Reproducing Popularity Bias in Recommendation: The Effect of Evaluation Strategies

SAVVINA DANIIL, Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
MIRJAM CUPER, National Library of the Netherlands, The Hague, The Netherlands
CYNTHIA C. S. LIEM, Delft University of Technology, Delft, The Netherlands
JACCO VAN OSSENBRUGGEN, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
LAURA HOLLINK, Centrum Wiskunde & Informatica, Amsterdam, The Netherlands

The extent to which popularity bias is propagated by media recommender systems is a current topic within the community, as is the uneven propagation among users with varying interests for niche items. Recent work focused on exactly this topic, with movies being the domain of interest. Later on, two different research teams reproduced the methodology in the domains of music and books, respectively. The results across the different domains diverge. In this paper, we reproduce the three studies and identify four aspects that are relevant in investigating the differences in results: data, algorithms, division of users in groups and evaluation strategy. We run a set of experiments in which we measure general popularity bias propagation and unfair treatment of certain users with various combinations of these aspects. We conclude that all aspects account to some degree for the divergence in results, and should be carefully considered in future studies. Further, we find that the divergence in findings can be in large part attributed to the choice of evaluation strategy.

CCS Concepts: • **Information systems** → **Recommender systems**; • **General and reference** → **Evaluation**; • **Applied computing** → *Media arts;*

Additional Key Words and Phrases: Recommender systems, popularity bias, evaluation, reproduction

ACM Reference format:
Savvina Daniil, Mirjam Cuper, Cynthia C. S. Liem, Jacco van Ossenbruggen, and Laura Hollink. 2024. Reproducing Popularity Bias in Recommendation: The Effect of Evaluation Strategies. *ACM Trans. Recomm. Syst.* 2, 1, Article 5 (March 2024), 39 pages.
https://doi.org/10.1145/3637066

1 INTRODUCTION

In this article we reproduce three papers that study popularity bias in media recommender systems. Websites that host media content are known to employ recommender systems to filter through the content and provide the user with personalized suggestions. In the case of collaborative filtering, neither explicit demographics of the user nor information about the content are needed to encode

Authors' addresses: S. Daniil and L. Hollink, Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, North Holland, 1098 XG, The Netherlands; e-mails: s.daniil@cwi.nl, l.hollink@cwi.nl; M. Cuper, National Library of the Netherlands, Prins Willem-Alexanderhof 5, The Hague, South Holland, 2595 BE, The Netherlands; e-mail: mirjam.cuper@kb.nl; C. C. S. Liem, Delft University of Technology, Van Mourik Broekmanweg 6, Delft, South Holland, 2628 XE, The Netherlands; e-mail: c.c.s.liem@tudelft.nl; J. van Ossenbruggen, Vrije Universiteit Amsterdam, De Boelelaan 1111, Amsterdam, North Holland, 1081 HV, The Netherlands; e-mail: jacco.van.ossenbruggen@vu.nl.

2770-6699/2024/03-ART5 $15.00
https://doi.org/10.1145/3637066

their taste, but only consumption and browsing history. Despite the lack of explicit input of user or item characteristics in the system, collaborative filtering approaches are still known to suffer from bias [34]. Popularity bias has been identified as a relevant issue with negative implications from a multi-stakeholder perspective [2]. In short, popularity bias is the algorithmic phenomenon where items already popular in the users' profiles tend to become even more popular due to being disproportionally recommended.

In the context of media recommenders, different research teams have studied the impact of popularity bias on the users. Specifically, they focused on the extent to which it impacts them disproportionally based on their interest for popular items, which they refer to as unfairness. In 2019, Abdollahpouri et al. [3] published a paper called "The Unfairness of Popularity Bias in Recommendation" where the propagation of popularity bias by different algorithms and for different user groups was reported in the setting of movie recommendation. Two subsequent papers reproduced the work to evaluate the same phenomenon in music recommendation (Kowald et al. [26]) and book recommendation (Naghiaei et al. [27]). The papers used a similar process and metrics to evaluate the unfairness of popularity propagation, but different datasets, data preprocessing, and some different algorithms. While all studies reported on the same types of results, diverging results were presented. Both Kowald et al. [26] and Naghiaei et al. [27] proposed that the differences in data characteristics might be the cause.

The studies take significant steps in the direction of understanding the unfairness of popularity bias, and a new metric for popularity bias is proposed by Abdollahpouri et al. [3]. The effect of popularity bias on several baseline and state-of-the-art collaborative filtering algorithms is analyzed. The view of unfairness is user-centric, unlike previous work on the matter, which contributes to the significance of these studies in the field of recommender systems. It is, therefore, pertinent to comprehend the source of divergence in their reported results when it comes to whether certain algorithms propagate popularity bias, as well as the extent to which certain user groups are unfairly treated.

The following results were presented by all three studies:

- **Overall Popularity Bias Propagation:** The studies observed whether frequency of an item in the users' profile and in an algorithm's recommended list correlated (the higher the correlation the larger the bias), and whether only a few items were getting recommended to all users. Their results diverged in the following ways:
  - Abdollahpouri et al. [3] and Naghiaei et al. [27] found that only for certain algorithms there is positive correlation, with the correlation found by the latter being stronger than by the former.
  - Kowald et al. [26] found that this correlation exists for all tested algorithms rather than for only some of them.
- **Popularity Bias Propagation per User Group:** The studies evaluated unfairness of popularity bias propagation by using the $\%\Delta$ *Group Average Popularity* ($\%\Delta GAP$) metric, which is defined as the difference in average popularity of items in a user's profile and in an algorithm's recommended list, averaged for a group of users. Specifically, they calculated $\%\Delta GAP$ for different user groups defined by their propensity for popular items, and then compared the results between groups and algorithms. The higher the $\%\Delta GAP$ for a group, the more unfairly this group is treated.
  - Abdollahpouri et al. [3] and Naghiaei et al. [27] found that popularity bias is larger for users who prefer niche (i.e., not popular) items than for other user groups. Certain algorithms examined by Naghiaei et al. [27] did not propagate popularity bias for any of the user groups, which was not the case for Abdollahpouri et al. [3].
  - Kowald et al. [26] observed no such clear difference between the groups for any algorithm.

In this article, we perform an extensive reproduction study of the three above mentioned papers (Abdollahpouri et al. [3], Kowald et al. [26], Naghiaei et al. [27]). We are motivated by the divergence in results and wish to investigate and comprehensively report on the source. We do not only attempt to replicate and verify the individual claims, but also locate properties of the recommendation and evaluation process that have an impact on popularity bias. Therefore, our reproduction study allows us to draw conclusions further than the transferability of a claim to a different domain, which was the goal of the two reproduction studies (Kowald et al. [26], Naghiaei et al. [27]). By zooming in on these studies and understanding how their differences in implementation resulted in divergent results, we can offer insights and suggestions on the topic of popularity bias evaluation, and reproducibility in recommender systems in general.

First, we study the choices made by Kowald et al. [26] and Naghiaei et al. [27] when reproducing Abdollahpouri et al. [3]. We recognize the challenges that stem from either lack of clarity around or differentiation in the strategies the three studies used to evaluate popularity bias. We identify four aspects that are relevant in investigating the differences in results between the three studies:

(1) data; the studies use three datasets, with different characteristics such as size, sparsity and distribution of item popularity.
(2) algorithms; the studies evaluate mostly different algorithms, with some exceptions.
(3) division of users in groups; the studies define propensity for popular items differently and divide them accordingly.
(4) evaluation strategy; the studies make different choices in the testing process.

Second, we run a set of experiments in which we measure popularity bias with various combinations of these aspects. For example, we run the evaluation strategy of one paper on the dataset of the other paper. We find that evaluation strategy has a significant impact on the reported results. Specifically, certain algorithms trained on the same data and with the same view of popularity either show or not show propagation of popularity bias depending on the evaluation strategy. We believe that our observations can prove useful for future efforts to reproduce evaluations of recommender systems, as they give insight on how strategic choices affect the outcome even when the same evaluation metric is used.

2 RELATED WORK

2.1 Reproducibility in Recommender Systems

The ability to reproduce and examine published studies is valuable, as reproducibility of experimental results is a cornerstone of science [17]. However, the field of AI as a whole is known to face reproducibility issues [22]. Machine learning is highly dependent on the characteristics of the chosen training data, as well as parameter tuning to fix the inherent randomness of the algorithms [36]. Therefore, lack of published code and data render the replication of existing work by other researchers challenging [18]. Many reported results are irreproducible, though Gundersen and Kjensmo [17] found a significant increase in documentation over time. In the context of recommender systems, lack of reproducibility is recognized as one of the key components that have led the field into "a state of stagnation" [11]. While reviewing recommender systems papers published in big conferences such as KDD, IJCAI and SIGIR, Ferrari Dacrema et al. [14] found less than 50% of them to be reproducible. Cremonesi and Jannach [11] consider lack of incentive to be the main reason behind limited reproducibility, as the academic system's operation often motivates researchers to publish *more* instead of putting in the work to provide sufficient code, data and documentation.

Recently, AI conferences have started providing checklists to ensure that the submitted papers are sufficiently reproducible [28]. Additionally, many conferences such as RecSys, ECIR and SIGIR

have initiated reproducibility tracks where researchers can submit studies that reproduce, analyze or reflect on prior work. Gundersen et al. [16] provide a set of specific recommendations on how to ensure reproducibility in AI research. They motivate researchers by highlighting the importance of reproducibility for the improvement of science overall as well as the benefits for the researcher themselves. Beel et al. [5] outline actions that can make recommender systems research more reproducible, such as adopting practices from medical sciences, social sciences, and natural sciences, and conducting more comprehensive experiments, for example by varying model parameters and observing the effect. In this work, we attempt to further motivate reproducibility in recommender systems research by experimenting with various aspects of the recommendation process and showcasing the effect on the phenomenon of popularity bias.

2.2 Evaluation Strategy

In recommender systems research, reproducing the evaluation process followed in previous studies is not always trivial. To assist researchers in this endeavour, Said and Bellogín [29] describe the dimensions of evaluating recommender systems as dataset, data splitting, evaluation strategies, and metrics. Specifically, they identify and benchmark four stages of designing an evaluation protocol: data splitting, item recommendation, candidate item generation, and performance measurement. Application of metrics takes place in the final stage of evaluation, namely when measuring the performance of the scores produced during the recommendation process. Before measuring performance, it is necessary to define a crucial aspect of the evaluation strategy; generating candidate items for recommendation for each user, so that scores can be produced for them. The process of evaluating recommender systems includes reporting on results of commonly used metrics, which allows comparison between different algorithms to estimate their success in the context of the task at hand. To calculate the metrics in a way that meaningfully represent the success of the system, it is necessary that the evaluation strategy is suitable for the system, as well as for the metrics to be measured.

In terms of metrics, Gunawardana et al. [15] list a set of properties frequently taken into account when evaluating recommenders, such as accuracy, coverage, novelty, diversity, and so on. They point out that different applications have different needs, and thus the metrics to be evaluated should be chosen to appropriately reflect on the desired property. Said and Bellogín [29] argue that comparison of recommendation quality between different studies requires careful consideration of all stages of an evaluation protocol. They experimentally show that even when recommendation algorithms are similarly implemented, the results are not comparable when different evaluation strategies are used. In our study, we follow a similar approach of employing different evaluation strategies on the same task and comparing the results. We are prompted by variations in evaluation strategy choices made by published reproducibility studies, which highlights the importance of taking into account and addressing all evaluation steps in recommender systems research.

2.3 Popularity Bias

Studying and evaluating a specific phenomenon requires understanding its roots and effects. In the context of item popularity, it is known that consumption of media often follows a long tail distribution, where a few items are very popular and the rest are located in the heavy tail [4]. Recommender systems attempt to facilitate users in their effort to discover items appropriate for them, regardless of the items' position in the long tail. However, it has long been noted that collaborative filtering techniques can be prone to popularity bias, the phenomenon where popular items tend to be recommended over long-tail ones [8]. When an algorithm showcases such tendency, it may produce ranked lists with items not equally covered along the popularity tail [35]. Bellogín et al. [6] argue that common ranking metrics calculate overall assumed satisfaction of a population, and

therefore produce high values when the recommended list consists of mostly popular items, even when it is not personalized. Item popularity is not a de facto bad criterion for recommendation and can be leveraged to track item quality [9, 37]. However, very popular items are often more likely to be already known, which renders the recommendation of mostly popular items to a user potentially not useful for the user and the system in general [1].

In addition to the three studies examined in this article, other studies evaluate popularity bias in various contexts and propose methods to mitigate it [7, 20, 24, 38]. These studies vary in terms of methods to measure popularity bias and findings. Elahi et al. [13] find that propagation of popularity bias is dependent on the scenario and domain; in certain cases, popularity bias is reduced through the recommendation process. In a follow-up paper to Kowald et al. [26], Kowald and Lacic [25] study popularity bias by four collaborative filtering algorithms on all three datasets used by the three studies we are reproducing, namely MovieLens1M, LastFM and Book-Crossing. It is interesting to note that their results are similar across the three datasets, while they diverge from the results reported by Abdollahpouri et al. [3] and Naghiaei et al. [27]. This observation implies that the discrepancy in results showcased by the three studies cannot be explained solely by the differences between the datasets. In this work, we wish to investigate which aspects of the process account for the discrepancy. Our results and conclusions can contribute to consistency and reproducibility when evaluating popularity bias and other recommender systems phenomena by the research community.

3 OVERVIEW OF THE STUDIES TO BE REPRODUCED

In order to comprehend the differences in results between the three studies, we studied and collected the details of each approach to accurately reproduce their process. Note that while Kowald et al. [26] and Naghiaei et al. [27] made their code publicly available, we could not find a public repository for the code developed by Abdollahpouri et al. [3]. Therefore, we describe the characteristics of their study based on the text of the published paper, and private correspondence with the authors. Throughout the paper, we explicitly state whether our understanding is based on this correspondence.

3.1 Core Process

The studies followed similar processes to compare the popularity distribution in the data and in recommendations:

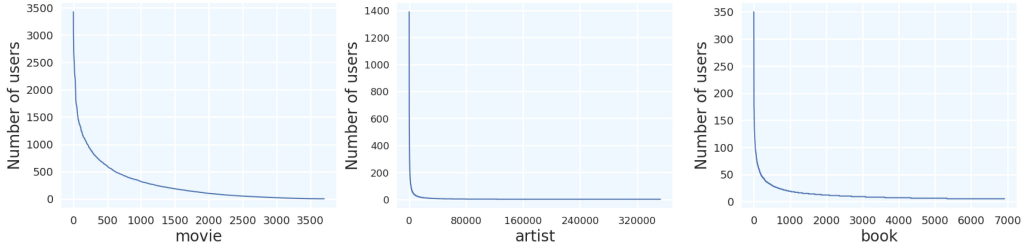
(1) Analyze distribution of popularity among items in given dataset.
(2) Label items as "popular" if they belong in the 20% most frequently rated items in the dataset.
(3) Analyze distribution of user propensity for popular items and divide users into three groups (*Niche*, *Diverse* and *Blockbuster focused*) accordingly.
(4) Set aside 80% of the ratings for training and 20% for testing.
(5) Train the algorithms on the training data.
(6) Recommend 10 items to each candidate user for each algorithm.
(7) Report on overall popularity bias propagation: compare the number of times each item is recommended with the number of times it is rated. Repeat for each algorithm.
(8) Report on popularity bias propagation per user group: compare average popularity of the items in profile with the items in recommendation for every user and average for each user group. Repeat for each algorithm.

3.2 Data

The studies used publicly available datasets which are commonly used in recommender systems literature. Abdollahpouri et al. [3] used MovieLens1M [19], Kowald et al. [26] used LastFM-1b

Table 1. The Characteristics of the Datasets used by the Three Studies

Dataset	#users	#items	#ratings	Sparsity
MovieLens1M	6,040	3,900	1,000,209	95.75%
LastFM-1b (subset)	3,000	352,805	1,755,361	99.83%
Book-Crossing (subset)	6,358	6,921	88,552	99.80%



(a) MovieLens1M (b) LastFM (subset) (c) Book-Crossing (subset)

Fig. 1. The long-tail distribution of item popularity in all datasets.

[32], and Naghiaei et al. [27] used Book-Crossing [39]. Kowald et al. [26] and Naghiaei et al. [27] processed their respective datasets in order to approach the size of MovieLens1M. Specifically:

— Kowald et al. [26] extracted 3,000 users from the original dataset, which contains 120,000 users. They also grouped the listening events into user-artist pairings. Subsequently, they scaled the number of times a user listened to an artist into a preference score from 0 to 1,000. Therefore, the algorithms in Kowald et al. [26] do not try to predict explicit rating, but rather the preference of a user towards an artist, which is represented by the number of times the user listened to this artist.

— Naghiaei et al. [27] only kept the explicit ratings from the original dataset. Afterwards, they removed users with more than 200 ratings. Finally, they removed users and items with very few ratings, until all users in the dataset had rated at least 5 items and all items in the dataset had been rated by at least 5 users.

Abdollahpouri et al. [3] do not explicitly mention whether they used the dataset intact or processed it before the analysis and recommendation process, and it was not clarified from our correspondence with the authors. Table 1 shows the characteristics of each dataset, namely number of users, items, ratings, and sparsity. Sparsity in the context of rating data refers to the percentage of possible user-item ratings that are missing from the data.

The resulting datasets differ in terms of size and sparsity. At the same time, item consumption is differently distributed among them. Figure 1 shows the number of users that rated each item in every dataset. While every figure shows that the rating data is skewed towards more popular items, the long-tail distribution is clearer in the subset of LastFM-1b than in the subset of Book-Crossing, and especially than in MovieLens1M.

3.3 Algorithms

Each study examined whether several different algorithms propagated popularity bias into their recommendations, as seen in Table 2. They all tested two baseline algorithms, MostPopular and Random. Other than these, the studies tested mostly different algorithms. Abdollahpouri et al. [3] tested four well known collaborative filtering algorithms. Kowald et al. [26] also tested four

Table 2. The Algorithms Tested by the Three Studies

	Abdollahpouri et al. [3]	Kowald et al. [26]	Naghiaei et al. [27]
UserKNN	✓	✓	✓
ItemKNN	✓		
UserKNN with means		✓	
UserItemAvg		✓	
SVD++	✓		
NMF		✓	✓
BMF	✓		✓
PMF			✓
WMF			✓
HPF			✓
NeuMF			✓
BPR			✓
VAECF			✓
MostPopular	✓	✓	✓
Random	✓	✓	✓

collaborative filtering algorithms, but partly deviated from the choices of Abdollahpouri et al. [3]. Specifically, they excluded SVD++ and ItemKNN to reduce computational cost, potentially due to the large number of items in the LastFM dataset. Finally, Naghiaei et al. [27] tested a total of nine collaborative filtering algorithms in order to cover a wider range of state-of-the-art approaches. Overall, the list of algorithms consists of Nearest Neighbour-based, Matrix Factorization-based, and Neural Network-based approaches. The only common collaborative filtering algorithm across all studies is UserKNN [31]. The divergence in the choice of algorithms between the original study and the reproduction studies raises the question of whether a different overall conclusion would be drawn if all studies tested the same set of algorithms.

3.4 Division of Users in Groups

The notion of item popularity is central in the three studies, and it is defined as frequency of either rating by the users (*popularity in profile*) or of recommendation by the algorithms (*popularity in recommendation*). Abdollahpouri et al. [3] deem an item popular in profile if it is one of the 20% most frequently rated items in the entire dataset. This choice is important since Abdollahpouri et al. [3] use this label to divide the users based on their propensity for popular items. Therefore, the fact that item popularity is differently distributed across the three datasets as shown in Figure 1 affects which users are considered *Niche*, *Diverse* or *Blockbuster focused*.

Specifically, the user division in Abdollahpouri et al. [3] happens as follows:

(1) The popularity of every item is calculated as the percentage of users who have rated it.
(2) The items are sorted based on their popularity.
(3) The 20% items with the highest popularity are labelled as popular.
(4) The *average propensity towards popular items* of every user is calculated as the average popularity of the items that this user has rated.
(5) The *popularity fraction* of every user is calculated as the percentage of the items in this user's profile that have the label *popular*.
(6) The users are sorted based on their popularity fraction.
(7) The top 20% users are labeled "Blockbuster focused".

Table 3. Overview of the User and Item Candidates on which Each Study based the Evaluation of Popularity Bias Propagation

Reference papers	User candidates	Item candidates
Abdollahpouri et al. [3]	Users in the test set	Items that the user has not rated in the training set
Kowald et al. [26]	Users in the test set	Items that the user has rated in the test set
Naghiaei et al. [27]	Users in the training set	All items

Note that, unlike Kowald et al. [26] and Naghiaei et al. [27], our description of the strategy used by Abdollahpouri et al. [3] stems from our correspondence rather than studying their code.

(8) The bottom 20% users are labeled "Niche".
(9) The remaining users are labeled "Diverse".

While Naghiaei et al. [27] follow the same process, Kowald et al. [26] differ; they do not use the popularity fraction to divide the users into groups. Instead, they use the *mainstreaminess score*, which is available for the users in the LastFM dataset. Mainstreaminess is defined as the overlap between a userâĂŹs listening history and the aggregated listening history of all users in the original dataset. The 3,000 users are strategically extracted by Kowald et al. [26]; 1,000 users with low mainstreaminess, 1,000 with medium, and 1,000 with high. In other words, the mainstreaminess score is used as a proxy for user propensity for popular items and the user groups are divided based on that instead of user preference for items labeled "popular" in the final dataset. Note that the mainstreaminess score cannot be computed on MovieLens1M or Book-Crossing, as it requires multiple interactions between users and items (i.e., play counts) [25].

3.5 Evaluation Strategy

A key difference between the studies' approaches is the strategy they adopt to evaluate the propagation of popularity bias. In all studies, the dataset of ratings is divided into a training and a test set with a 80-20% split. However, they differ in terms of which users they recommend items to, and when it comes to candidate item generation, a step of the evaluation process that was benchmarked by Said and Bellogín [29]. Said and Bellogín [29] discuss that this aspect of the evaluation protocol is crucial for the result, since by changing the item candidates for recommendation, a different ranking is evaluated. This is likely to have an effect on the measured popularity bias propagation. Table 3 shows an overview of the above mentioned characteristics of each study's evaluation strategy.

Let U^r denote the set of users that an algorithm recommends items to, and L_u the list of items that it ranks and chooses from to recommend to a user u in U^r.

Kowald et al. [26] recommend items to every user in the test set. To generate candidate items, they adopt the UserTest strategy [29]; the system only considers items that the user has rated in the test set. In other words, in their system U^r contains all users in the test set, and L_u for every user u in U^r contains every item i for which the rating (u, i) exists in the test set.

Naghiaei et al. [27] recommend items to all users in the training set. They adopt a version of the TrainItems strategy [29]; the system considers every possible item as candidates for a given user. In this case, U^r consists of all users in the training set, and L_u for every user u in U^r contains every item i. Note that Said and Bellogín [29] describe TrainItems as disregarding the items that the user has actually rated, but Naghiaei et al. [27] consider all items instead.

In our correspondence with Abdollahpouri et al. [3], they stated that they recommend items to all users in the test set. They also adopt the TrainItems strategy, but differently to Naghiaei et al. [27]. Specifically, for every user u in U^r which are all the users in the test set, L_u contains every item i that u has not rated in the training set. According to Said and Bellogín [29], this approach is suitable when simulating a real system where no test set is available.

3.6 Other Variations

While in our experiments we consider the aforementioned four aspects that vary across the three studies, there are other variations that we do not consider:

- Kowald et al. [26] and Naghiaei et al. [27] both used Python to run their experiments, but used different Python-based libraries to perform the training and recommendation process. Kowald et al. [26] used Surprise,[1] a toolkit for building and analyzing recommender systems that deal with explicit rating data [21]. Naghiaei et al. [27] used Cornac,[2] a framework for multimodal recommender systems [30]. Abdollahpouri et al. [3] stated through our private correspondence that they used Librec-auto,[3] a Python tool for running recommender systems experiments [33].
- In their paper, Abdollahpouri et al. [3] state that they tuned all collaborative filtering algorithms to reach a precision of 0.1, so that the results are comparable. On the other hand, neither Kowald et al. [26] nor Naghiaei et al. [27] tuned the algorithms. Instead, they used the default hyperparameters in the Python libraries.

4 EXPERIMENTAL SETUP

In order to investigate the cause of difference in results, we define a set of experiments that allows us to track which of the aspects that varies across the three studies affects whether popularity bias is propagated and whether the propagation unfairly impacts certain user groups. Each experiment consist of the following phases.

Training.

- Divide the dataset with ratings into training and test set using a 80-20% split.
- Train the algorithms on the training set.

Prediction.

- For every user u in U_r, predict ratings for item i in L_u based on each trained algorithm. (see notation in Section 3.5)
- Rank the items based on the predicted rating.
- Recommend to the user the top 10 items.

Evaluation.

- **Overall popularity bias propagation**: Note that the three studies indirectly use the concepts of correlation and item coverage to report on overall popularity bias, by plotting frequency of an item in profile versus in recommendation for every algorithm and visually observing whether there is correlation and whether certain items are almost never recommended. In this study, we quantify these concepts as follows:
  - For the set of items I calculate

$$Correlation = r(P_I, F_I) \tag{1}$$

where r is the Pearson correlation coefficient, P_I is the list of popularities of each item in the users' profiles, and F_I is the list of frequencies of each item in the users' recommended lists (see [10]).

[1] https://surpriselib.com/
[2] https://cornac.preferred.ai/
[3] https://librec-auto.readthedocs.io/

Table 4. The Characteristics of the Datasets used in Our Experiments

Dataset	#users	#items	#ratings	Sparsity
MovieLens1M	6,040	3,900	1,000,209	95.75%
LastFM-1b (subset of subset)	3,000	12,690	1,008,479	97.35%
Book-Crossing (subset)	6,358	6,921	88,552	99.80%

— For the set of items I calculate

$$Coverage = \frac{|R \cap I|}{|I|} \tag{2}$$

where R is the list of items recommended at least once.

— **Popularity bias propagation per user group**

— For every user group g calculate

$$\%\Delta GAP(g) = \frac{GAP(g)_r - GAP(g)_p}{GAP(g)_p} * 100\% \tag{3}$$

where $GAP(g)_p$ is the average popularity of the items in the users' profiles and $GAP(g)_r$ is the average popularity of the items in the users' recommended lists (see notation in Abdollahpouri et al. [3]).

— For every user group g calculate NDCG@10 [23]. Items for which no rating is available in the test set are assumed to have a utility of 0, as implemented by Ekstrand et al. [12].

We design the experiments by considering different choices for every aspect in which the studies deviated, and combining them in all possible ways. To train the models, we use the Cornac library, since it contains almost all algorithms needed. Similarly to Kowald et al. [26] and Naghiaei et al. [27], for every dataset we fix the random seed when splitting in training and test set for reproducibility. The code we developed for our experiments[4] has been made open source. In the subsequent subsections we summarize the choices for each of the four aspects.

4.1 Data

We train the algorithms using *MovieLens1M*, *LastFM*, and *Book-Crossing*. We use the entire *MovieLens1M* dataset, and the *Book-Crossing* subset that is used by Naghiaei et al. [27]. For computational reasons, we do not experiment with the entire *LastFM* subset that Kowald et al. [26] use. The large number of items makes some algorithms crash when combined with certain evaluation strategies, and we wish to evaluate as many combinations as possible. Instead, we sample by removing items with fewer than 20 ratings. This sampling does not decrease the number of users, but greatly decreases the number of items from 352,805 to 12,690. The characteristics of the resulting dataset can be seen in Table 4. To assess whether the sampling has a large impact on the conclusions, we compare our results with the results reported by Kowald et al. [26] and find that they are consistent.[5]

4.2 Algorithms

We train all collaborative filtering algorithms in Table 2, with the exception of *UserItemAvg* and *SVD++*, which are not available in Cornac. In total, we train 11 collaborative filtering algorithms. Note that we use the same default hyperparameters as Kowald et al. [26] and Naghiaei et al. [27].

[4] https://github.com/SavvinaDaniil/UnfairnessOfPopularityBias

[5] See Appendix, Section A.1.

4.3 Division of Users in Groups

We divide the users in groups in the following ways:

(1) ***PopularPercentage***: Divide the users based on the percentage of items in their profile that have the label "popular" with a 20%-60%-20% scheme, in the same way as Abdollahpouri et al. [3] and Naghiaei et al. [27]. An item gets the label "popular" if it is one of the 20% most frequently rated items in the dataset.
(2) ***AveragePopularity***: Divide the users based on the average popularity of items in their profile with a 20%-60%-20% scheme. This approach is not followed by any of the studies. Given that it may result into differently divided groups, and as it also does not depend on items being labeled "popular", it is interesting to investigate how it impacts the results in terms of whether certain user groups are unfairly treated by a recommender.
(3) ***Mainstreaminess***: Divide the users based on the mainstreaminess score in the same way as Kowald et al. [26]. This division is only tried on *LastFM*, given that the mainstreaminess score can only be calculated on this dataset.

Note that while each study used domain-specific terms to refer to each user group, for simplicity we will use the terms *Niche*, *Diverse* and *Blockbuster-focused* regardless the user division and dataset.

4.4 Evaluation Strategy

We apply all different evaluation strategies adopted by the studies. Specifically, the strategies vary with regards to which user and item are candidates for generating recommendations.

(1) ***Modified TrainItems*** (Naghiaei et al. [27]):
  (a) Recommend items to every user in the training set.
  (b) Choose out of all the items.
(2) ***UserTest*** (Kowald et al. [26]):
  (a) Recommend items to every user in the test set.
  (b) Choose out of all the items the user has rated in the test set.
(3) ***TrainItems*** (Abdollahpouri et al. [3]):
  (a) Recommend items to every user in the test set.
  (b) Choose out of all the items the user has not rated in the training set.

5 RESULTS

In this section, we present the results across all popularity bias metrics for every experiment. First, we discuss overall popularity bias propagation, and then we focus on propagation per user group.

5.1 Overall Popularity Bias Propagation

To evaluate overall popularity bias propagation for every combination of aspects, we report on correlation and coverage as described in Section 4. Tables 5–7 show the above mentioned values for each algorithm and each evaluation strategy, for *MovieLens1M*, *LastFM* and *Book-Crossing*, respectively. In addition to the correlation and coverage tables, we choose one algorithm, namely *NMF*, and plot frequency in profile versus in recommendation for every dataset and evaluation strategy. Figure 2 includes the resulting scatter plots. Given the large number of experiments, we include the scatter plots for the other algorithms in the Appendix, Figures 4 to 13. We analyze how the different aspects affect the results. The type of user division in groups is not relevant here as it does not relate to whether an algorithm overall propagates popularity bias.

5.1.1 Data. The resulting correlation and coverage vary across the three datasets. There is not a clear pattern of a dataset being more or less prone to popularity bias across all algorithms.

Table 5. Correlation and Item Coverage for the MovieLens1M Dataset

	Modified TrainItems		**UserTest**		**TrainItems**	
	Correlation	Coverage	Correlation	Coverage	Correlation	Coverage
UserKNN	**−0.0373**	0.0124	0.7913	0.6654	**−0.0375**	0.0108
ItemKNN	−0.0261	0.5229	0.9028	0.6832	−0.0439	0.4509
UserKNN with means	−0.0821	0.0688	0.8012	0.6805	−0.0838	0.0618
BPR	**0.4121**	**0.0216**	0.8448	0.5901	**0.4818**	**0.0586**
MF	0.1685	0.5982	0.8841	0.7337	0.1080	0.5990
PMF	0.3429	0.2415	0.8399	0.6786	0.2761	0.2515
NMF	**−0.0456**	0.1012	0.7977	0.6646	**−0.0474**	0.1012
WMF	0.3313	0.0157	0.8114	0.5923	0.3656	0.0229
HPF	**0.6524**	**0.1058**	0.8928	0.6141	**0.7535**	**0.1743**
NeuMF	**0.4864**	**0.0583**	0.8652	0.5947	**0.5657**	**0.1033**
VAECF	**0.6312**	**0.1913**	0.8906	0.6190	**0.7240**	**0.2715**
Mean	0.2576	**0.1762**	0.8474	**0.6469**	0.2784	**0.1914**

Table 6. Correlation and Item Coverage for the LastFM Dataset

	Modified TrainItems		**UserTest**		**TrainItems**	
	Correlation	Coverage	Correlation	Coverage	Correlation	Coverage
UserKNN	**0.0537**	0.0796	0.6586	0.3773	**−0.0170**	0.0403
ItemKNN	**0.5614**	0.6686	0.8881	0.7338	**−0.0178**	0.7097
UserKNN with means	0.1821	0.1422	0.6732	0.5031	−0.0259	0.0694
BPR	**0.3823**	**0.0094**	0.8192	0.2065	**0.4556**	**0.0177**
MF	−0.0001	0.0008	0.9039	0.7095	−0.0034	0.0351
PMF	−0.0081	0.0009	0.6223	0.2983	−0.0083	0.0009
NMF	**−0.0412**	0.0133	0.7565	0.5489	**−0.0413**	0.0144
WMF	0.2999	0.1981	0.8191	0.4063	0.2621	0.2062
HPF	**0.5819**	**0.0383**	0.8390	0.3407	**0.6836**	**0.0878**
NeuMF	**0.4844**	**0.0213**	0.8385	0.2332	**0.5586**	**0.0356**
VAECF	**0.6402**	**0.0866**	0.8574	0.3006	**0.7376**	**0.1256**
Mean	0.2851	**0.1145**	0.7887	0.4235	0.2349	0.1221

However, we notice in Table 7 that the correlation averaged over all algorithms is higher than in Tables 5 and 6 for a given evaluation strategy. In other words, when trained on the *Book-Crossing* dataset, the algorithms on average result in recommended lists where item frequency is more highly correlated with item popularity than when trained on the other two datasets. By this metric, the subset of *Book-Crossing* used by Naghiaei et al. [27] is on average more prone to popularity bias than *MovieLens1M* and *LastFM*.

Table 7 shows that the mean item coverage is also higher for *Book-Crossing* than for the other two datasets in Tables 5 and 6 for a given evaluation strategy. For example, given *Modified TrainItems*, the mean item coverage for Book-Crossing is 0.2892, for *MovieLens1M* 0.1762, and for *LastFM* 0.1145. Looking at Figure 1 and Table 4, we see that *Book-Crossing* is very sparse and popularity is scattered. Even very popular books have been rated by less than 350 users, in contrast with *MovieLens1M* where some movies have been rated by more than half of the users. It is therefore expected that item coverage is higher, as popularity is less concentrated. Note also that when *UserTest* is applied, the item coverage is higher for *Book-Crossing* than for *MovieLens1M*, and especially for *LastFM*, not just on average but for every individual algorithm. These observations

Table 7. Correlation and Item Coverage for the Book-Crossing Dataset

	Modified TrainItems		**UserTest**		**TrainItems**	
	Correlation	Coverage	Correlation	Coverage	Correlation	Coverage
UserKNN	**0.0407**	0.2672	0.9121	0.8130	**0.0385**	0.2588
ItemKNN	0.7249	0.9835	0.9115	0.8138	0.7275	0.9766
UserKNN with means	0.0454	0.3658	0.9105	**0.8148**	0.0421	0.3569
BPR	**0.4554**	**0.0014**	0.9208	0.7992	**0.4626**	**0.0023**
MF	0.0507	0.2177	0.9105	0.8152	0.0498	0.1608
PMF	0.1454	0.0673	0.9142	0.8103	0.1406	0.0657
NMF	**−0.0364**	0.2237	0.9124	0.8138	**−0.0365**	0.2137
WMF	0.8312	0.8951	0.9195	0.8047	0.8304	0.7180
HPF	**0.8362**	**0.0714**	0.9174	0.8067	**0.8410**	**0.0923**
NeuMF	**0.4202**	**0.0014**	0.9209	0.8000	**0.4244**	**0.0023**
VAECF	**0.7338**	**0.0864**	0.9177	0.8065	**0.7418**	**0.0971**
Mean	**0.3861**	**0.2892**	**0.9152**	**0.8089**	**0.3875**	**0.2677**

lead us to wonder whether high correlation is sufficient to conclude popularity bias propagation, when it is not combined with low item coverage.

5.1.2 Algorithms. The algorithms propagate popularity bias in different degrees. Tables 5 to 7 show that *BPR*, *HPF*, *NeuMF* and *VAECF* consistently result in relatively high positive correlation, as well as low item coverage for *TrainItems* and *Modified TrainItems*. In this sense, *BPR*, *HPF*, *NeuMF* and *VAECF* are consistently prone to popularity bias. Note that these algorithms were only tested by Naghiaei et al. [27], which means that the choice of algorithms may have a large impact on whether popularity bias will be observed by a study. On the other hand, Figure 2 shows that *NMF* shows no correlation for any dataset for *TrainItems* and *Modified TrainItems*, as is the case for *UserKNN*.

5.1.3 Evaluation Strategy. Evaluation strategy has a large impact on the reported result. Across all datasets and algorithms, there is a strong positive correlation between popularity in profile and in recommendation when *UserTest* is employed. This observation is in line with the conclusions of Kowald et al. [26] who used *UserTest* in their study, as well as their follow-up paper [25]. This is due to *UserTest* only recommending to a user items that they have already consumed in the test set. For example, if a test user has rated 8 items in the test set, only these items are candidates for recommendation and they will all be recommended to that user given a 10-item recommendation. In this case, it is reasonable that popularity in profile and in recommendation correlate; popular items are more likely to be in the users' test sets and therefore be candidates for recommendation.

It is also the case that for every dataset item, coverage is on average higher when *UserTest* is employed. For example, Table 5 shows that for *MovieLens1M* average coverage is 0.6469 when *UserTest* is employed, while only 0.1762 and 0.1914 when *Modified TrainItems* and *TrainItems* are employed, respectively. Given that candidates for recommendation are only the items from a user's test set, it follows that more items will be covered by the recommendation process as users have different tastes. This is especially the case for *Book-Crossing* given its sparsity. The consistency in results when *UserTest* is deployed prompts to consider that the popularity bias reported might mostly be a result of the evaluation process instead of the algorithm's functionality or data characteristics.

For *TrainItems* and *Modified TrainItems*, the results fluctuate per algorithm. Figure 2 shows that for all datasets, *NMF* propagates popularity bias when evaluated with *UserTest*, but does not with the other two strategies. *TrainItems* and *Modified TrainItems* show similar results overall. However, in some cases the fact that *TrainItems* excludes items that the user has rated in the training set does

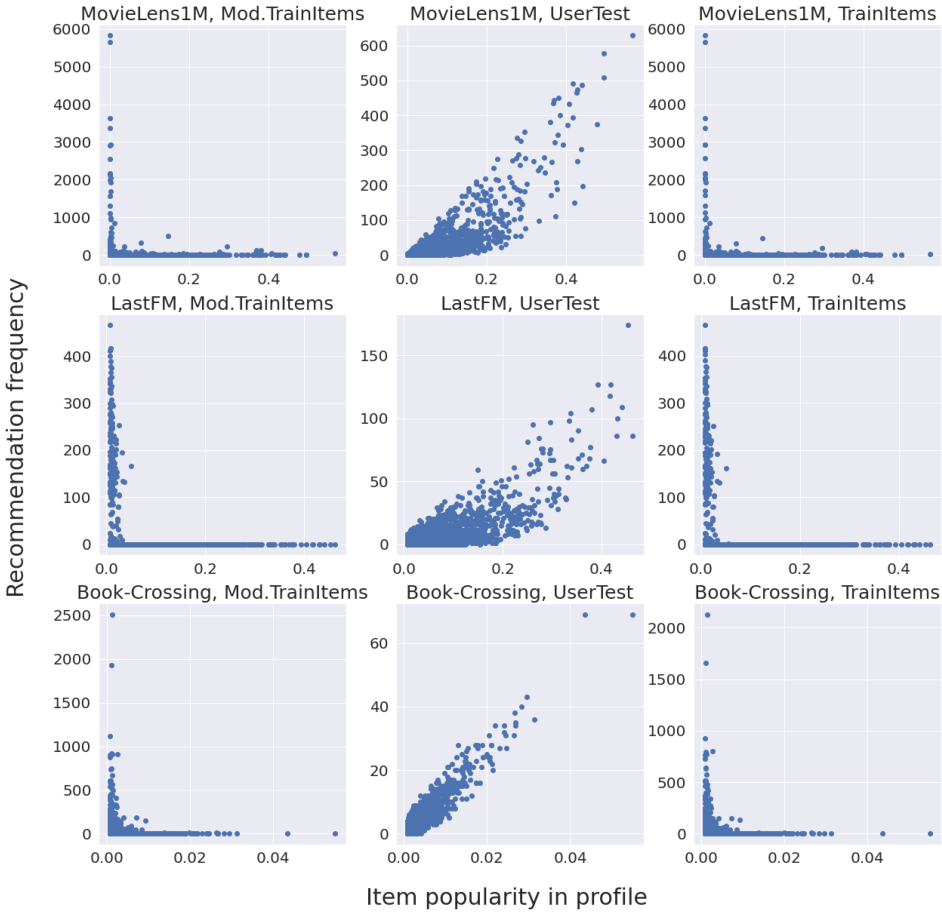


Fig. 2. Item popularity in profile versus frequency of recommendation by the algorithm NMF, for every dataset and evaluation strategy.

impact the conclusion on whether popularity bias is propagated. For example, *ItemKNN* trained on *LastFM* shows no correlation when *TrainItems* is employed, but positive correlation with *Modified TrainItems*.

5.2 Popularity Bias Propagation per User Group

To assess popularity bias propagation per user group, we report on the $\%\Delta GAP$ metric and on $NDCG@10$, as described in Section 4. Given the large number of experiments, we choose to present the results on *LastFM* since all three ways of dividing in user groups were possible on this dataset. Tables 8–10 show the $\%\Delta GAP$ value for each user group, algorithm and user division, using *Modified TrainItems*, *UserTest* and *TrainItems*, respectively. Tables 11–13 show $NDCG@10$ for each user group, algorithm and user division, using *Modified TrainItems*, *UserTest* and *TrainItems*, respectively. The results on the other datasets are included in the Appendix, Tables 15 to 26.

5.2.1 Algorithms. It is apparent in Tables 8 to 10 that certain algorithms consistently produce recommended lists with higher average popularity, while others do not. *BPR*, *HPF*, *NeuMF* and *VAECF* showcase high $\%\Delta GAP$ values for all groups, regardless the evaluation strategy. For

Table 8. %Δ*GAP* for Every User Group, Algorithm and Way of Dividing Users, when *Modified TrainItems* is Used

	Modified TrainItems								
	PopularPercentage			**AveragePopularity**			**Mainstreaminess**		
	N	D	BF	N	D	BF	N	D	BF
UserKNN	−32.6*	−32.6*	−47.8	−27.6*	−30.9*	−52.7	−53.2	−28.2**	−30.4*
ItemKNN	−20.8	−13.4*	−28.6	−17.0*	−11.6*	−33.3	−17.4	−13.6	−24.2
UserKNN with means	6.3**	−6.4*	−25.7	10.9**	−4.7*	−30.0	−34.7	−4.6*	6.0**
BPR	**596.6****	**352.6**	240.0	**611.0****	**358.2**	230.8	**379.8***	**355.4***	319.4
MF	**−49.0**	**−68.9**	**−78.3**	**−47.1**	**−68.6**	**−79.1**	**−67.6**	**−68.1**	**−71.4**
PMF	−69.7	−81.5	−87.1	−68.6	−81.3	−87.6	−80.8	−81.1	−83.1
NMF	−78.6*	−87.1*	−90.9	−77.8*	−87.0	−91.1	−86.1**	−86.8*	−88.4
WMF	**−2.8**	14.2**	1.2	**0.6**	17.8**	−7.0	−30.4	20.0*	32.5**
HPF	**150.4***	143.8*	125.2	**166.0****	148.6*	110.1	93.2	153.2*	**167.6****
NeuMF	**399.1****	**242.3***	157.3	**399.6****	**243.9***	159.2	**262.2***	**239.0***	213.4
VAECF	126.6	181.0**	156.3*	125.1	182.7**	153.7*	156.6	175.8**	169.4*

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 9. %Δ*GAP* for Every User Group, Algorithm and Way of Dividing Users, when *UserTest* is Used

	UserTest								
	PopularPercentage			**AveragePopularity**			**Mainstreaminess**		
	N	D	BF	N	D	BF	N	D	BF
UserKNN	21.9	35.2*	31.2*	23.7	36.7**	27.6	19.6	37.9*	39.0*
ItemKNN	−2.1	2.8	17.6**	−3.4	2.9	17.4**	9.8*	2.2	7.2
UserKNN with means	16.5	31.7*	29.1*	18.2	33.1**	25.6	9.1	33.3*	43.1**
BPR	**171.6***	**168.7***	108.1	**181.9***	**172.6***	98.4	**136.3**	**178.2****	142.0
MF	**−0.4**	**−0.1**	**0.7**	**1.0**	**0.6**	**−1.2**	**0.2**	**−0.5**	**0.6**
PMF	57.2	74.0**	62.3	59.7	77.4**	54.7	52.6	77.5*	75.0*
NMF	16.9	21.6	30.1**	15.7	23.1	27.2*	16.8	22.2	30.4**
WMF	20.6	40.2**	31.3	23.1	41.9**	27.0	13.7	42.7*	47.8*
HPF	61.3	89.9**	72.1	71.4	92.7**	62.7	52.8	96.3*	93.3*
NeuMF	**123.8***	**137.1***	90.6	**131.3***	**139.2***	84.9	**112.4**	**140.5****	114.8
VAECF	63.7	114.8**	87.1*	69.7	117.0**	80.5*	84.4	116.9**	100.7*

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

example, Table 8 shows that when *Modified TrainItems* is deployed and users are divided in groups based on *PopularPercentage* or *AveragePopularity*, *NeuMF* recommends to *Niche* users items almost 4 times more popular than they have rated in their profiles (i.e., the %Δ*GAP* is 399.1 and 399.6, respectively). Note that the same algorithms consisently result in high correlation and low coverage, as discussed in Section 5.1.2.

Tables 11 and 13 show that given *Modified TrainItems* and *TrainItems*, *BPR*, *HPF*, *NeuMF* and *VAECF* also result in high *NDCG*@10 regardless the user division. However, the *NDCG*@10 values are significantly lower for the *Niche* group. We can conclude that these algorithms tend to recommend mostly popular items to all users when trained on *LastFM*, at least when their default hyperparameters are used. On the contrary, *MF* mostly results in negative %Δ*GAP* values across Tables 8 to 10, meaning that the average popularity in the recommended lists is reduced compared

Table 10. %Δ*GAP* for Every User Group, Algorithm and Way of Dividing Users, when *TrainItems* is Used

	TrainItems								
	PopularPercentage			**AveragePopularity**			**Mainstreaminess**		
	N	D	BF	N	D	BF	N	D	BF
UserKNN	−57.9**	−82.0*	−87.3	−56.1**	−81.8**	−87.9**	−80.2	−81.6	−80.1
ItemKNN	−57.6**	−70.0*	−77.8	−56.8**	−70.6*	−76.5	−49.4**	−75.6*	−85.2
UserKNN with means	−51.8**	−81.1*	−87.3	−49.4**	−81.1*	−87.7	−77.5*	−81.2	−79.3
BPR	**580.6****	**323.6**	214.0	**596.4****	**328.1**	206.5	**362.1****	**325.0***	288.2
MF	**−53.6****	**−70.4***	**−79.8**	**−51.7****	**−70.1***	**−80.7**	**−70.5***	**−69.5***	−72.9
PMF	−70.1*	−81.8*	−87.2	−69.0	−81.6	−87.6	−81.0	−81.3	−83.2
NMF	−78.6	−87.1	−90.9	−77.8*	−87.0	−91.1	−86.1	−86.8	−88.4
WMF	−8.2	−5.0*	−19.5	−5.6*	−2.9*	−24.3	−35.7	−0.8	5.8**
HPF	**120.4****	100.1*	86.8*	**132.1****	103.4*	76.4	65.5	106.1*	**121.7****
NeuMF	**382.6****	**215.3***	138.1	**384.5****	**216.5***	140.3	**244.7****	**211.8***	187.9
VAECF	118.6	158.0**	133.4*	117.8	159.3**	131.3*	145.6	153.0	140.7

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 11. *NDCG*@10 for Every User Group, Algorithm and Way of Dividing Users, when *Modified TrainItems* is Used

	Modified TrainItems								
	PopularPercentage			**AveragePopularity**			**Mainstreaminess**		
	N	D	BF	N	D	BF	N	D	BF
UserKNN	0.007	0.009	0.002*	0.010	0.007	0.004	0.010	0.005	0.007
ItemKNN	0.007	0.009	0.008	0.008	0.007	0.011	0.013	0.007	0.005
UserKNN with means	0.009	0.007	0.001*	0.011	0.006	0.002	0.012	0.004*	0.002*
BPR	**0.188****	0.325	0.352	**0.188****	0.326	0.350	**0.251****	0.335	0.323
PMF	0.013	0.012	0.003**	0.013	0.012	0.004**	0.015	0.012	0.006*
NMF	0.006	0.008	0.006	0.007	0.008	0.005	0.007	0.010	0.004
WMF	0.200	0.222	0.213	0.206	0.227	0.192	0.155	0.242	0.251
HPF	**0.278****	0.340	0.359	**0.292****	0.335	0.357	**0.260****	0.355	0.378
NeuMF	**0.209****	0.339*	0.393	**0.219****	0.337*	0.390	**0.271****	0.350	0.351
VAECF	**0.297****	0.376	0.396	**0.294****	0.374	0.405	**0.326****	0.379	0.388

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly lower than for the two other groups, while *indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

to the users' profiles. Finally, some algorithms like *WMF* are inconsistent when it comes to the average popularity of the items they recommend to each group.

Additionally, Tables 8 to 10 show that *BPR* and *NeuMF*'s recommendations generally result to higher %Δ*GAP* for the *Niche* and *Diverse* groups compared to the *Blockbuster-focused* group. However, other algorithms do not showcase such consistency, as %Δ*GAP* fluctuates across evaluation strategies and user divisions. Consequently, it is challenging to deduce that a set of algorithms tend to treat *Niche* users unfairly while others do not. It might be that such conclusions can be drawn given a specific context, and not about an algorithm's inherent functionality.

5.2.2 User Division in Groups. The way the users are divided in groups in some cases determines whether the *Niche* user group is unfairly treated, as well as to what extent. For example, we see in Tables 8 and 10 that for *Modified TrainItems* and *TrainItems*, *HPF* results in higher %Δ*GAP* for *Niche* users with *PopularPercentage* and *AveragePopularity*, whereas with *Mainstreaminess* the

Table 12. *NDCG*@10 for Every User Group, Algorithm and Way of Dividing Users, when *UserTest* is used

	UserTest								
	PopularPercentage			**AveragePopularity**			**Mainstreaminess**		
	N	D	BF	N	D	BF	N	D	BF
UserKNN	0.669	0.648	0.640*	0.668	0.646	0.645	0.643*	0.636*	0.672
ItemKNN	0.621*	0.620*	0.659	0.621*	0.620*	0.661	0.619*	0.622*	0.643
UserKNN with means	0.671	0.656	0.657	0.671	0.656	0.658	0.648*	0.647*	0.684
BPR	0.655	0.668	0.677	0.651*	0.667	0.683	**0.625****	0.670*	0.706
PMF	0.665	0.646	0.662	0.661	0.644*	0.672	0.635*	0.642*	0.682
NMF	0.642	0.632	0.646	0.635	0.634	0.649	0.640	0.626	0.645
WMF	0.678	0.664	0.657	0.673	0.664	0.661	0.655*	0.654*	0.687
HPF	0.710	0.708	0.720	0.706	0.711	0.716	**0.673****	0.707*	0.752
NeuMF	0.657*	0.675	0.690	0.656*	0.675	0.690	**0.633****	0.680*	0.710
VAECF	0.695	0.693	0.707	0.693	0.694	0.705	**0.663****	0.694*	0.730

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. ** indicates that the corresponding result is significantly lower than for the two other groups, while * indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 13. *NDCG*@10 for Every User Group, Algorithm and Way of Dividing Users, when *TrainItems* is Used

	TrainItems								
	PopularPercentage			**AveragePopularity**			**Mainstreaminess**		
	N	D	BF	N	D	BF	N	D	BF
UserKNN	0.010	0.015	0.002*	0.014	0.012	0.006	0.015	0.010	0.010
ItemKNN	0.013	0.019	0.028	0.013*	0.018	0.032	0.025	0.018	0.016
UserKNN with means	0.015	0.014	0.004**	0.020	0.012	0.004**	0.019	0.011	0.006*
BPR	**0.236****	0.456	0.479	**0.242****	0.454	0.479	0.324**	0.460	0.466
PMF	0.013	0.013	0.003**	0.013	0.012	0.004*	0.015	0.012	0.006*
NMF	0.006	0.008	0.006	0.007	0.008	0.005	0.007	0.010	0.004
WMF	0.307	0.301	0.282	0.309	0.314	0.242**	0.225**	0.333	0.338
HPF	**0.389****	0.507	0.544	**0.409***	0.505*	0.530	**0.369****	0.532*	0.572
NeuMF	**0.279****	0.489*	0.550	**0.300****	0.488*	0.533	**0.366****	0.505	0.508
VAECF	**0.448****	0.556*	0.594	**0.458***	0.555*	0.587	**0.469****	0.565	0.591

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. ** indicates that the corresponding result is significantly lower than for the two other groups, while * indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Blockbuster-focused group receive the highest increase in average popularity. Additionally, *PopularPercentage* and *AveragePopularity* also sometimes result in different conclusions on which group is unfairly treated. Table 8 shows that given *Modified TrainItems*, *WMF* recommends more or less popular items to the *Niche* group depending on which user division is assumed (%Δ*GAP* of −2.8 for *PopularPercentage* and 0.6 for *AveragePopularity*).

PopularPercentage and *AveragePopularity* mostly result in similar trends in terms of which group receives unfair treatment. On the other hand, *Mainstreaminess* often leads to different conclusions than *PopularPercentage* and *AveragePopularity*. This is somewhat expected since *Mainstreaminess* is not dependent on the propensity for popularity within the given data, but the *mainstreaminess score* that characterizes the underlying population where this subset of *LastFM* stems from. Therefore, in order to conclude unfair treatment of a group, it is necessary to define the characteristics of

the group in question and whether we refer to specific behavior within the training dataset or we incorporate external information as well.

5.2.3 Evaluation Strategy. As is the case for overall popularity bias propagation, the choice of evaluation strategy largely influences $\%\Delta GAP$. When *UserTest* is employed, almost all algorithms provide recommendations with increased average popularity compared to user profile for all user groups regardless the type of division (see Table 9). Even in cases where $\%\Delta GAP$ is negative, the decrease is small. Overall, when *UserTest* is deployed, the $\%\Delta GAP$ values do not vary between algorithms as much as given the other two evaluation strategies.

On the other hand, when *Modified TrainItems* and *TrainItems* are used, the results fluctuate between algorithms. Furthermore, the recommended lists of the algorithms which consistently perpetuate popularity bias for all groups (*BPR*, *HPF*, *NeuMF* and *VAECF*) showcase higher $\%\Delta GAP$ with *Modified TrainItems* compared to *TrainItems* across all types of user divisions (see Tables 8 and 10). This likely signifies that excluding items that a user has rated from the candidate list reduces popularity bias. Given that by definition many users have rated the popular items, then the popular items are excluded from many users' candidate items list when *TrainItems* is deployed.

The choice of evaluation strategy also has a large effect on $NDCG@10$. Table 25 shows that *UserTest* results in higher $NDCG@10$ than the other two strategies. Since the pool of candidate items is limited to a user's test items when *UserTest* is deployed, it follows that the recommended list will likely be similar to the 'ideal' list. Additionally, the produced $NDCG@10$ values are generally higher for *TrainItems* than for *Modified TrainItems* across algorithms and user division in groups. This is due to the exclusion of training items from a user's candidate list when *TrainItems* is deployed. In our calculation of $NDCG@10$, we only considered items to have a nonzero utility for the user if they were consumed unbeknownst to the system, i.e., the test items (see [12]).

6 DISCUSSION AND FUTURE WORK

The results show that data, algorithms, user division in groups, and evaluation strategy influence the conclusion on whether popularity bias is propagated and whether the users that have lesser propensity for popular items are disproportionately affected. The datasets have impactful differences, especially when it comes to distribution of item popularity; *Book-Crossing* is a very sparse dataset, which results in higher item coverage on average compared to the other two datasets, even when correlation between popularity in profile and in recommendation is also high. At the same time, the choice of which algorithms to include in the study influences the results; some matrix factorization algorithms (*HPF*, *NeuMF*, *VAECF*) consistently propagate popularity bias, and in most cases recommend disproportionately many popular items to the *Niche* users. User division in groups often defines whether unfairness can be concluded, especially when a proxy measure of propensity towards popular items is used that does not stem from the specific training dataset. Finally, evaluation strategy, and specifically the generation of user and item candidates is overall crucial given the effect it has on whether both overall popularity bias propagation and user group unfairness can be observed.

The results indicate that perceived propagation of popularity bias is sensitive to various aspects of the evaluation process that are often unaddressed. The fact that tweaking these aspects determines whether propagation of popularity bias can be concluded or not renders the individual conclusions of the studies unique to their context and set up, and less generalizable. We conclude that it is challenging to report on popularity bias as a phenomenon that persists as a result of an algorithm's functionality. While some algorithms we studied do consistently propagate popularity bias given our metrics, for most algorithms the results largely fluctuate depending on the other aspects. Similarly, even though the choice of dataset does have an impact, we showed that the

divergence in results across the reproduced studies could not be solely attributed to the datasets' characteristics.

Consequently, we recommend careful consideration of each of these aspects when popularity bias is evaluated. Future studies on the topic should reflect on:

— **Data**: Which domain does the data come from? What is the size of the dataset? How sparse is it? What is the long-tail item distribution?
— **Algorithms**: What type of optimization is performed? How sensitive is it to item popularity?
— **User division in groups**: How is user propensity for popular items defined in this domain? What characteristics would deem a user *Niche*? Is behavior in the dataset the relevant factor, or is external information needed?
— **Evaluation strategy**: Which exact aspect of popularity bias is the study evaluating? How can it be translated into choices for every step of the evaluation process?

Evaluation in the case of recommender systems is generally challenging. Offline evaluation, in particular, is restricted by the lack of data or lack of absolute ground truth. The fact that a user has rated an item highly does not necessarily mean that they prefer it to an unrated one; it can simply mean lack of awareness. In other words, high accuracy in offline evaluation does not equate to a successful system. As a result, the choice of evaluation strategy is often application-specific, which extends further than the choice of metrics. For example, the same algorithm might be differently evaluated when used in an application which recommends 5 items to a user instead of 10. Some metrics can only be evaluated with certain strategies because of the data they require. For example, Mean Absolute Error can only be calculated for user-item pairings that do exist in the dataset and thus *UserTest* would be appropriate in this case. However, other metrics leave room for different choices and there might not be one right way to evaluate them.

To appropriately evaluate a metric, it is important to consider what is the exact phenomenon it is intended to measure. While popularity bias is not a recent topic within recommender systems research, there is a certain lack of specificity around the exact meaning of it. Correlation, coverage, and difference in average popularity can all be useful in measuring some sides of popularity bias, among others. Having said that, their interpretation requires careful design of an evaluation process that answers the question we are wondering about. In the case of the three studies each evaluation strategy answers a different question:

— *Modified TrainItems* does not disregard the items a user has already consumed from the recommendation process. Such strategy is unsuitable when assessing general performance, as it leads to information leakage. It could be used to assess popularity bias in a system that does recommend to users items that they have already consumed and positively rated. However, the information leakage needs to be considered and accounted for.
— *UserTest* only recommends to a user items that they have already consumed unknowingly to the system (i.e., from the test set), which renders it inappropriate for evaluating overall popularity bias. However, it might still be interesting to observe whether the items a user has rated and are unseen to the system are differently ranked by the system than by the user, because of popularity bias.
— *TrainItems* measures whether a learned model propagates popularity bias into unseen user-item combinations. It is our belief that this strategy is the most appropriate to measure popularity bias, as it more closely resembles a real world scenario of learning the preferences of the users and recommending items to them that they have not yet rated.

Studies on the topic of popularity bias should account for these differences in evaluation strategy by specifying the research question, as well as the evaluation strategy that accompanies it. To ensure reproducibility of such studies, a way ahead could be to adapt the dimensions of evaluation benchmarked by Said and Bellogín [29] into a checklist for submissions in conferences and journals. Such an initiative can motivate researchers to critically think about the evaluation strategy they employ in their experimentation, and allow for easier comparison between the reported results of different studies.

Our study has limitations that future work should address. It is our intention to approach the phenomenon of popularity bias fundamentally by locating specific data and recommendation characteristics that instigate its propagation. While the aspects we identified through reproducing these studies are very important, there are additional ones we plan to consider. First, our reported results on UserKNN differ significantly from Abdollahpouri et al. [3] and Naghiaei et al. [27], even when the same evaluation strategy and data are used. The reason is presumably tuning and implementation; Abdollahpouri et al. [3] state that they tuned all the algorithms to reach similar precision in order to appropriately compare them. On the other hand, Naghiaei et al. [27] manually trained UserKNN instead of using Cornac like they did for the other algorithms. This prompts us to consider tuning an important aspect of the process as well, and future work should focus on the effect it has on the propagation of popularity bias. In practice, algorithms are designed to satisfy some accuracy metric instead of being trained with their default hyperparameters, and thus it is realistic to assume that tuning takes place. Second, by measuring propagation of popularity bias with cross-validation instead of one-shot prediction, we could account for random small differences between algorithms, and generalize their comparison. Finally, all three studies used different Python libraries to perform the recommendation process. For future work, we plan to explore the effect of potentially different implementations of the same algorithm across these libraries.

7 CONCLUSION

In this paper, we reproduced the analysis on the propagation of popularity bias by commonly used collaborative filtering algorithms performed by three studies using different datasets from the media domain. The studies evaluated overall popularity bias propagation, as well as whether users with niche tastes were unfairly treated by the system. The results reported differed for both evaluation tasks. We identified four aspects which varied across the three studies and could potentially account for the divergence in results: data, algorithms, division of users in groups, and evaluation strategy. We designed and carried out experiments to investigate to what extent each aspect impacted the results by combining all possible choices for each aspect. We found that all aspects affected the result to some degree. Evaluation strategy specifically largely accounted for the divergence, as the one employed by the study in the music domain resulted in reporting propagation of popularity bias for all datasets and all algorithms. We conclude that clarity around the evaluation strategy employed during the recommendation process is necessary to reproduce and compare analysis for different algorithms and datasets, especially for phenomena like popularity bias whose evaluation is not standardized in literature yet.

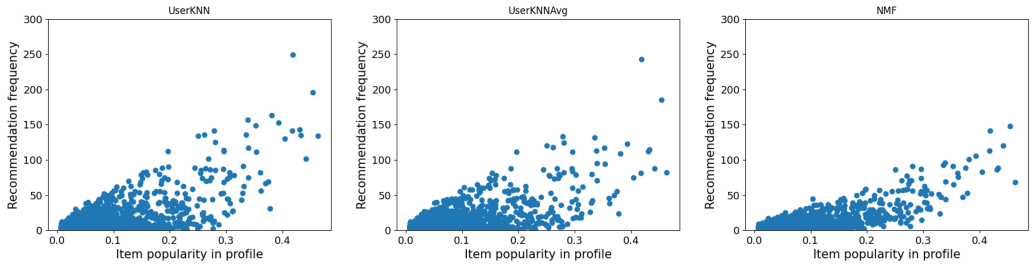
A APPENDIX

This section serves as supplementary material.

A.1 LastFM: Comparison between Complete and Filtered Dataset

As described in Section 4.1, for computational reasons we filtered the LastFM dataset. We assessed whether the sampling has a large impact on the conclusions by comparing our results with the results reported by Kowald et al. [26]. In order to ensure sufficient compareability between the

results, we applied the code provided by Kowald et al. [26] on our sampled dataset. We plot the relation between popularity in profile and recommendation frequency for every item in the sampled dataset, given the algorithms *UserKNN*, *UserKNN with means*, and *NMF*, which are the algorithms reported by Kowald et al. [26]. Figure 3 shows that there is a consistent correlation between popularity in profile and frequency of recommendation. This conclusion aligns with the results reported by Kowald et al. [26] and supports our argument that the consistent correlation stems from the evaluation strategy deployed, regardless the filtering of the dataset.



(a) UserKNN (b) UserKNN with means (c) NMF

Fig. 3. Relation between recommendation frequency and popularity in profile for the sampled LastFM dataset.

Additionally, we calculated the *MAE* metric for each user group of the sampled dataset, as seen in Table 14. While the exact numbers differ from the ones reported by Kowald et al. [26], the trends are consistent, with the *Niche* user group receiving the worst rating predictions for the given algorithms, evaluation strategy and division of users in groups.

Table 14. MAE Results for *UserKNN*, *UserKNN with Means* and *NMF*, Applied on the Sampled LastFM Dataset

User group	UserKNN	UserKNN with means	NMF
Niche	62.088**	55.422**	45.473**
Diverse	54.599	45.171	37.537
Blockbuster-focused	57.353	52.147	45.584
All	57.397	50.088	42.239

The worst results are always given for the *Niche* user group (statistically significant according to a t-test with $p < .005$ as indicated by **). Across the algorithms, the best results are provided by *NMF*.

A.2 Extensive Results

A.2.1 Overall Popularity Bias Propagation. We plot item popularity in profile versus frequency of appearance in the recommended lists of each algorithm, for every dataset and evaluation strategy, in Figures 4 to 13.

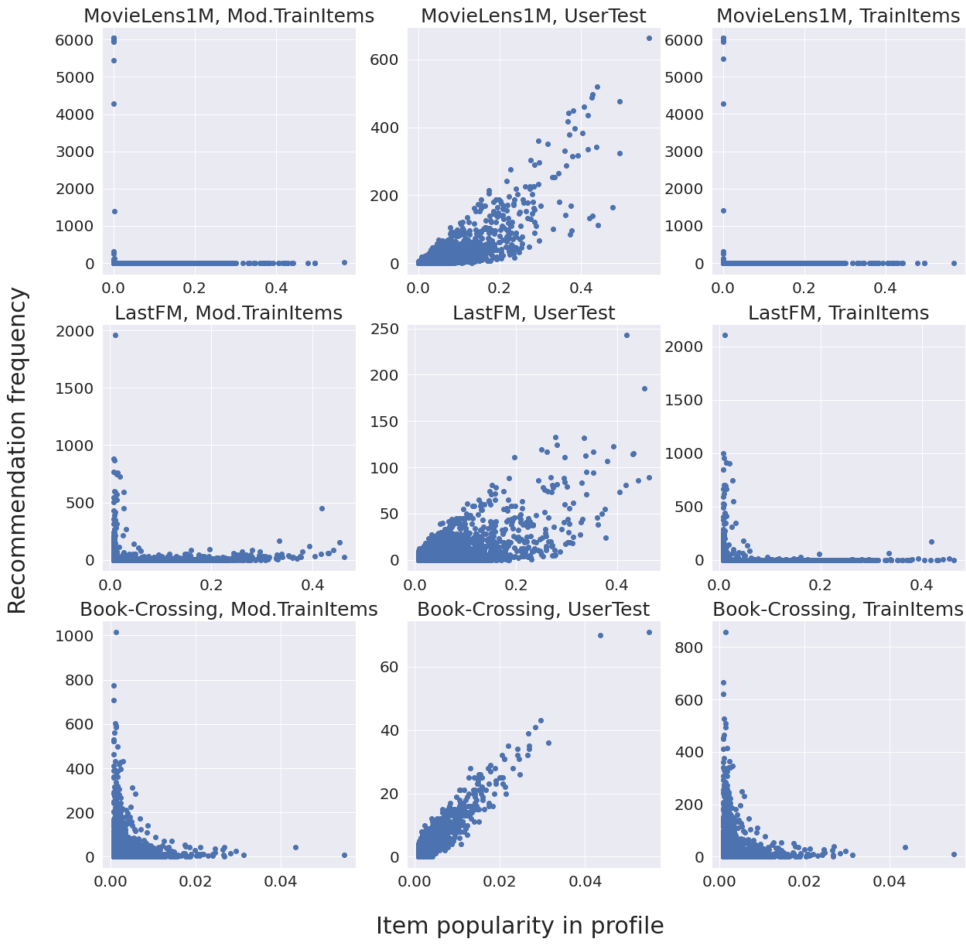


Fig. 4. Item popularity in profile versus frequency of recommendation by the algorithm UserKNN, for every dataset and evaluation strategy.

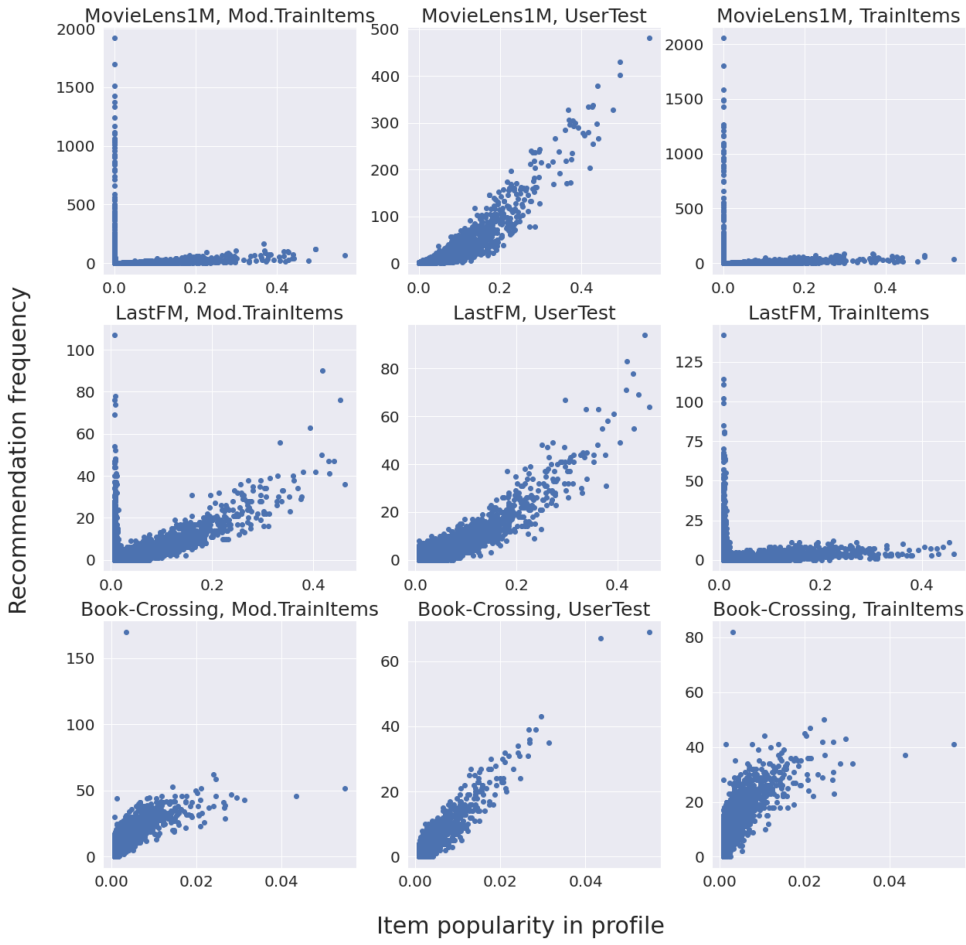


Fig. 5. Item popularity in profile versus frequency of recommendation by the algorithm ItemKNN, for every dataset and evaluation strategy.

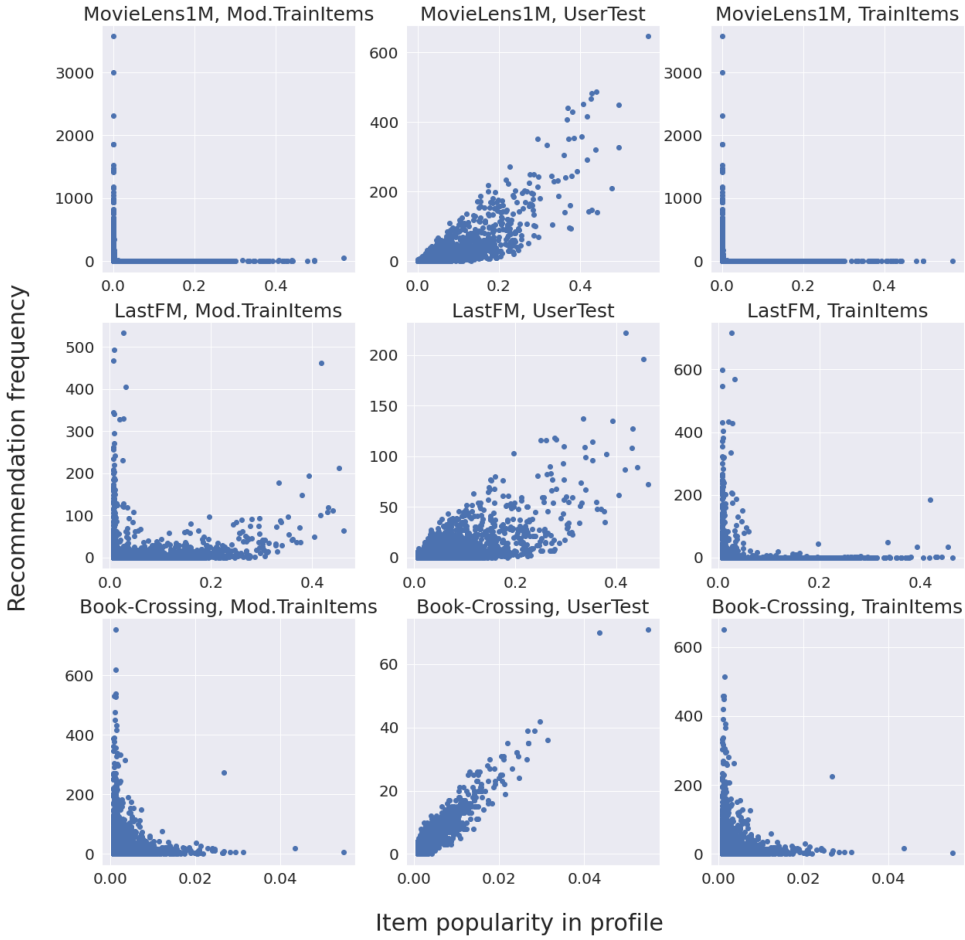


Fig. 6. Item popularity in profile versus frequency of recommendation by the algorithm UserKNN with means, for every dataset and evaluation strategy.

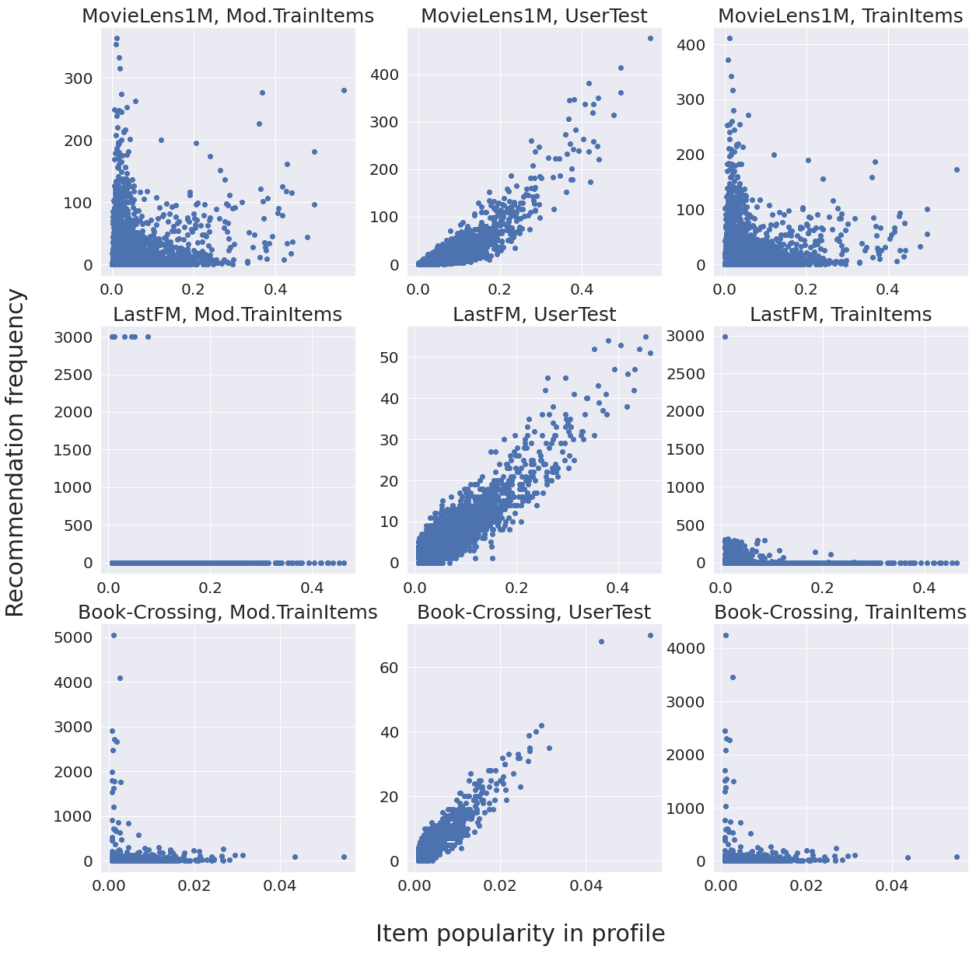


Fig. 7. Item popularity in profile versus frequency of recommendation by the algorithm MF, for every dataset and evaluation strategy.

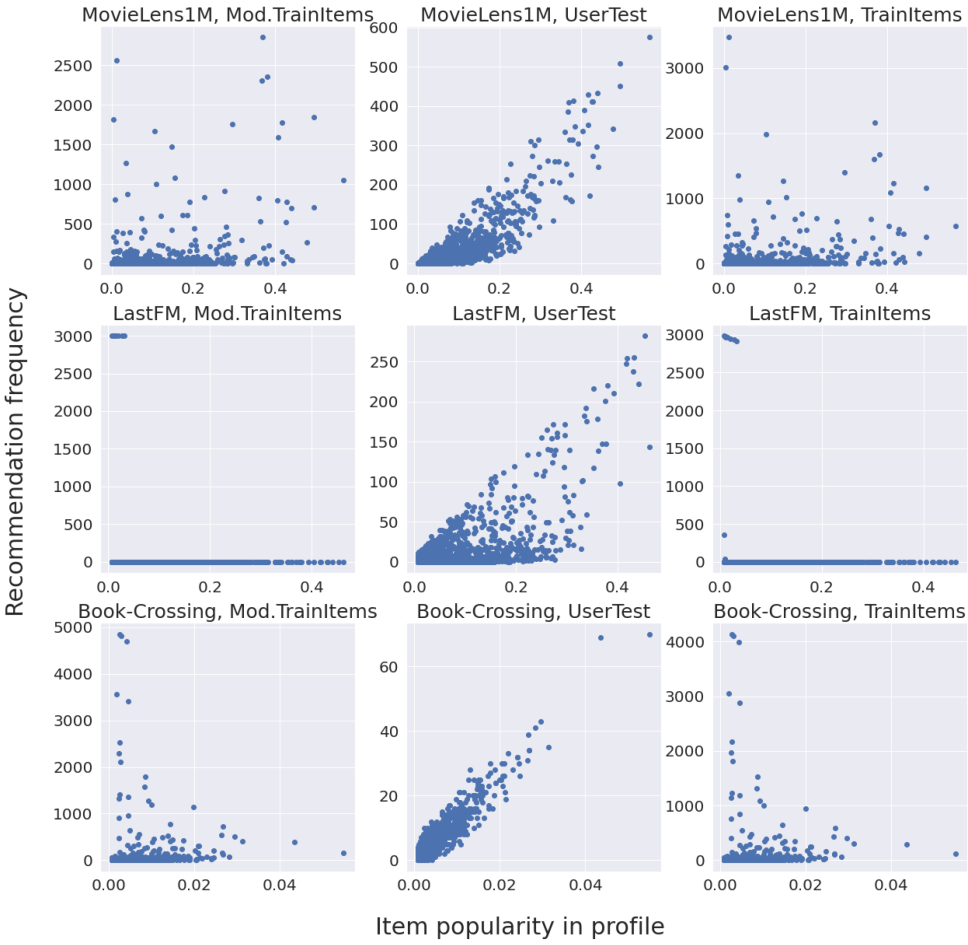


Fig. 8. Item popularity in profile versus frequency of recommendation by the algorithm PMF, for every dataset and evaluation strategy.

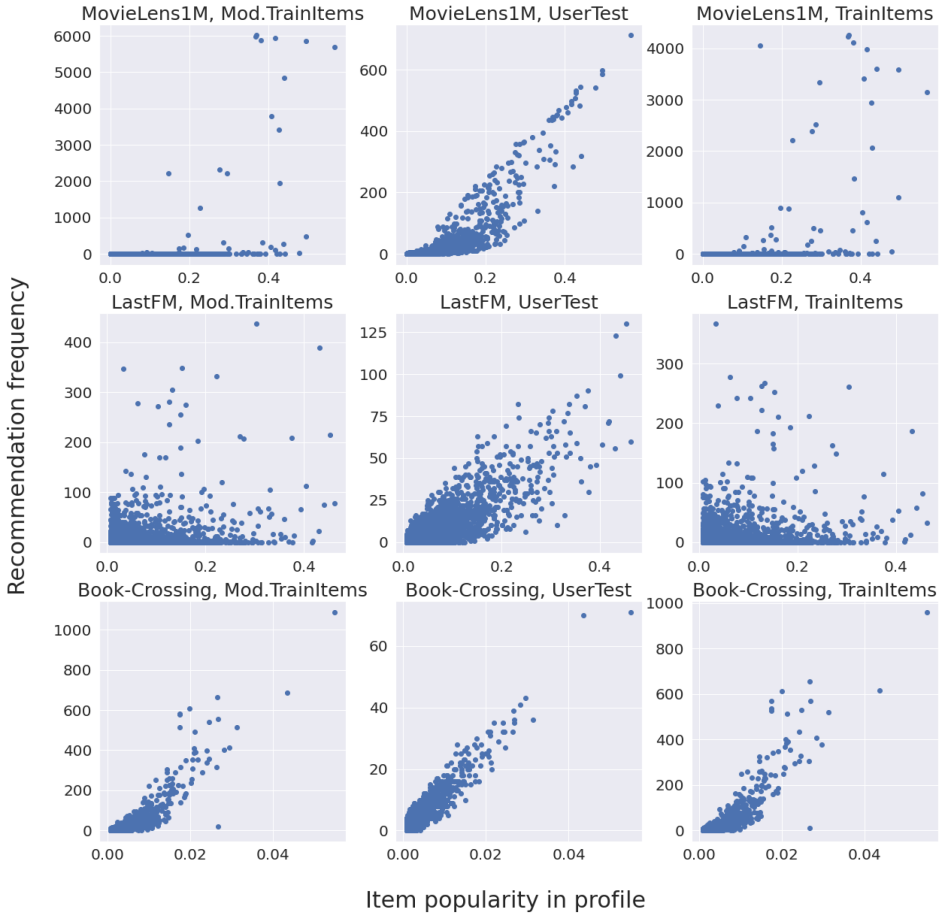


Fig. 9. Item popularity in profile versus frequency of recommendation by the algorithm WMF, for every dataset and evaluation strategy.

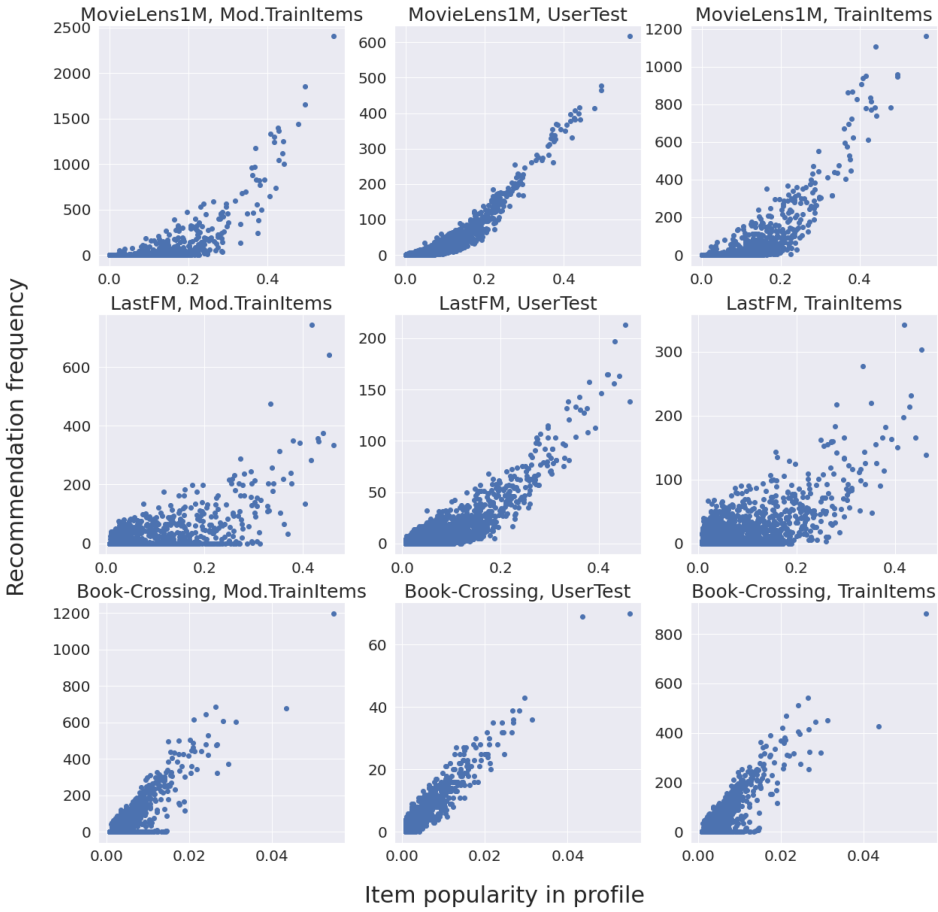


Fig. 10. Item popularity in profile versus frequency of recommendation by the algorithm HPF, for every dataset and evaluation strategy.

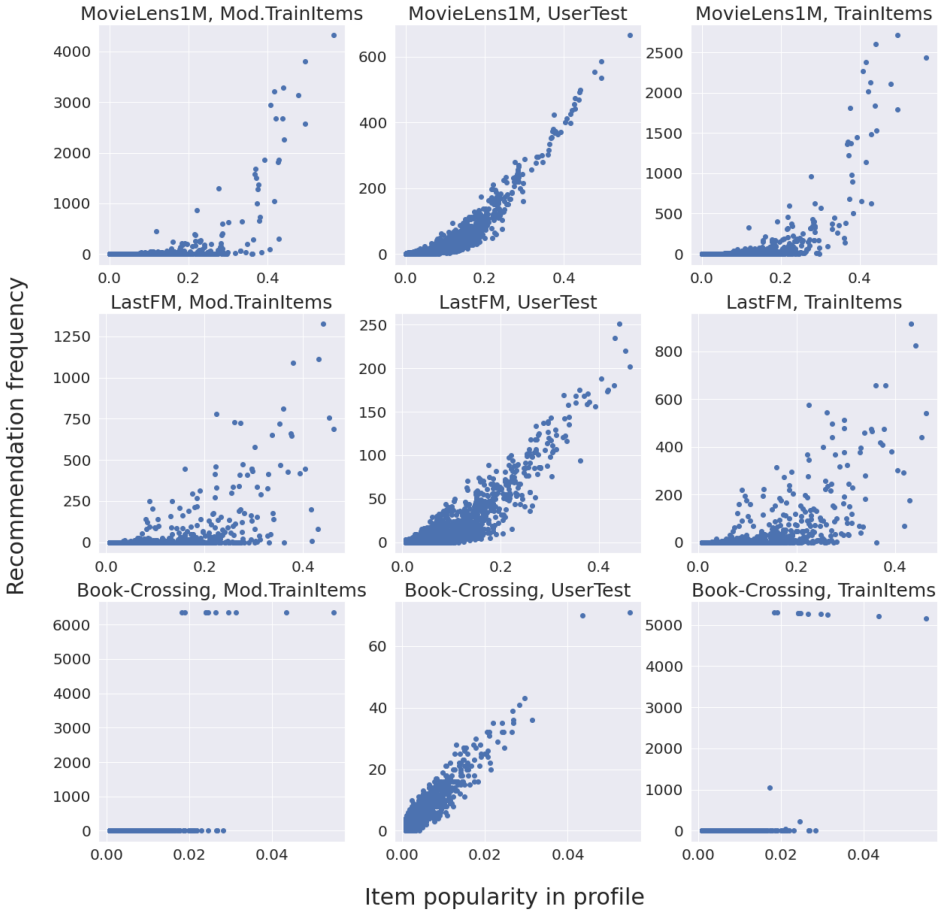


Fig. 11. Item popularity in profile versus frequency of recommendation by the algorithm NeuMF, for every dataset and evaluation strategy.

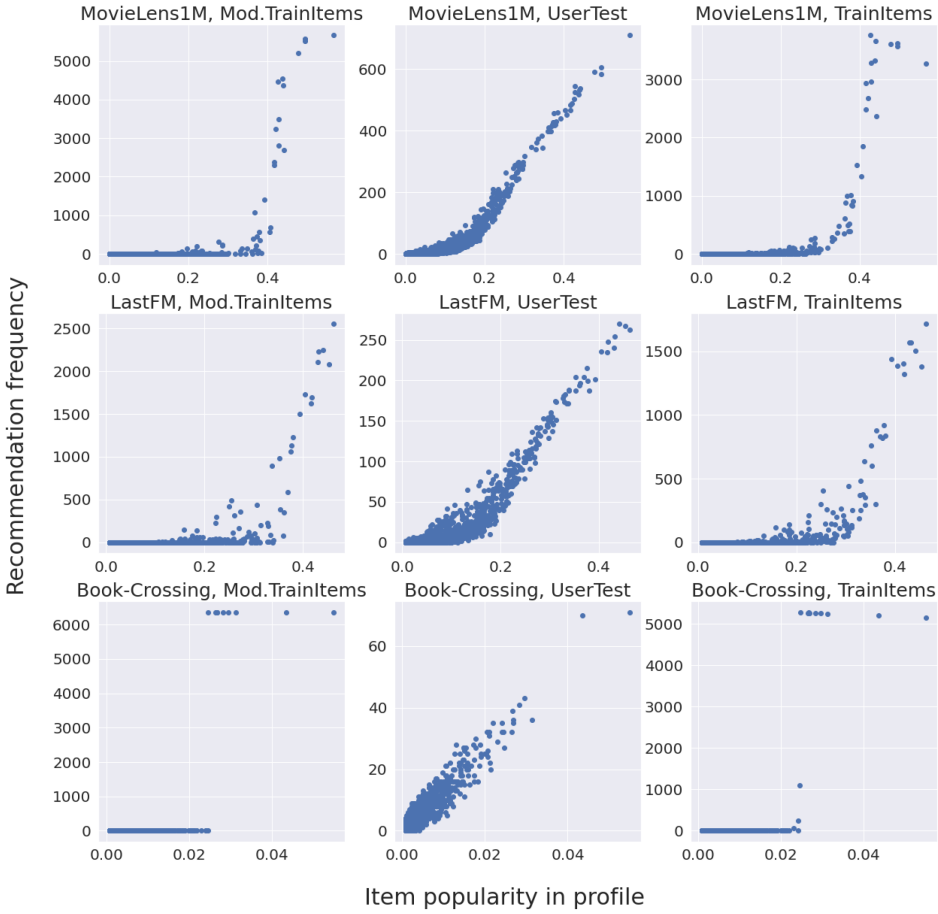


Fig. 12. Item popularity in profile versus frequency of recommendation by the algorithm BPR, for every dataset and evaluation strategy.

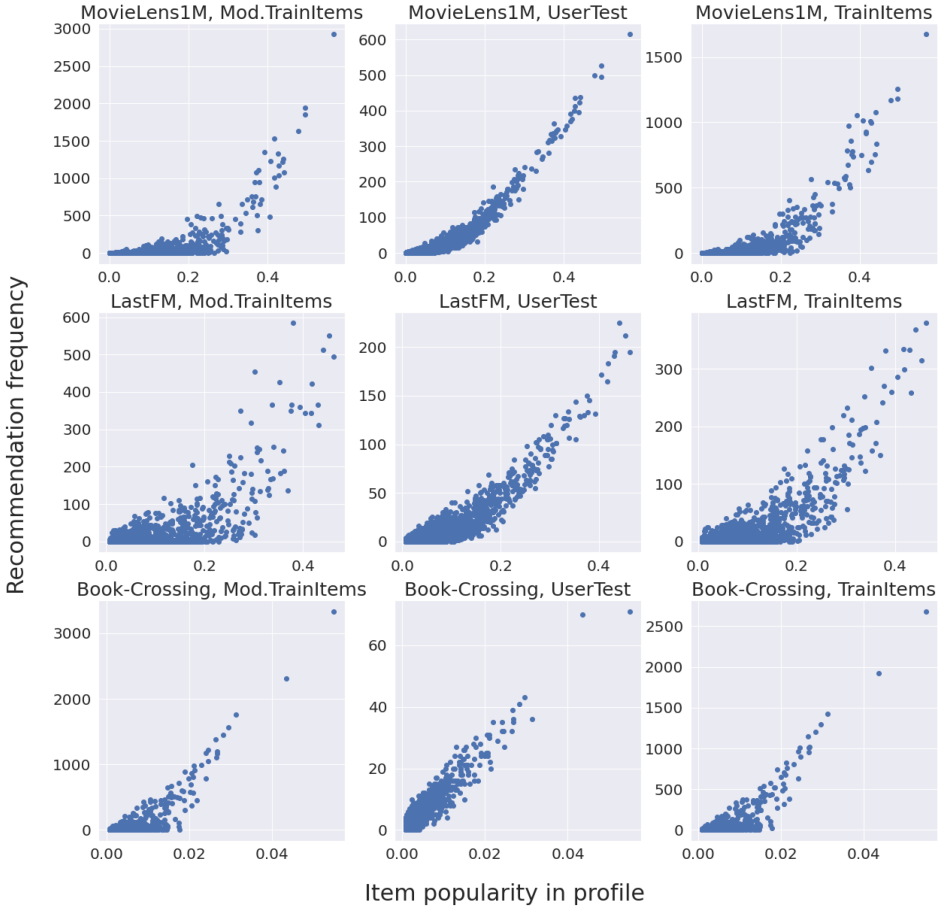


Fig. 13. Item popularity in profile versus frequency of recommendation by the algorithm VAECF, for every dataset and evaluation strategy.

A.2.2 Popularity Bias Propagation per User Group. We include the %Δ*GAP* values and *NDCG*@10 for the datasets MovieLens1M and Book-Crossing, for every algorithm, evaluation strategy and user division, in Tables 15 to 26.

Table 15. Percentage of Increase in Average Popularity of Items in Recommendation Versus in the MovieLens1M Dataset (%Δ*GAP*) for Every User Group, Algorithm and Way of Dividing Users, when *Modified TrainItems* is Used

	Modified TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	−99.4*	−99.5*	−99.7	−99.3	−99.5	−99.7
ItemKNN	−47.5**	−77.6*	−96.4	−35.6**	−78.9	−98.3
UserKNN with means	−98.4*	−99.1*	−99.5	−98.2	−99.1	−99.6
BPR	292.6	187.8	120.5	300.2	191.7	112.2
MF	−34.2**	−57.6*	−68.8	−28.2**	−57.6*	−71.0
PMF	71.7**	38.8*	20.0	73.7**	36.6*	7.9
NMF	−91.6	−94.3	−95.4	−91.4	−94.2	−95.6
WMF	249.8**	156.5*	95.3	258.4**	159.8*	87.4
HPF	126.3**	109.8*	83.0	130.6**	110.6*	80.4
NeuMF	214.8	156.6	108.2	221.0	159.5	101.6
VAECF	91.3	115.0**	91.3	93.2*	116.3**	88.2

**indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 16. Percentage of Increase in Average Popularity of Items in Recommendation Versus in the MovieLens1M Dataset (%Δ*GAP*) for Every User Group, Algorithm and Way of Dividing Users, when *UserTest* is Used

	UserTest					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	44.0	24.2	8.8	48.4	24.9	6.0
ItemKNN	55.5	21.7	6.2	61.9	21.6	4.4
UserKNN with means	37.4	21.8	8.7	41.0	22.5	6.2
BPR	98.5	55.5	20.1	110.4	56.6	13.8
MF	24.7	15.5	6.0	28.5	15.6	4.4
PMF	44.2	25.3	10.0	50.6	25.7	6.8
NMF	50.1	28.0	11.1	55.9	28.7	7.7
WMF	88.3	47.8	17.2	98.6	48.7	11.9
HPF	59.9	38.8	15.1	68.0	39.6	10.4
NeuMF	71.2	47.0	18.2	81.2	47.9	12.6
VAECF	47.3	40.8	16.7	54.5	42.0	11.7

**indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 17. Percentage of Increase in Average Popularity of Items in Recommendation Versus in the MovieLens1M Dataset (%Δ*GAP*) for Every User Group, Algorithm and Way of Dividing Users, when *TrainItems* is Used

	TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	−99.5	−99.7	−99.8	−99.5	−99.7	−99.8
ItemKNN	−60.6**	−82.9*	−97.1	−52.4**	−83.7*	−98.6
UserKNN with means	−99.6	−99.7	−99.8	−99.6	−99.7	−99.8
BPR	269.9	170.2	108.9	274.4	174.2	101.5
MF	−45.1**	−62.9*	−70.7	−42.3**	−62.5*	−72.5
PMF	45.5**	16.6*	3.0	45.5**	15.5*	−6.6
NMF	−93.0	−95.3	−96.2	−93.0	−95.1	−96.3
WMF	215.6*	129.5*	79.3	220.3**	133.2*	72.0
HPF	104.3**	82.9*	60.0	108.8**	83.4*	57.8
NeuMF	187.8	133.4	93.5	191.2	136.0	88.4
VAECF	82.2*	92.8**	72.5	83.6*s	93.7*	70.5

**indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 18. Percentage of Increase in Average Popularity of Items in Recommendation Versus in the Book-Crossing Dataset (%Δ*GAP*) for Every User Group, Algorithm and Way of Dividing Users, when *Modified TrainItems* is Used

	Modified TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	25.2**	−54.5*	−73.3	70.9*	−51.2	−80.3
ItemKNN	56.2**	−34.4*	−61.3	109.4**	−28.1	−72.4
UserKNN with means	22.1**	−54.9*	−73.5	67.2	−51.4	−80.8
BPR	1119.0*	523.9*	295.2	1434.7	572.3	228.4
MF	23.0**	−23.9*	−57.8	44.7**	−18.3*	−62.4
PMF	186.2**	45.0*	−5.2	253.0**	53.1*	−15.1
NMF	−47.1	−72.8	−82.7	−33.0	−70.8	−85.6
WMF	39.0	109.9**	81.4*	49.1	118.9**	67.0*
HPF	283.6**	155.3*	91.3	369.7**	168.4	71.7
NeuMF	1030.7	478.7	266.6	1323.5	523.6	204.6
VAECF	420.8**	281.1*	193.2	553.3**	300.9	158.4

**indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 19. Percentage of Increase in Average Popularity of Items in Recommendation Versus in the Book-Crossing Dataset (%Δ*GAP*) for Every User Group, Algorithm and Way of Dividing Users, when *UserTest* is Used

	UserTest					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	−5.5	1.3	4.1	2.5	−0.6	3.7
ItemKNN	−5.5	0.9	3.9	2.4	−1.0	3.6
UserKNN with means	−5.6	1.0	4.1	2.3	−0.9	3.6
BPR	−5.2	2.4	4.3	2.8	0.6	3.8
MF	−5.6	0.8	4.0	2.4	−1.0	3.6
PMF	−5.3	1.2	4.2	2.6	−0.6	3.7
NMF	−5.5	1.1	4.0	2.4	−0.7	3.5
WMF	−5.4	2.1	4.3	2.6	0.4	3.8
HPF	−5.3	1.8	4.2	2.6	0.0	3.7
NeuMF	−5.2	2.3	4.3	2.8	0.6	3.8
VAECF	−5.6	1.9	4.3	2.4	0.1	3.8

**indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 20. Percentage of Increase in Average Popularity of Items in Recommendation Versus in the Book-Crossing Dataset (%Δ*GAP*) for Every User Group, Algorithm and Way of Dividing Users, when *TrainItems* is Used

	TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	24.9**	−55.1*	−73.2	69.8**	−51.7	−80.2
ItemKNN	55.7**	−34.9*	−61.5	110.1**	−28.9*	−72.3
UserKNN with means	21.7**	−55.7*	−73.3	65.6*	−52.0	−80.7
BPR	1116.6	517.8	291.0	1434.6	568.3	220.3
MF	26.1**	−24.2*	−60.9	49.6**	−17.9*	−66.0
PMF	185.9**	41.8*	−8.4	256.0**	50.8*	−20.0
NMF	−47.2	−72.9	−82.9	−32.9	−71.0	−85.6
WMF	94.7	138.8**	92.2	107.0*	155.7**	68.7
HPF	271.4**	142.6*	82.3	364.7**	159.2	54.5
NeuMF	1027.0	470.5	260.6	1323.3	518.0	193.9
VAECF	419.1**	274.1*	191.1	554.3**	297.0	149.3

**indicates that the corresponding result is significantly higher than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 21. *NDCG*@10 in the MovieLens1M Dataset for Every User Group, Algorithm and Way of Dividing Users, when *Modified TrainItems* is Used

	Modified TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	0.004	0.000	0.000	0.004	0.000	0.000
ItemKNN	0.046	0.037	0.005**	0.059	0.033*	0.003**
UserKNN with means	0.002	0.000	0.000	0.002	0.000	0.000
BPR	0.316	0.336	0.320	0.311	0.337	0.322
MF	0.125	0.082*	0.040**	0.135	0.081*	0.032**
PMF	0.197	0.194	0.204	0.206	0.194	0.196
NMF	0.011	0.005*	0.004*	0.012	0.005*	0.004*
WMF	0.263	0.286	0.278	0.266	0.282	0.289
HPF	0.368	0.373	0.362	0.377	0.371	0.357
NeuMF	0.332	0.355	0.340	0.330	0.356	0.339
VAECF	0.357	0.376	0.381	0.356	0.380	0.370

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly lower than for the two other groups, while *indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 22. *NDCG*@10 in the MovieLens1M Dataset for Every User Group, Algorithm and Way of Dividing Users, when *UserTest* is Used

	UserTest					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	0.951**	0.957*	0.961	0.951**	0.957*	0.961
ItemKNN	0.949**	0.952	0.951*	0.951	0.952	0.951
UserKNN with means	0.953**	0.960*	0.964	0.953**	0.959*	0.964
BPR	0.943**	0.950*	0.958	0.944**	0.950*	0.958
MF	0.952**	0.957	0.960	0.953*	0.956*	0.961
PMF	0.965**	0.968	0.971	0.964**	0.968	0.971
NMF	0.952**	0.958*	0.963	0.953**	0.958*	0.964
WMF	0.954**	0.960*	0.965	0.956*	0.960*	0.965
HPF	0.949**	0.953*	0.959	0.949*	0.953*	0.960
NeuMF	0.944**	0.951*	0.957	0.944**	0.950*	0.957
VAECF	0.940**	0.951*	0.957	0.941**	0.950*	0.958

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly lower than for the two other groups, while *indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 23. *NDCG*@10 in the MovieLens1M Dataset for Every User Group, Algorithm and Way of Dividing Users, when *TrainItems* is Used

	TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	0.004	0.000	0.000	0.004	0.000	0.000
ItemKNN	0.135	0.075*	0.010**	0.162	0.068*	0.004**
UserKNN with means	0.002	0.000	0.000	0.002	0.000	0.000
BPR	0.477	0.489	0.444*	0.492	0.484	0.442**
MF	0.148	0.091*	0.042**	0.163	0.089*	0.033**
PMF	0.273	0.259	0.251	0.293	0.256*	0.239*
NMF	0.013	0.006*	0.004*	0.014	0.005*	0.004*
WMF	0.375	0.396	0.375	0.392	0.387	0.385
HPF	0.600	0.607	0.566*	0.630	0.603	0.549**
NeuMF	0.501	0.531	0.492*	0.516	0.529	0.484*
VAECF	0.584	0.595	0.579	0.605	0.596	0.553**

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly lower than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 24. *NDCG*@10 in the Book-Crossing Dataset for Every User Group, Algorithm and Way of Dividing Users, when *Modified TrainItems* is Used

	Modified TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	0.002	0.002	0.002	0.002	0.002	0.001
ItemKNN	0.015	0.009	0.005*	0.010	0.011	0.004*
UserKNN with means	0.002	0.001	0.002	0.003	0.001	0.001
BPR	0.007**	0.040*	0.068	0.000**	0.024*	0.122
MF	0.003	0.006	0.003	0.001*	0.006	0.004
PMF	0.004**	0.012	0.014	0.003**	0.011	0.019
NMF	0.002	0.002	0.000	0.001	0.002	0.000
WMF	0.026**	0.040*	0.059	0.029*	0.037*	0.063
HPF	0.005**	0.032*	0.054	0.007**	0.027*	0.068
NeuMF	0.005**	0.038*	0.064	0.000**	0.023*	0.113
VAECF	0.013**	0.052*	0.093	0.013**	0.042*	0.122

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly lower than for the two other groups, while *indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 25. *NDCG*@10 in the Book-Crossing Dataset for Every User Group, Algorithm and Way of Dividing Users, when *UserTest* is Used

	UserTest					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	0.971	0.968	0.972	0.972	0.968*	0.971
ItemKNN	0.974	0.965*	0.968	0.974	0.965*	0.967*
UserKNN with means	0.970	0.963**	0.969	0.970	0.964*	0.966
BPR	0.969	0.965	0.969	0.970	0.965*	0.970
MF	0.973	0.970	0.975	0.973	0.970	0.973
PMF	0.974	0.970	0.973	0.973	0.970	0.973
NMF	0.972	0.968	0.971	0.973	0.968	0.970
WMF	0.971	0.966**	0.973	0.972	0.966**	0.973
HPF	0.969	0.965	0.968	0.970	0.965*	0.969
NeuMF	0.970	0.964*	0.969	0.970	0.964**	0.970
VAECF	0.969	0.965	0.970	0.968	0.966	0.969

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. *indicates that the corresponding result is significantly lower than for the two other groups, while **indicates that it is significantly lower than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

Table 26. *NDCG*@10 in the Book-Crossing Dataset for Every User Group, Algorithm and Way of Dividing Users, when *TrainItems* is Used

	TrainItems					
	PopularPercentage			**AveragePopularity**		
	N	D	BF	N	D	BF
UserKNN	0.002	0.002	0.002	0.002	0.002	0.001
ItemKNN	0.016	0.009	0.006**	0.012	0.010	0.006
UserKNN with means	0.002	0.001	0.002	0.003	0.001	0.001
BPR	0.007**	0.042*	0.070	0.000**	0.026*	0.127
MF	0.003	0.006	0.004	0.001*	0.007	0.005
PMF	0.005**	0.013	0.017	0.003**	0.012	0.022
NMF	0.002	0.002	0.000	0.001	0.002	0.000
WMF	0.062*	0.067*	0.098	0.068*	0.064*	0.101
HPF	0.006**	0.038*	0.067	0.009**	0.032*	0.082
NeuMF	0.005**	0.039*	0.066	0.000**	0.024*	0.116
VAECF	0.018**	0.061*	0.108	0.018**	0.051*	0.138

N, *D* and *BF* signify *Niche*, *Diverse* and *Blockbuster-focused*. **indicates that the corresponding result is significantly lower than for the two other groups, while *indicates that it is significantly higher than for one of the other groups. Statistical significance was concluded based on a t-test with $p < 0.005$.

REFERENCES

[1] Himan Abdollahpouri. 2019. Popularity bias in ranking and recommendation. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. Association for Computing Machinery, New York, NY, USA, 529–530.

[2] Himan Abdollahpouri. 2020. *Popularity Bias in Recommendation: A Multi-stakeholder Perspective*. Ph.D. Dissertation. University of Colorado at Boulder.

[3] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2019. The unfairness of popularity bias in recommendation. In *RecSys Workshop on Recommendation in Multistakeholder Environments (RMSE '19)*. https://ceur-ws.org/Vol-2440/paper4.pdf RecSys Workshop on Recommendation in Multistakeholder Environments (RMSE); Conference date: 20-09-2019.
[4] Chris Anderson. 2006. *The Long Tail: Why the Future of Business is Selling Less of More*. Hachette UK.
[5] Joeran Beel, Corinna Breitinger, Stefan Langer, Andreas Lommatzsch, and Bela Gipp. 2016. Towards reproducibility in recommender-systems research. *User Modeling and User-adapted Interaction* 26 (2016), 69–101.
[6] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical biases in information retrieval metrics for recommender systems. *Information Retrieval Journal* 20, 6 (2017), 606–634.
[7] Rodrigo Borges and Kostas Stefanidis. 2021. On mitigating popularity bias in recommendations via variational autoencoders. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. 1383–1389.
[8] Òscar Celma and Pedro Cano. 2008. From hits to niches? Or how popular artists can bias music recommendation and discovery. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition (NETFLIX '08)*. Association for Computing Machinery, New York, NY, USA, Article 5, 8 pages. https://doi.org/10.1145/1722149.1722154
[9] Giovanni Luca Ciampaglia, Azadeh Nematzadeh, Filippo Menczer, and Alessandro Flammini. 2018. How algorithmic popularity bias hinders or promotes quality. *Scientific Reports* 8, 1 (2018), 15951.
[10] Israel Cohen, Yiteng Huang, Jingdong Chen, and Jacob Benesty . 2009. Pearson correlation coefficient. *Noise Reduction in Speech Processing* (2009), 1–4.
[11] Paolo Cremonesi and Dietmar Jannach. 2021. Progress in recommender systems research: Crisis? What crisis? *AI Magazine* 42, 3 (2021), 43–54.
[12] Michael D. Ekstrand, Mucun Tian, Ion Madrazo Azpiazu, Jennifer D. Ekstrand, Oghenemaro Anuyah, David McNeill, and Maria Soledad Pera. 2018. All the cool kids, how do they fit in?: Popularity and demographic biases in recommender evaluation and effectiveness. In *Conference on Fairness, Accountability and Transparency*. PMLR, 172–186.
[13] Mehdi Elahi, Danial Khosh Kholgh, Mohammad Sina Kiarostami, Sorush Saghari, Shiva Parsa Rad, and Marko Tkalčič. 2021. Investigating the impact of recommender systems on user-based and item-based popularity bias. *Information Processing & Management* 58, 5 (2021), 102655.
[14] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 101–109.
[15] Asela Gunawardana, Guy Shani, and Sivan Yogev. 2022. Evaluating recommender systems. In *Recommender Systems Handbook*. Springer, 547–601.
[16] Odd Erik Gundersen, Yolanda Gil, and David W. Aha. 2018. On reproducible AI: Towards reproducible research, open science, and digital scholarship in AI publications. *AI Magazine* 39, 3 (2018), 56–68.
[17] Odd Erik Gundersen and Sigbjørn Kjensmo. 2018. State of the art: Reproducibility in artificial intelligence. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.
[18] Benjamin Haibe-Kains, George Alexandru Adam, Ahmed Hosny, Farnoosh Khodakarami, Massive Analysis Quality Control (MAQC) Society Board of Directors, Levi Waldron, Bo Wang, Chris McIntosh, Anna Goldenberg, Anshul Kundaje, Casey S. Greene, Tamara Broderick, Michael M. Hoffman, Jeffrey T. Leek, Keegan Korthauer, Wolfgang Huber, Alvis Brazma, Joelle Pineau, Robert Tibshirani, Trevor Hastie, John P. A. Ioannidis, John Quackenbush, and Hugo J. W. L. Aerts. 2020. Transparency and reproducibility in artificial intelligence. *Nature* 586, 7829 (2020), E14–E16.
[19] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (tiis)* 5, 4 (2015), 1–19.
[20] Lei Hou, Xue Pan, and Kecheng Liu. 2018. Balancing the popularity bias of object similarities for personalised recommendation. *The European Physical Journal B* 91 (2018), 1–7.
[21] Nicolas Hug. 2020. Surprise: A Python library for recommender systems. *Journal of Open Source Software* 5, 52 (2020), 2174. https://doi.org/10.21105/joss.02174
[22] Matthew Hutson. 2018. Artificial intelligence faces reproducibility crisis. *Science* 359, 6377 (2018), 725–726. https://doi.org/10.1126/science.359.6377.725 arXiv:https://www.science.org/doi/pdf/10.1126/science.359.6377.725
[23] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.
[24] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2014. Correcting popularity bias by enhancing recommendation neutrality. *RecSys Posters* 805 (2014).
[25] Dominik Kowald and Emanuel Lacic. 2022. Popularity bias in collaborative filtering-based multimedia recommender systems. In *Advances in Bias and Fairness in Information Retrieval: Third International Workshop, BIAS 2022, Stavanger, Norway, April 10, 2022, Revised Selected Papers*. Springer, 1–11.

[26] Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2020. The unfairness of popularity bias in music recommendation: A reproducibility study. In *Advances in Information Retrieval*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer International Publishing, Cham, 35–42.
[27] Mohammadmehdi Naghiaei, Hossein A. Rahmani, and Mahdi Dehghan. 2022. The Unfairness of Popularity Bias in Book Recommendation. (2022). arXiv:cs.IR/2202.13446
[28] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence dâĂŹAlché Buc, Emily Fox, and Hugo Larochelle. 2021. Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program. *Journal of Machine Learning Research* 22 (2021).
[29] Alan Said and Alejandro Bellogín. 2014. Comparative recommender system evaluation: Benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender systems*. 129–136.
[30] Aghiles Salah, Quoc-Tuan Truong, and Hady W. Lauw. 2020. Cornac: A comparative framework for multimodal recommender systems. *Journal of Machine Learning Research* 21, 95 (2020), 1–5. http://jmlr.org/papers/v21/19-805.html
[31] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. Collaborative filtering recommender systems. In *The Adaptive Web*. Springer, 291–324.
[32] Markus Schedl. 2016. The LFM-1b dataset for music retrieval and recommendation. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. 103–110.
[33] Nasim Sonboli, Masoud Mansoury, Ziyue Guo, Shreyas Kadekodi, Weiwen Liu, Zijun Liu, Andrew Schwartz, and Robin Burke. 2021. Librec-Auto: A tool for recommender systems experimentation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 4584âĂŞ4593. https://doi.org/10.1145/3459637.3482006
[34] Catherine Stinson. 2022. Algorithms are not neutral. *AI and Ethics* (2022), 1–8.
[35] Emre Yalcin and Alper Bilge. 2021. Investigating and counteracting popularity bias in group recommendations. *Information Processing & Management* 58, 5 (2021), 102608.
[36] Li Yang and Abdallah Shami. 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* 415 (2020), 295–316.
[37] Zihao Zhao, Jiawei Chen, Sheng Zhou, Xiangnan He, Xuezhi Cao, Fuzheng Zhang, and Wei Wu. 2021. Popularity bias is not always evil: Disentangling benign and harmful bias for recommendation. *CoRR arXiv preprint arXiv:2109.07946* abs/2109.07946 (2021). https://arxiv.org/abs/2109.07946
[38] Ziwei Zhu, Yun He, Xing Zhao, and James Caverlee. 2021. Popularity bias in dynamic recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2439–2449.
[39] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web*. 22–32.

Received 30 November 2022; revised 5 November 2023; accepted 2 December 2023