Final report

Analysis of Algorithm Performance at the User Level

Afthab Shibin Kootteeri

University of Siegen

Intelligent Systems Group (ISG)

Dept. of Electrical Engineering and Computer Science

December 20, 2024

Confirmation

I hereby confirm that this report is entirely my own work and that I have not used any additional assistance or resources other than indicated. All quotations, paraphrases, information and ideas that have been taken from other sources (including the Internet as well as other electronic sources) and other persons' work have been cited appropriately and provided with the corresponding bibliographical references. The same is true of all drawings, sketches, pictures and other illustrations that appear in the text. I am aware that the neglect to indicate the used sources is considered as fraud and plagiarism in which case sanctions are imposed that can lead to the suspension or permanent expulsion of students in serious cases.

Siegen, 20/12/2024	[illegible]
Place, Date	Signature

Abstract

Recommender systems are now integral to various domains including e-commerce and streaming platforms. These recommendations are essential for increasing user experiences by predicting user preferences. The interactions of these platforms increase based on how good recommendations the users get. Most of the recommender systems use the same recommendation algorithm for all users. This research investigates whether selecting algorithms based on users improves recommendation quality.

We have used the Recpack library for applying item similarity algorithms like ItemKNN, SLIM, NMFItemToItem, SVDItemToItem and hybrid similarity algorithm KUNN and factorization algorithms NMF and SVD on MovieLens100K, MovieLens1M, MovieLens10M, CiteULike, Globo datasets. Normalized Discounted Cumulative Gain (NDCG) was used as the performance metric. NDCG value considering the top 10 best-scoring item predictions was used for the analysis. The overall NDCG value and the per-user NDCG values are calculated for every algorithm-dataset pair for evaluating the recommendation quality, we found that the per-user NDCG value differs largely compared to the overall NDCG value of the algorithm.

The per-user NDCG score distributions varied significantly across algorithms in the same dataset. However, when comparing overall NDCG values, ItemKNN and SLIM have better values than NMF and SVD. But when we compare per user NDCG value, some algorithm performed poorly for more users compared to other algorithm. For instance, NMFItemToItem algorithm has more than 0.7 proportion of users with NDCG values equal to zero and SVD has only 0.52 proportion of users with zero NDCG values in the Globo dataset. The overall NDCG values for NMFItemToItem and SVD in the same dataset are 0.108 and 0.2 respectively. Even though there aren't much difference in overall NDCG scores, the proportion of users got zero NDCG was higher in NMFItemToItem dataset.

The users get recommendations differently with different algorithms. In this project, we analyse whether selecting algorithms based on user is better than using the same recommendation algorithms for all users. The evaluation method used in this project has provided valuable insights into a way of thinking that facilitate more personalized and accurate recommendations.

Contents

1 Introduction

Recommendation systems are widely employed in various modern-day domains to enhance user experiences by providing personalized content such as music[1], news, jobs, products, movies[2] and books [3]. The recommender system learns from user preferences, interactions, and past behaviours to provide users with personalized recommendations. It relies on various algorithms and models to provide accurate recommendations for users. To ensure that recommendations meet user needs and expectations, recommender systems are usually evaluated.

Common evaluation strategies include online evaluations and offline evaluations. In offline evaluation, metrics are calculated for each algorithm. Key metrics include precision, recall, normalized discounted cumulative gain NDCG, root mean square error, and mean absolute error. Typically, these metric values will be a single number such as NDCG $= 0.315$ for an algorithm that recommends relevant items for all the users. The algorithm performance of recommending relevant items for a large set of users is evaluated using this metric value.

The problem is that, the practice of evaluating each algorithm at the user level and dynamically selecting the best-performing algorithm for each user based on the individual user level evaluation is not well understood or systematically studied. Currently, recommendation systems often rely on fixed combinations of algorithms, which may fail to serve all users optimally. The overlook of algorithm performance per user may result in a disproportionate impact on minority user groups. Optimizing entirely for broad engagement can cause a lack of relevant recommendations for users whose preferences are not suited to the fixed combination of algorithms.For instance, even though a single recommendation algorithm can perform exceptionally well for

60% of users, it may be delivering subpar recommendations for the remaining users. So, evaluating algorithm performance per user may impact the recommendation quality positively and it may lead majority of users achieving relevant recommendations.

However, open questions are, how can we design recommender systems to maximize the proportion of users receiving high-quality recommendations while minimizing the number of users with bad recommendations? How can the recommendation algorithm performance per user be measured and compared effectively to check the algorithm performance per user? If we are using overall algorithm performance evaluation instead of evaluating algorithm performance per user, how does it impact the satisfaction of all users? These questions aim to explore the variation of the algorithm's performance at a user level.

Our goal with this project is to systematically evaluate the performance of recommendation algorithms at the individual user level. We aim to understand how much variability in recommendation each user endures when receiving these recommendations from different algorithms. The objective is to check whether a single algorithm performs ideally for all users, or if there is any algorithm that works reasonably well for every user instead of giving bad recommendations for some.

We hypothesize that recommender-systems research would benefit from evaluating an algorithm's performance at a user level. This approach, rather than relying solely on an overall performance metric, can better identify algorithms that deliver more relevant recommendations to the majority of users. We develop a framework to assess the recommendation quality per user by using per-user NDCG scores of six different algorithms across five datasets. A subsequent idea is to quantify the proportion of users achieving good per-user metric values, the proportion achieving bad per-user metric values, and the proportion of users whose metric values are close to the overall performance metric of the algorithm.

2 Related Work

A wide variety of recommendation algorithms and evaluation methods have been developed and refined and are being used in different areas over the years. Different collaborative filtering algorithms including item similarity algorithms, hybrid similarity algorithms, and factorization algorithms have been evaluated traditionally at the global level using various datasets. Item similarity algorithms such as ItemKNN [4] , SLIM [5] , NMFItemToItem, and SVDItemToItem make use of the existing relationships between items to provide recommendations. Hybrid similarity algorithm KUNN [6] uses not only user similarity but also item similarity in combination with it to make recommendations. NMF and SVD are factorization algorithms [7] which are model-based methods that infer latent factors to capture user and item characteristics.

The importance of ranking-based evaluation metrics has been the topic of the research field for quite a while and Normalised Discounted Cumulative Gain (NDCG) is a key metric for top-n recommendation. The studies conducted in [8] and [9] provides a detailed analysis of the NDCG metric. NDCG metric places a high value on placing relevant items at the top of the recommendation list. It incorporates not only the relevance of an item but also its position in the ranked list.

An algorithm performance evaluation normally involves obtaining a single metric value for various evaluation metrics on different datasets. A study [10] compared evaluation metrics including NDCG metric for various libraries. The study involved calculating various metric values for different libraries, but the algorithm's performance was evaluated across the entire dataset. Another study involving a comparison of common recommendation algorithms implemented using three well-known recommendation frameworks [11] also exists.

In this work, we focus on evaluating algorithm recommendation performance on the user level to give rise to more dominant recommendations for users. We believe that no one has evaluated algorithm performance per user and a development in this area may benefit the research community. We aim to discover whether finding algorithm performance at the user level can offer insights into tailing algorithmic choices to individual users.

3 Methodology

Our setup consists of a structured pipeline built using the Recpack library to gather necessary data and facilitate the evaluation of the recommendation algorithm at an individual user level and across the entire dataset. We evaluate the performance of different algorithms across multiple datasets using this pipeline. The primary evaluation metric selected was the Normalized Discounted Cumulative Gain (NDCG) for its ability to capture both positional significance and ranking relevance.

3.1 RecPack Library

Recpack is a Python library which serves as an experimentation toolkit for top-N recommendations using implicit feedback data [12]. It offers the necessary tools to conduct reproducible and reusable experiments. It comes with a range of different metrics, datasets, recommendation scenarios, and algorithms. We used the Recpack library's feature to integrate multiple datasets and implement various algorithms to meet the requirements of our research.

Recpack provides a collection of classes designed to manage some of the most used datasets. These different classes can be used to download and store necessary datasets. Recpack datasets have default preprocessing steps to transform them into a user-item interaction matrix. This matrix in Recpack is represented by an interaction matrix object optionally with timestamps

Recpack offers a set of recommendation scenarios for us to choose from, allowing users to evaluate a recommendation algorithm while avoiding data leakage. More than twenty state-of-the-art recommendation algorithms are included in this library, providing a diverse selection for experimentation.

For this project, we selected the following algorithms:

- **Item Similarity Algorithms**: SLIM, ItemKNN, NMFItemToItem, and SVDItemToItem.

- **Hybrid Similarity Algorithms**: KUNN.

- **Factorization Algorithms**: NMF and SVD.

The Recpack library includes the most commonly used metrics in the Top-N ranking. We used the NDCG@10 metric to evaluate the recommendation performance of algorithms across entire datasets. Additionally, to evaluate each algorithm performance per-user basis, we calculated the NDCG@10 of individual users. Recpack allowed us to do this fine-grained analysis, facilitating a detailed evaluation of each algorithm's performance. A diagram illustrating a typical pipeline of the Recpack library is shown in the Figure 1.

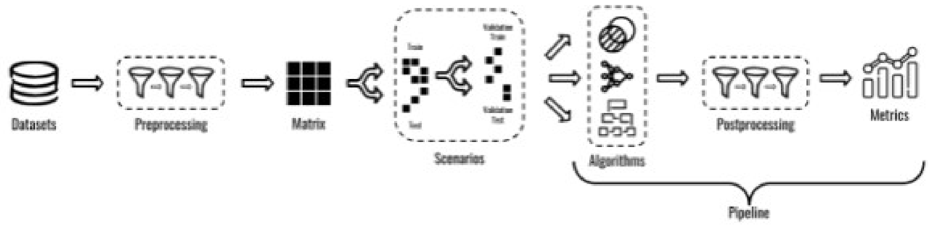

Figure 1: Pipeline for Top-N Recommendation Evaluation

3.2 Datasets

The datasets for this study were selected from the datasets included in the Recpack library. The Recpack library provides access to various datasets. By simply instantiating a Dataset object we could easily use any datasets present in the Recpack library. We selected datasets based on the computational requirements, excluding those with long processing times. The following datasets were chosen:

- **MovieLens Data**: MovieLens100K, MovieLens1M, and MovieLens10M datasets including movie ratings [13].
- **Globo Dataset**: large dataset collected from a news portal, including user interaction logs, specifically tracking page views [14].
- **CiteULike**: This dataset includes user-created collections of articles [15].

Table 1 presents the number of users across each of the datasets.

Dataset	**Users**
MovieLens100K	942
MovieLens1M	6038
MovieLens10M	69797
Globo	218223
CiteULike	5550

Table 1: Datasets and Number of Users

3.3 Metric

Normalized Discounted Cumulative Gain (NDCG) metric used for evaluation of the selected recommendation algorithms. The reason behind the selection of this metric system is that it considers how well the ranking of items corresponds to their relevance and positions in the result set. The NDCG is calculated in two key steps:

1. Discounted Cumulative Gain (DCG) is calculated using the relevance scores of the items. The formula for DCG at the rank p is given as in the equation 1

$$\text{DiscountedCumulativeGain}(u) = \sum_{i \in \textit{Top-K}(u)} \frac{y_{u,i}^{\text{true}}}{\log_2(\text{rank}(u,i)+1)}$$

   Where rel_i is the relevance of an item at position i.

2. Normalization of the DCG by the Ideal Discounted Cumulative Gain (IDCG) represents the best possible DCG that can be achieved for a perfect ranking.

$$\text{NormalizedDiscountedCumulativeGain}(u) = \frac{\text{DCG}(u)}{\text{IDCG}(u)}$$

IDCG computed as:

$$\text{IDCG}(u) = \sum_{j=1}^{\min(K, |\mathcal{Y}_u^{\text{true}}|)} \frac{1}{\log_2(j+1)}$$

In the Recpack library, the NDCG metric class has parameter K, which is the recommendation list size of the Top-K item predictions. We choose the parameter K values as 3, 5, 10, and 20 to find the overall NDCG values. The NDCG values are always in the interval [0,1]. A score of one indicates that the recommendation algorithm successfully delivered items of interest to the user and a score of zero suggests that the recommendation algorithm failed for that user.

3.4 Experiment Setup

3.4.1 Preprocessing Data

The required datasets are loaded and preprocessed using the set of classes that `Recpack` offers. This class implements the standard preprocessing steps. The preprocessing step generates a consecutive index for both the user and items. This is important because it is common to represent the user-item interaction data in a matrix form. For instance, in the case of MovieLens100K, the rating was binarized by retaining only interactions with a rating of 4 or higher. To ensure sufficient interaction density, users with fewer than 3 interactions and items with fewer than 5 interactions are removed.

3.4.2 Experimental Scenarios

We need to split the data into training, validation, and test data. This splitting of the data ensures robust evaluation and avoids possible data leakage. Using recpack. scenarios module, we applied the WeakGeneralisation scenario, which is a frequently used scenario.Figure 2 illustrate the data splitting process using the WeakGeneralisation scenario. We split 75% of each user's interactions into training data, while the remaining 25% are used as test target data. For the algorithm's parameter optimisation, the full training data is further split into two subsets: 75 % for validation training data and 25% for validation target data.

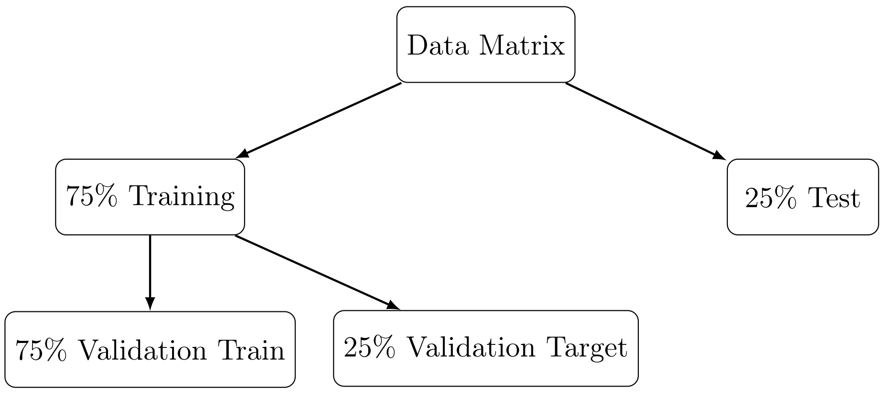

Figure 2: Data Splitting using WeakGeneralisation scenario

3.4.3 Pipeline Construction

We constructed a pipeline for optimizing the hyperparameters of the selected algorithms and obtaining the performance metrics. This pipeline integrated the preprocessed and split data, algorithms, hyper-parameter grids, and performance metrics. For instance, the KUNN algorithm is optimized using parameters Ku, and Ki. Ku is the number of neighbours to keep in the user similarity matrix and Ku is the number of items to keep as neighbours in the item similarity matrix. The NDCG value is calculated at the user level and across the dataset for evaluating the algorithm performance for recommending relevant items.

3.4.4 Resource Constraints

The selection of datasets and algorithms was based on the available computational resources. The algorithms are selected based on their runtime and memory requirements. We tried all the available Item Similarity, Hybrid Similarity, and Factorization

Algorithms in the Recpack library. However, any algorithm that takes more than one day for its evaluation, was excluded from consideration. Similarly, datasets were also selected based on these constraints, ensuring that the chosen algorithms should be completed within one day and the memory requirement should not exceed the maximal available memory of our system.

3.4.5 Algorithms and Hyperparameters

For all selected algorithms, the default values, along with two or three values close to the default value, are used. These different hyperparameters and their values are specified in the grid to try every combination of hyper-parameter values. Table 2 provides an overview of the different algorithms and their corresponding hyperparameters, including the values we tried.

Table 2: Hyperparameters and Values for Algorithms

Algorithm	**Hyperparameter**	**Values**
ItemKNN	K	100, 200, 500
	similarity	'cosine', 'conditional_probability'
	pop_discount	None, 0.5, 0.7
	normalize_X, normalize_sim	True, False
SLIM	l1_reg	0.0003, 0.0005, 0.0007
	l2_reg	0.00003, 0.00005, 0.00007
	fit_intercept	True
	Ignore_neg_weights	True
NMFItemToItem	num_components	10, 50, 100, 200
SVDItemToItem	num_components	10, 50, 100, 200
NMF	num_components	50, 100, 200
	alpha	0.0001, 0.001, 0.1
	l1_ratio	0, 0.5, 1
SVD	num_components	50, 100, 200
KUNN	Ku	50, 100, 200
	Ki	50, 100, 200

4 Results

The per-user NDCG values of all selected algorithms and dataset pairs are examined. Along with that, the overall NDCG of each algorithm on different datasets is analyzed. We selected the NDCG metric parameter value k as 10 to analyze our results.

4.1 Overall NDCG Scores

The overall NDCG for the selected algorithms on selected datasets is in the range of 0.107 to 0.352. Table 3 summarizes the overall NDCG@10 received by each algorithm across five datasets. This table provides a summary of each algorithm's performance and the variability of NDCG across datasets.

Algorithm	**CiteULike**	**Globo**	**MovieLens100K**	**MovieLens10M**	**MovieLens1M**
ItemKNN	0.233	0.197	0.317	0.293	0.319
SLIM	0.172	0.200	0.323	0.352	0.346
NMFItemToItem	0.107	0.108	0.254	0.191	0.197
SVDItemToItem	0.110	0.134	0.233	0.208	0.220
NMF	0.118	0.166	0.253	0.227	0.281
SVD	0.136	0.168	0.314	0.299	0.308
KUNN	0.250	0.189	0.336	0.303	0.345

Table 3: Comparison of Overall NDCG Scores for Algorithms Across Multiple Datasets

The highest NDCG value (0.352) was achieved by the SLIM algorithm on the MovieLens10M dataset, indicating the algorithm's strong performance in a large structured dataset. The NMFItemToItem algorithm on the CiteULike dataset achieved the lowest NDCG value. This shows the challenges faced by certain algorithms in a sparse data setting. When the dataset size increases, the SLIM algorithm consistently outperforms other algorithms. The NMFItemToItem algorithm consistently produces the lowest NDCG value across all five datasets. The consistency of different algorithms across datasets is illustrated in Figure 3.

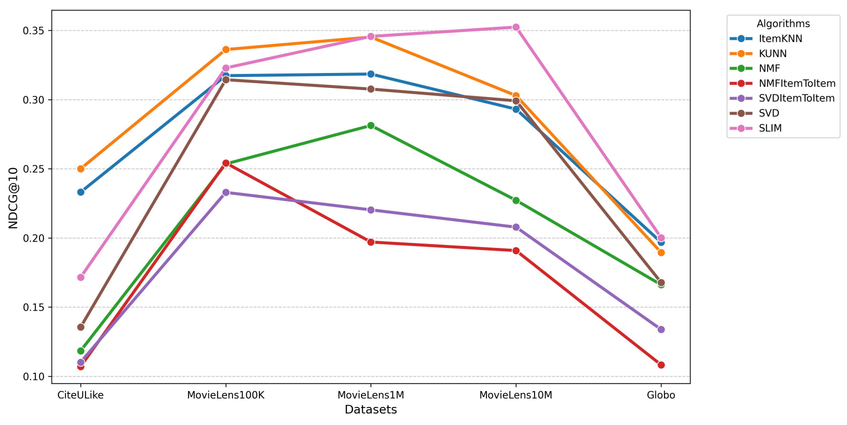

Figure 3: Performance Consistency of Algorithms Across Datasets

4.2 User Level Ndcg Scores

Figure 4 presents a violin plot for an overview of the distribution of NDCG scores of users across algorithms and datasets. It also consists of the overall NDCG scores of each algorithm in datasets to understand how different each algorithm's performance is on the user level compared to its overall performance. The user NDCG values are widely distributed between zero and one value in almost all algorithm dataset pairs. KUNN, SVD, and SLIM are the algorithms with higher median NDCG scores, particularly in MovieLens 100K and 1M datasets. Algorithms like KUNN, ItemKNN, SVD, and SLIM display a wider distribution of NDCG which indicates that these algorithms perform inconsistently across users. No single algorithm generated an NDCG score of more than 0.8 for even 25% of users, highlighting the difficulty in achieving highly relevant recommendations for users.

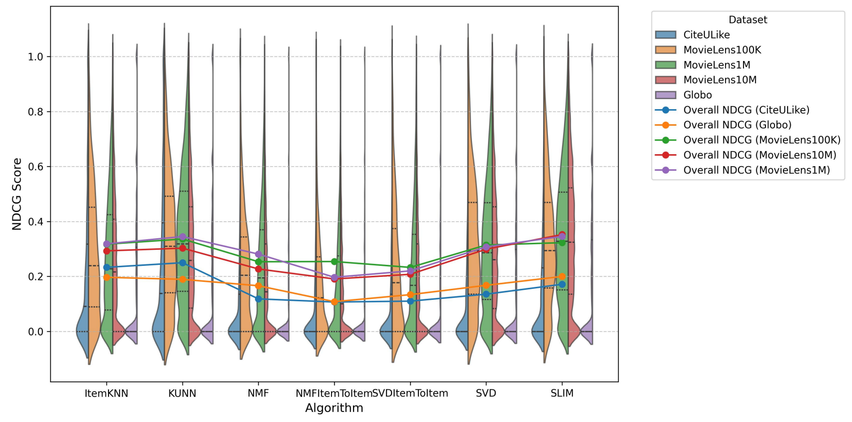

Figure 4: Per-User Ndcg Scores Across Algorithms and Datasets

The proportion of users for whom the algorithms performed poorly to provide relevant recommendations is shown in Figure 5. NMF and NMFItemToItem demonstrate relatively higher failure rates across users with the proportion of users with an NDCG $= 0$ for 0.729 and 0.733 respectively on the Globo dataset. Algorithms SLIM, KUNN, and SVD performed well for users in MovieLens datasets,which is shown by their low proportion of users with zero NDCG scores. The lowest was by SLIM on the MovieLens1M dataset with 0.119. The effect of the dataset is evidently visible. For instance, in the CiteULike dataset, most algorithms struggled, with the SLIM algorithm failing to recommend relevant items for nearly 51% of users.

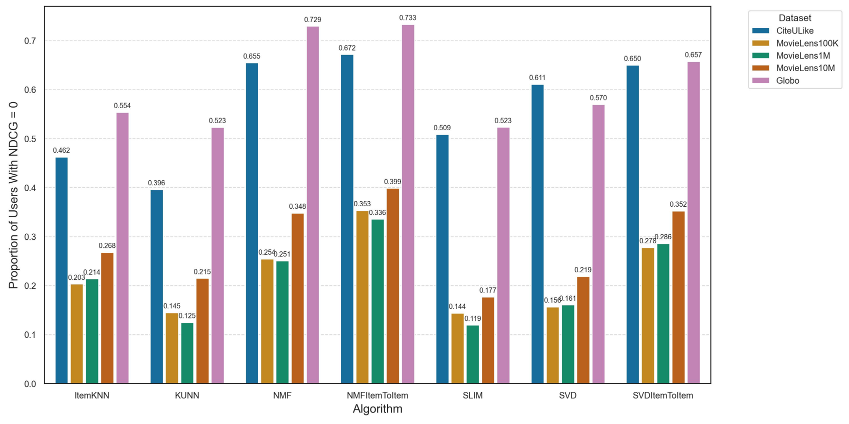

Figure 5: Proportion of Users With an Ndcg Score of 0

Across all algorithms and datasets, the proportion of users achieving an NDCG score of one is extremely low, which is shown in the Figure 6. When compared to other algorithms, KUNN is the only algorithm with a proportion of users exceeding 0.03 in two datasets. To provide more clarity on how different the user NDCG scores are compared to the overall NDCG score, we can make use of Figure 7. This figure explains the proportion of users whose NDCG scores are close to the overall NDCG score of each algorithm. Only a maximum of 33% of users got an NDCG value which is close to the overall NDCG score of an algorithm from all these algorithm-dataset pairs. In all cases, this proportion ranged between 0.1 to 0.35.

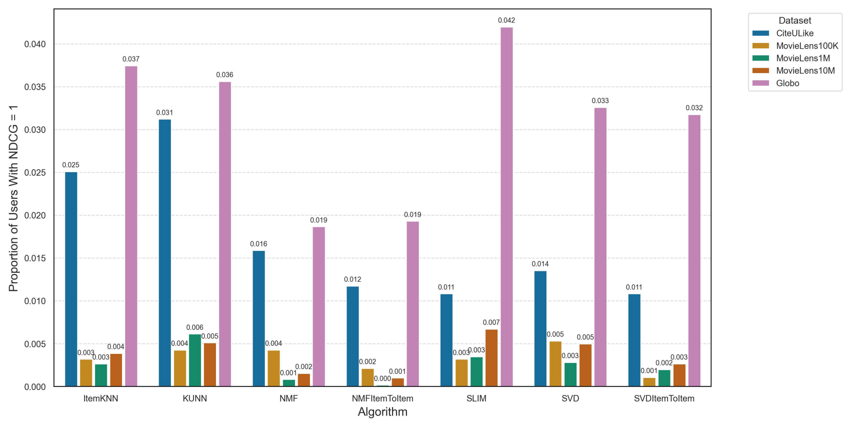

Figure 6: Proportion of Users With an NDCG Score of 1

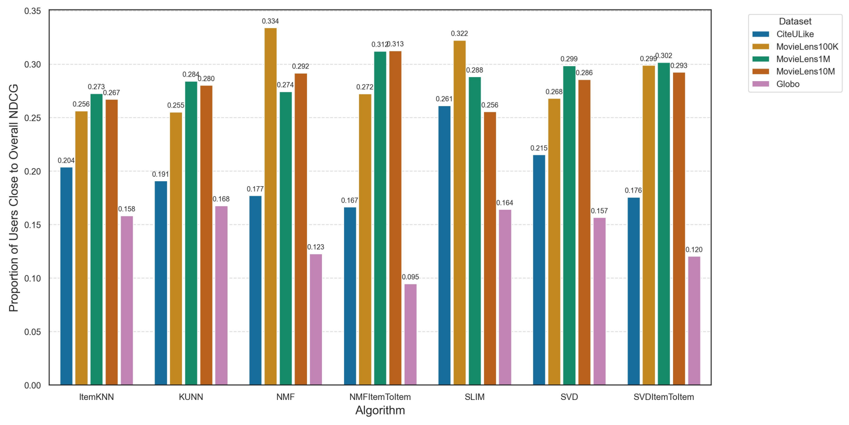

Figure 7: User Proportion Close to Overall Algorithm NDCG

4.3 Analysis of Users With NDCG of Zero Across Algorithms

The distribution of users who scored zero NDCG across multiple recommendation algorithms is shown in the Table 4 and Figure 8.The table shows the number of users who scored an NDCG of zero across algorithms. It consists of users who scored zero in a single algorithm as well as those users with an NDCG score of zero across all algorithms. The pie chart indicates the aggregate distribution of all users across all of the algorithm-dataset pairs who scored an NDCG zero in one multiple or all algorithms.

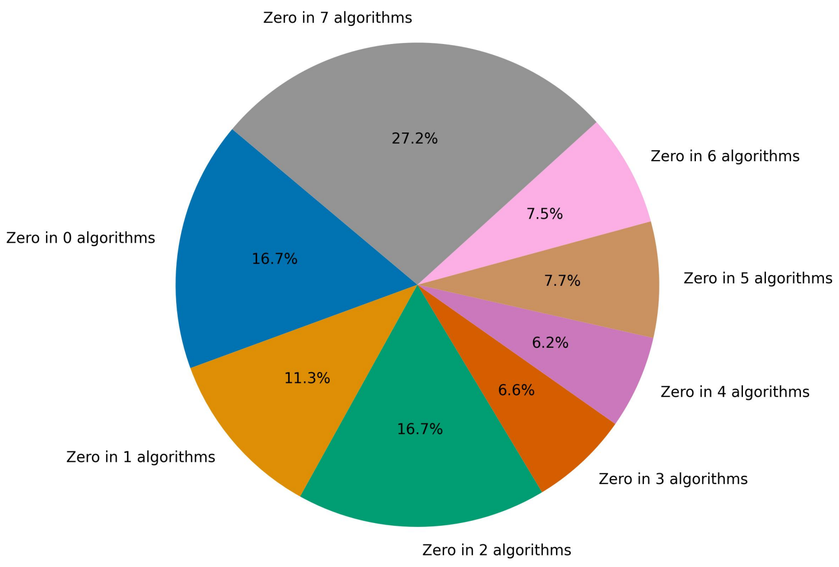

Figure 8: Aggregated Distribution of Users Scoring NDCG Zero Across Algorithms

Even though some algorithms may fail to recommend relevant items to a user, others can still perform considerably well. But for some users, none of the algorithms may recommend well or only a single algorithm might work. 27.2% of users scored zero in all 7 algorithms, indicating the lack of relevant recommendations for them. While 16.7% of users did not score zero in any of the algorithms. There exist some users to whom only one algorithm worked well to recommend relevant items, it was just 7.5% of the total users. Approximately 30.7% of the total users present in the Globo dataset scored an NDCG score of zero across all algorithms.

Number of Algorithms	**CiteULike**	**MovieLens100K**	**MovieLens1M**	**MovieLens10M**	**Globo**	**Total**
None	950	425	2724	26707	15417	46223
1	402	193	1186	12106	17503	31390
2	408	114	855	10422	34342	46141
3	338	55	383	4347	13234	18357
4	697	36	232	3157	13085	17207
5	895	40	195	2971	17341	21442
6	550	28	171	2932	17087	20768
7	1303	49	289	6524	66989	75154

Table 4: Number of Algorithms with Zero NDCG Score Across Datasets

5 Discussion

In light of the previous section, it stands clear that evaluating algorithm performance at the user level can help us to determine the most suitable recommendation algorithm for a specific user. The performance of each algorithm in recommending relevant items to users may not be included in the overall NDCG of that algorithm. Every algorithm works differently for different users. So, if we can Identify a more suitable algorithm to the corresponding user, it may enhance the quality of recommendation as well as the satisfaction of that user. Our results suggest a way to identify the best suitable algorithm for each user.

The per-user NDCG scores are different compared to the overall NDCG scores of the algorithms. The variation in per-user NDCG score shown in the given result suggests that each algorithm works differently for users. While some algorithms performed comparatively well for some users across different datasets, others struggled. Not even one algorithm performed exceptionally well for all the users. This suggests that no single algorithm can fulfill the relevant recommendations for all the users. Evaluating algorithms on a user level can be critical in tailoring new recommendation methods for every user, based on their profiles and interaction patterns.

A substantial proportion of users have an NDCG score of zero with some algorithms, across a dataset. For instance, the NMFItemToItem algorithm performed poorly on the Globo dataset at the user level and 73% of users scored zero in this case. Additionally, the users who scored NDCG close to the overall NDCG of the algorithm are only in the range between 10 to 35%. This fact indicates that the overall NDCG cannot fully represent how an algorithm performed for users, whose behaviour and interactions are different.

When we consider the aggregated distribution of zero score across all algorithm-dataset pairs, a significant proportion of users scored non-zero NDCG across all algorithms. Every algorithm we tried was able to give somewhat relevant recommendations to different users. Out of all users, 16.7% received recommendations that were somewhat relevant across all algorithms evaluated.. In contrast, all algorithms failed to provide 27.2% of total users with relevant recommendations. This highlights the importance of including more state-of-the-art algorithms capable of capturing the complex factors that influence recommendation relevance for individual users.

For a proportion of users, only a single algorithm was able to recommend effectively. While certain algorithm were able to catch some user's preferences and engagement patterns that affect the quality of recommendations, others failed. This performing algorithm is notably better for these users compared to others in understanding their preferences. Using an appropriate algorithm specifically tailored to user patterns can improve the quality of recommendations that the user achieves. However, the impact of this evaluation at the user level on more complex algorithms and dynamic datasets needs to be looked into.

6 Summary

The central aim of our project was to explore the possibility of evaluating algorithm performance at the individual user level to monitor whether certain algorithms perform better than others for users. We compared the performance of 7 algorithms such as ItemKNN, SLIM, NMF, SVD, NMFItemToItem, SVDItemToItem, and KUNN on datasets MovieLens100K, MovieLens1M, MovieLens10M, CiteULike, and Globo at the user level. Ranking sensitive metric NDCG was used for the evaluation with the recommendation list size set to 10. All datasets, algorithms selection, and pipeline construction were managed using the Recpack library. The results suggested that for users some algorithms performed better than other ones.

To our knowledge, we are the first to evaluate algorithm performance at the user level. Our result conveys that the algorithm is performing differently for individual users, this can guide more researchers and organizations to start inspecting more into tailoring the algorithm selection based on individual users. When we start identifying algorithms most suitable for every user, it may lead to more engaging and satisfying recommendations. Our study opens the door for the researchers in this area to explore different recommendation strategies. Our findings show that no single algorithm we evaluated consistently outperformed others, suggesting the importance of systematically assessing recommendation algorithms at the user level. However extensive experimentation and analysis are required for selecting the most effective algorithm for individual users. Improvements in this direction would enable more precise, user-centered recommendations in the real world.

7 Future Work and Limitations

The analysis is solely based on more static datasets, which may not include the dynamic behaviours or changes in preferences of the user. Another limitation may be the influence of the hyperparameter settings and implementation details of particularly selected algorithms. There may be more efficient hyperparameter values for selected algorithms. Also, we have only tried 7 algorithms, which may not include emerging state-of-the-art approaches, for instance, deep learning-based recommendation algorithms. We worked with limited resources in terms of memory and time, so additional datasets and algorithms are missing in this study.

Future work can extend the analysis by including more datasets and algorithms. More state-of-the-art methods for recommendation can be included in the study. It may reveal additional Insights into more personalized recommendations. An understanding of user behaviour patterns and characteristics could provide a much deeper comprehension of the way to optimize recommendations for individual users. We have used NDCG as an evaluation metric, which is a rank-sensitive metric. Using different evaluation metrics beyond NDCG would contribute to a more thorough analysis of algorithm performance. Lastly, incorporating an automatic framework for dynamically selecting the best-fit algorithm for each user could provide users with more relevant and satisfying personalized recommendations.

Bibliography

[1] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. Okuno, “An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, pp. 435 – 447, 03 2008.

[2] J. Bobadilla, F. Serradilla, and J. Bernal, “A new collaborative filtering metric that improves the behavior of recommender systems,” *Knowledge-Based Systems*, vol. 23, no. 6, pp. 520–528, 2010.

[3] R. Gonzalez Crespo, O. Sanjuán, J. Cueva Lovelle, B. Pelayo García-Bustelo, J. Labra Gayo, and P. Pablos, “Recommendation system based on user interaction data applied to intelligent electronic books,” *Computers in Human Behavior*, vol. 27, pp. 1445–1449, 07 2011.

[4] M. Deshpande and G. Karypis, “Item-based top-n recommendation algorithms,” *ACM Trans. Inf. Syst.*, vol. 22, p. 143–177, Jan. 2004.

[5] X. Ning and G. Karypis, “Slim: Sparse linear methods for top-n recommender systems,” pp. 497–506, 12 2011.

[6] K. Verstrepen and B. Goethals, “Unifying nearest neighbors collaborative filtering,” in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys ’14, (New York, NY, USA), p. 177–184, Association for Computing Machinery, 2014.

[7] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback

datasets," in *2008 Eighth IEEE International Conference on Data Mining*, pp. 263–272, 2008.

[8] O. Jeunen, I. Potapov, and A. Ustimenko, "On (normalised) discounted cumulative gain as an off-policy evaluation metric for top-n recommendation," 2024.

[9] Y. Wang, L. Wang, Y. Li, D. He, T.-Y. Liu, and W. Chen, "A theoretical analysis of ndcg type ranking measures," 2013.

[10] Y.-M. Tamm, R. Damdinov, and A. Vasilev, "Quality metrics in recommender systems: Do we calculate metrics consistently?," in *Fifteenth ACM Conference on Recommender Systems*, RecSys '21, p. 708–713, ACM, Sept. 2021.

[11] A. Said and A. Bellogín, "Comparative recommender system evaluation: benchmarking recommendation frameworks," in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, (New York, NY, USA), p. 129–136, Association for Computing Machinery, 2014.

[12] L. Michiels, R. Verachtert, and B. Goethals, "Recpack: An(other) experimentation toolkit for top-n recommendation using implicit feedback data," in *Proceedings of the 16th ACM Conference on Recommender Systems*, RecSys '22, (New York, NY, USA), p. 648–651, Association for Computing Machinery, 2022.

[13] F. Harper, "The movielens datasets," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, pp. 1–19, 12 2015.

[14] G. de Souza Pereira Moreira, D. Jannach, and A. M. da Cunha, "Contextual hybrid session-based news recommendation with recurrent neural networks," *arXiv preprint*, vol. arXiv:1904.10367, p. 49, 2019.

[15] H. Wang, B. Chen, and W.-J. Li, "Collaborative topic regression with social regularization for tag recommendation," in *IJCAI*, 2013.