecture of Recommender System

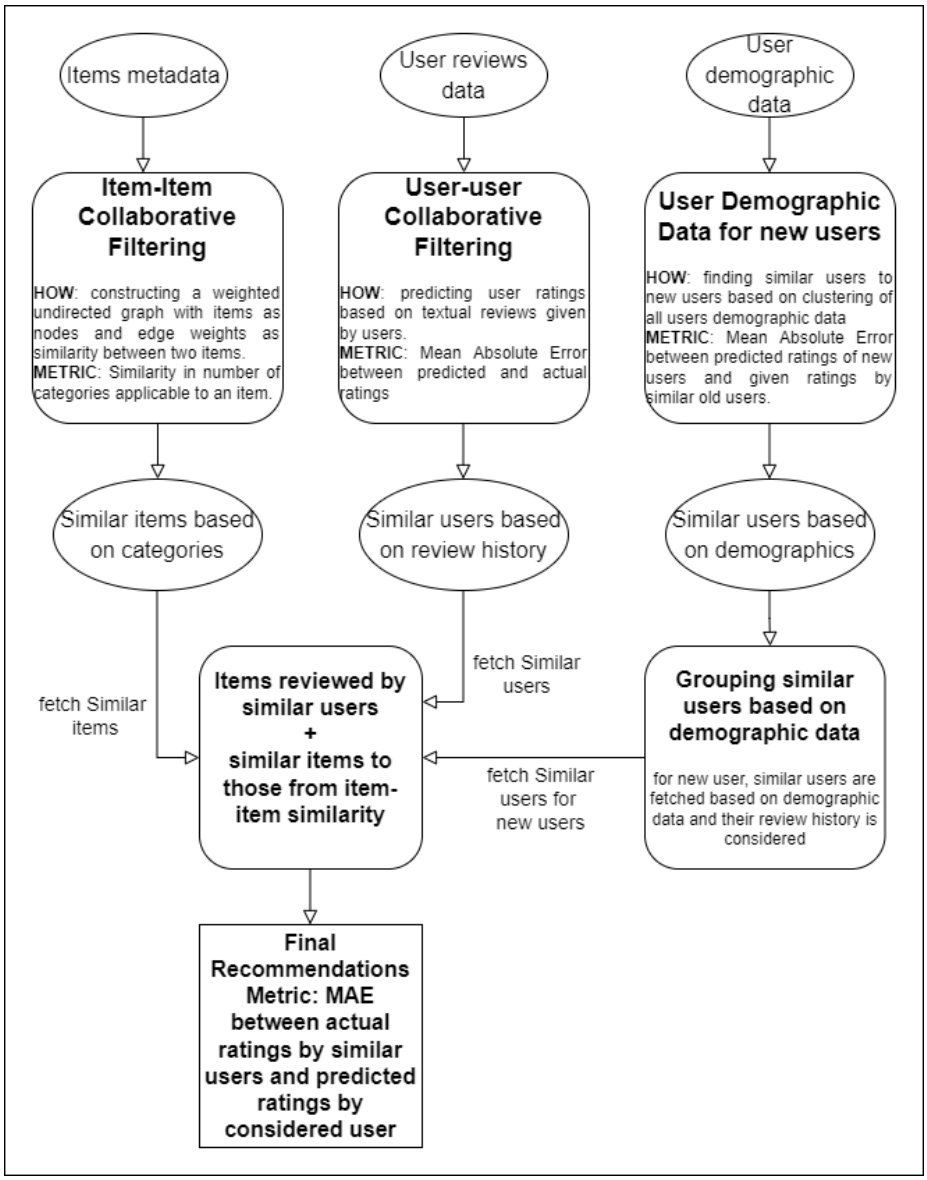

Fig. 3. Process of Recommendation using proposed Hybrid method

User-user Collaborative Filtering: User-User Collaborative filtering considers all users' purchase and rating history by constructing a sparse user-item matrix with values as ratings given by users to corresponding items. Each row as a 1-D vector depicts the items a particular user has rated. Top N similar users are fetched by calculating cosine similarity between the vectors. Using the similarity gives us the most similar users to a given user. In the case of new users, it groups them based on demographic data. So, whenever a user is given who does not have a rating history or has purchased and rated very few items, similar users to him will be found using clustering done based on the demographic data of all the users. Then the items in the purchase history of fetched similar users will be given as input to the item-item graph, and more items will be fetched to recommend to the given user. In this way, the recommendations given by the model will not be limited to the purchase history of other users. However, they will also recommend more similar items to those already purchased by a set of similar users.

Item-Item similarity: An item-item graph is constructed, and the edge weight between any two nodes is calculated based on the similarity between categories that apply to those two items. Then after setting a threshold value for neighboring items in the graph, similar items to one item are fetched from the same graph. Finds similar items using items metadata based on the Similarity formula given below:

$$Similarity = \frac{No.of\ words\ that\ are\ common\ between\ Categories}{Total\ no.of\ words\ in\ both\ Categories} \quad (1)$$

where, $0 \leq$ similarity ≤ 1,
such that: 0 is the least similar and 1 the most similar.

Following graph related measures were used:

- DegreeCentrality: this is the measure of centrality. As the graph is undirected, it is defined as the count of the number of neighbors a node has.
- ClusteringCoeff: by definition, this is a measure of the degree to which nodes in a graph tend to cluster together.

Recommendation Methodology

- For Registered users: similar users will be fetched by collaborative filtering using a sparse user-item rating matrix. The similarity of the two users is computed using the dot product of their rating matrices.

- For New users: similar users will be fetched using clustering of demographics data. Clustering will be done based on specific demographic features of users like – age and gender. After fetching similar users from a demographics perspective, top-rated products of those users will be recommended to the concerned user.

- Positively reviewed and rated items of filtered users (from collaborative filtering in case of old users and clustering in case of new users) will be fed into the item-item graph to get recommendations.
- For New Item: New items will already be included in the item-item graph considering their metadata.

Evaluation For recommended items, the familiar words between the set of categories for the input item and the recommended item are checked for accuracy. For each set of similar users, ratings are predicted for items they have not yet rated. Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are considered metrics.

$$RMSE = \sqrt{\frac{\sum(y_i - y_p)^2}{n}} \qquad (2) \qquad\qquad MAE = \frac{|(y_i - y_p)|}{n} \qquad (3)$$

where,
$y_i =$ actual value,
$y_p =$ predicted value and
$n =$ number of observations

Categories similarity calculation:

1. The purchased books list of a customer is shuffled and divided into two lists of equal length: purchased and validation set.
2. For each book in the purchased set, top-k books are recommended to the user. All these books form the list of recommended books.
3. The categories of all the books in recommended and validation sets are fetched from the dataset and appended together for all books to form two strings – one containing all the categories of books in the validation set and the other having all the categories of books in recommended set.
4. Using the previously mentioned formula, the similarity is calculated between these two strings.

4 Results and Analysis

4.1 Inferences

Item-item collaborative filtering is done by constructing a similarity graph which considers all the category labels that apply to each item in the dataset.
Based on the input item, the neighboring nodes in the graph that satisfy a predefined threshold similarity value are chosen, called the Degree-1 network of that item.

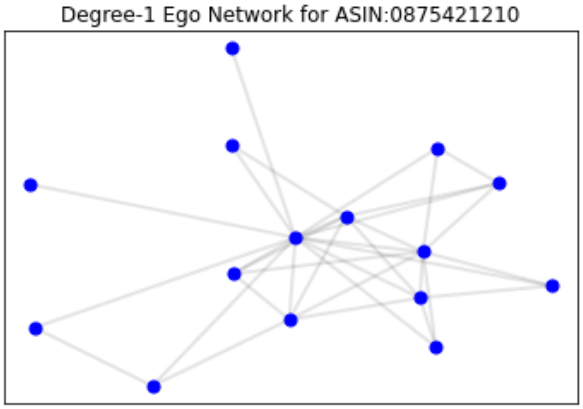

Fig. 4. Degree-1 Network of given product

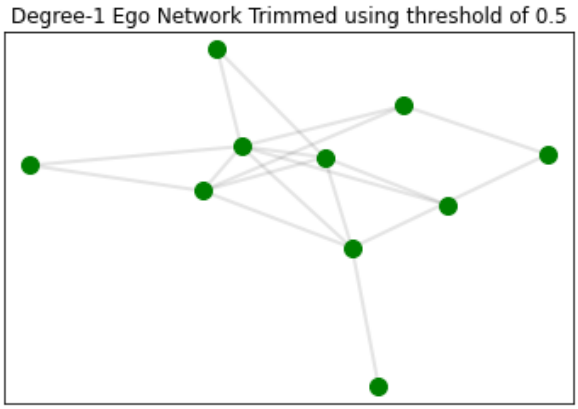

Fig. 5. Trimmed Graph with edge weight threshold of 0.5

4.2 User ratings prediction based on textual reviews

Rating is predicted based on sentiment analysis of textual reviews given by the users. A predefined python package 'TextBlob' is used to get the polarity of reviews, which utilizes rule-based Natural Language Processing methods to estimate the polarity of input text in the range of [-1,1], where -1 being the most negative and one being the most positive.

Then, based on the comparison of predicted sentiment and actual sentiment from the given ratings, accuracy is calculated:

```
reviews.head()
```

	ASIN	UserId	Rating	Reviews	polarity	ReviewSentiment	GivenSentiment
0	1882931173	AVCGYZL8FQQTD	4.0	julie strain fans collection photos pages wort...	0.133333	1.0	1.0
1	0826414346	A3VA4XFS5WNJO3	4.0	far aware first booklength study work dr seuss...	0.222832	1.0	1.0
2	0829814000	A3OQWLU31BU1Y	5.0	hadnt small church pastor long began hear davi...	0.152381	1.0	1.0
4	0253338352	AN9WUW5BG7M39	5.0	publisheraddresses interplay diverse spiritual...	0.110065	1.0	1.0
5	0802841899	A2H2LORTA5EZY2	4.0	useful thorough text book would recommend anyo...	0.250000	1.0	1.0

```
total = len(reviews)
correct = reviews[reviews['ReviewSentiment'] == reviews['GivenSentiment']]
correct_prediction = len(correct)

print("Accuracy of Review Sentiment: ", (correct_prediction/total)*100)
```

```
Accuracy of Review Sentiment:  89.94597676396633
```

Fig. 6. Accuracy of Review sentiment

4.3 User ratings prediction based on textual reviews

Using the user-item rating sparse matrix mentioned above, similar users are fetched, and ratings are predicted for the items the user has not yet rated. Then RMSE and MAE are calculated based on the predicted ratings and actual ratings the user gives.

```
# RMSE Score

diff_sqr_matrix = (test - pred)**2
sum_of_squares_err = diff_sqr_matrix.sum().sum()

rmse = np.sqrt(sum_of_squares_err/total_non_nan)
print(rmse)
```

```
1.8779388709758476
```

```
# Mean abslute error

mae = np.abs(pred - test).sum().sum()/total_non_nan
print(mae)
```

```
1.299362163268938
```

Fig. 7. RMSE and MAE of predicted Ratings

4.4 Final Recommendations

Similar users are fetched based on the input users ID in the final step of generating recommendations, and ratings are predicted for non-rated items. Then the items with the highest predicted rating for that particular user are given input into the item-item

similarity graph to fetch recommendations for each item that the user is likely to rate high.

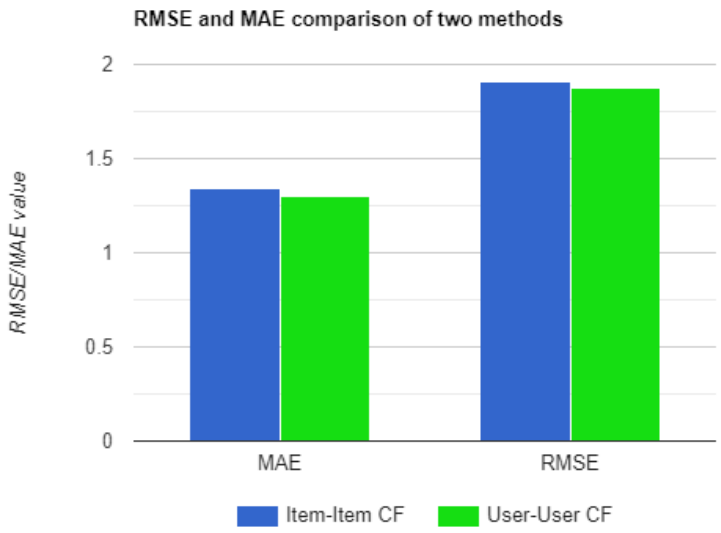

Fig. 8. Comparison of proposed (user-based CF based on textual reviews) and existing recommendation method (item based CF) [20],[15]: Comparing the MAE and RMSE values on Amazon Books dataset using Item-item CF and user-user CF. It is seen that there is a slight reduction in both metrics when we try to predict ratings of items using user-based CF based on user-item sparse matrix.

Conclusion

The growing amount of user and item data is being used to improve the accuracy and efficiency of product recommender systems, which has improved the domain's appeal. We have attempted to use the depth of data in terms of features and size, trying to provide reliable recommendations for new users and items. This study examines several approaches, tactics, and recommendation algorithms in-depth and suggests a hybrid architecture to address the cold-start problem, which uses an item-item similarity graph and user-user collaborative filtering based on textual reviews (For new users, we consider their demographic data). In turn, the combination of products and user demographics data can be used to address data sparsity, resulting in recommendations being generated even for cold-start users and products. To further try and improve the quality of recommendations, one can consider more features of users' data to group and find users of similar interests. Furthermore, the same model can be scaled to construct a graph of a larger dataset of items to include items across differ-

ent categories, and users can also be categorized according to different demographic data like location and age group to improve the accuracy of recommendations across multiple users and multiple categories of products. We sincerely hope that the research and analysis presented in this work will aid other researchers in better understanding and exploring the applications of various models to improve the overall quality of recommender engines.

```
# Print Top 5 Recommendations
print('\nTop 5 Recommendations by AvgRating then by TotalReviews for Users Purchased the book:')
print('\n-------------------------------------------------------------------------------------')
print('ASIN\t', 'Title\t', 'SalesRank\t', 'TotalReviews\t', 'AvgRating\t', 'DegreeCentrality\t', 'ClusteringCoeff')
for asin in T5_byAvgRating_then_byTotalReviews:
    print(asin)
```

```
Top 5 Recommendations by AvgRating then by TotalReviews for Users Purchased the book:

-------------------------------------------------------------------------------------
ASIN     Title   SalesRank       TotalReviews    AvgRating       DegreeCentrality        ClusteringCoeff
('0875421229', "Cunningham's Encyclopedia of Magical Herbs (Llewellyn's Sourcebook Series)", 6565, 96, 4.5, 68, 0.65)
('0875421849', "Living Wicca: A Further Guide for the Solitary Practitioner (Llewellyn's Practical Magick)", 6003, 95, 4.5, 30,
0.81)
('0875421318', "Earth, Air, Fire, and Water: More Techniques of Natural Magic (Llewellyn's Practical Magick Series)", 7286, 57,
4.5, 10, 0.73)
('0875421261', "Cunningham's Encyclopedia of Crystal, Gem, and Metal Magic", 14867, 39, 4.0, 17, 0.54)
('0875421245', "The Magical Household: Spells & Rituals for the Home (Llewellyn's Practical Magick Series)", 111836, 21, 4.0,
8, 0.7)
```

```
print(len(recommendedCategories), len(purchasedCategories))
```

```
4155 3858
```

```
recommendedCategories = set(recommendedCategories)
purchasedCategories = set(purchasedCategories)
```

```
intersect = recommendedCategories & purchasedCategories
union = recommendedCategories | purchasedCategories
```

```
if(len(union)) > 0:
    similarity = round(len(intersect)/len(union), 2)
```

```
print("Similarity between the categories set of recommended books and validation set in purchased books is:", similarity)
```

```
Similarity between the categories set of recommended books and validation set in purchased books is: 0.96
```

Fig. 9. Final generated Recommendations and similarity measure in purchased and recommended items

References

1. Srifi, M., Oussous, A., Ait Lahcen, A., & Mouline, S. (2020). Recommender systems based on collaborative filtering using review texts—A survey. Information, 11(6), 317.
2. Pan, R., Ge, C., Zhang, L., Zhao, W., & Shao, X. (2020). A new similarity model based on collaborative filtering for new user cold start recommendation. IEICE TRANSACTIONS on Information and Systems, 103(6), 1388-1394.
3. Zhang, Z., Zhang, Y., & Ren, Y. (2020). Employing neighborhood reduction for alleviating sparsity and cold start problems in user-based collaborative filtering. Information Retrieval Journal, 23(4), 449-472.
4. Natarajan, S., Vairavasundaram, S., Natarajan, S., & Gandomi, A. H. (2020). Resolving data sparsity and cold start problem in collaborative filtering recommender system using linked open data. Expert Systems with Applications, 149, 113248.
5. Liu, S., Ounis, I., Macdonald, C., & Meng, Z. (2020, July). A heterogeneous graph neural model for cold-start recommendation. In Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval (pp. 2029-2032). Study of Cold-start Product Recommendations and its Solutions 13
6. Kapoor, N., Vishal, S., & Krishnaveni, K. S. (2020, June). Movie recommendation system using nlp tools. In 2020 5th International Conference on Communication and Electronics Systems (ICCES) (pp. 883-888). IEEE.
7. Hernando, A., Bobadilla, J., Ortega, F., & Guti´errez, A. (2017). A probabilistic model for recommending to new cold-start non-registered users. Information Sciences, 376, 216-232.
8. Leskovec, J., Adamic, L. A., & Huberman, B. A. (2007). The dynamics of viral marketing. ACM Transactions on the Web (TWEB), 1(1), 5-es.
9. McAuley, J., & Leskovec, J. (2013, October). Hidden factors and hidden topics: understanding rating dimensions with review text. In Proceedings of the 7th ACM conference on Recommender systems (pp. 165-172).
10. Valdiviezo-D´ıaz, P., & Bobadilla, J. (2018, August). A hybrid approach of recommendation via extended matrix based on collaborative filtering with demographics information. In International Conference on Technology Trends (pp. 384-398). Springer, Cham.
11. Qian, T., Liang, Y., & Li, Q. (2019). Solving cold start problem in recommendation with attribute graph neural networks. arXiv preprint arXiv:1912.12398.
12. Lam, X. N., Vu, T., Le, T. D., & Duong, A. D. (2008, January). Addressing coldstart problem in recommendation systems. In Proceedings of the 2nd international conference on Ubiquitous information management and communication (pp. 208-211).
13. Togashi, R., Otani, M., & Satoh, S. I. (2021, March). Alleviating cold-start problems in recommendation through pseudo-labelling over knowledge graph. In Proceedings of the 14th ACM International Conference on Web Search and Data Mining (pp. 931-939).
14. Chicaiza, J., & Valdiviezo-Diaz, P. (2021). A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions. Information, 12(6), 232.
15. Ricci, F., Rokach, L., Shapira, B., Kantor, P.: Recommender systems handbook. 1. Springer, NewYork (2011).
16. Devika, P., Jisha, R. C., & Sajeev, G. P. (2016, December). A novel approach for book recommendation systems. In 2016 IEEE international conference on computational intelligence and computing research (ICCIC) (pp. 1-6). IEEE.
17. Kavinkumar, V., Reddy, R. R., Balasubramanian, R., Sridhar, M., Sridharan, K., & Venkataraman, D. (2015, August). A hybrid approach for recommendation system with added

feedback component. In 2015 International Conference on Advances in Computing, Communications, and Informatics (ICACCI) (pp. 745-752). IEEE.
18. Bindu, K. R., Visweswaran, R. L., Sachin, P. C., Solai, K. D., & Gunasekaran, S. (2017). Reducing the cold-user and cold-item problem in recommender system by reducing the sparsity of the sparse matrix and addressing the diversity-accuracy problem. In Proceedings of International Conference on Communication and Networks (pp. 561-570). Springer, Singapore.
19. Tan, Y., Zhang, M., Liu, Y., & Ma, S. (2016, July). Rating-boosted latent topics: Understanding users and items with ratings and reviews. In IJCAI (Vol. 16, pp. 2640-2646).
20. Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. Knowledge-based systems, 46, 109-132.
21. Sharma, J., Sharma, K., Garg, K., & Sharma, A. K. (2021). Product Recommendation System a Comprehensive Review. In IOP Conference Series: Materials Science and Engineering (Vol. 1022, No. 1, p. 012021). IOP Publishing.