Multiobjective Hyperparameter Optimization of Recommender Systems

Marta Moscati[1,*], Yashar Deldjoo[2], Giulio Davide Carparelli[2,3] and Markus Schedl[1,4]

[1]Institute of Computational Perception, Johannes Kepler University Linz, Altenberger Straße 69, Linz, 4040, Austria
[2]Polytechnic University of Bari, Bari, Italy
[3]Fiscozen, Via Venti Settembre 27, Milan, 20123, Italy
[4]Human-centered AI Group, AI Lab, Linz Institute of Technology, Altenberger Straße 69, Linz, 4040, Austria

Abstract

The quality of recommendations can be evaluated in terms of accuracy and beyond-accuracy metrics; this renders recommendation a multiobjective task. Several works apply multiobjective optimization techniques for training recommender systems (RSs) or for late fusion of recommendations. However, for the hyperparameter selection, only accuracy is considered. In this paper, we include metrics for accuracy, coverage, novelty, and fairness of recommendations towards groups of users of different activity, and items of different popularity, in the hyperparameter optimization of RSs. We apply the concept of Pareto dominance to select the optimal hyperparameter configurations. Then, by performing multiple univariate linear regressions of the values of beyond-accuracy metrics on the values of NDCG for the optimal hyperparameter configurations, we quantify the interplay of accuracy and beyond-accuracy metrics in terms of the the slope of the lines of best fit. Furthermore, by performing experiments in the domains of movie rating, music streaming, and food and household delivery and with four recommendation algorithms we provide insight in the generalizability of the interplay between accuracy and beyond-accuracy metrics. Our analysis shows that for 8 out of 12 combinations of algorithms and domains, the line of best fit for at least one beyond-accuracy metric has a negative slope, indicating a trade-off relationship and supporting the multiobjective hyperparameter optimization. Our analysis further shows that both the sign and the absolute value of the slope of the line of best fit depend on the recommendation algorithm as well as the recommendation domain, indicating the non-generalizability of the interplay between accuracy and beyond-accuracy metrics in the hyperparameter optimization.

Keywords

Recommender Systems, Evaluation, Hyperparameter Optimization, Multiobjective, Domain Generalization

1. Introduction

In the last decades, recommender systems (RSs) have become ubiquitous in our daily decisions due to the widespread of online services such as streaming and e-commerce platforms. RSs can be defined as software solutions that provide content consumers, i.e., users, convenient access to relevant content [1], allowing them to overcome the information overload they often face. In the RS research community, the accuracy of recommendations, i.e., the amount to which

Perspectives on the Evaluation of Recommender Systems Workshop (PERSPECTIVES 2023), September 19th, 2023, co-located with the 17th ACM Conference on Recommender Systems, Singapore, Singapore.

*Corresponding author.
marta.moscati@jku.at (M. Moscati); yashar.deldjoo@poliba.it (Y. Deldjoo); giulio.davide.97@gmail.com (G. D. Carparelli); markus.schedl@jku.at (M. Schedl)
0000-0002-5541-4919 (M. Moscati); 0000-0002-6767-358X (Y. Deldjoo); 0000-0003-1706-3406 (M. Schedl)

the items recommended to a user match the user's historic preferences, is most commonly used as the criterion for designing the loss functions. Additionally, metrics related to accuracy, such as precision, recall, mean average precision, and normalized discounted cumulative gain (NDCG), are commonly adopted for comparing the performance of RSs and for selecting the hyperparameter configurations of the models. However, there is now a nearly unanimous consensus within the RS community that accuracy alone is insufficient to capture the quality of recommendations [2, 3]. This is the case both since users may have multiple decision criteria [4, 1], and since users are not the only stakeholder of RSs [5, 6, 7]. Therefore, the optimization of RSs should encode the quality of recommendations in terms of accuracy, as well as beyond-accuracy metrics. This holds both for training, i.e., automatically adapting the parameters of model-based recommendation algorithms, and for *hyperparameter optimization*, i.e., selecting values for variables that are not adjusted during training, such as the embedding dimension of models based on latent representations, or the number of hidden layers and nodes of models based on neural networks.

In this work, we focus on the multiobjective hyperparameter optimization of recommendation algorithms. More concretely, we measure the quality of recommendations in terms of NDCG@10, coverage, novelty, and two metrics for the fairness of recommendations: Fairness of exposure of items of different popularity and fairness of relevance of recommendations for users of different activity. Since some of these aspects of recommendation quality can be competing with each other [4, 1], the existance of a hyperparameter configuration optimizing all the evaluation metrics is not guaranteed. This might be the case, for instance, on a music streaming platform, in which a hyperparameter configuration that achieves a high accuracy by recommending tracks that are similar to those already listened to by the user, might therefore achieve a low performance in terms of novelty.

The originality of our contribution consists in applying the concept of Pareto optimality to the hyperparameter optimization of RSs; our aim is twofold. First, to provide a deeper understanding of the interplay between accuracy and beyond-accuracy metrics. To this purpose, when analyzing the performance of a RS, instead of selecting one hyperparameter configuration, we represent the RS with the set of hyperparameter configurations that are Pareto-optimal w.r.t. the metrics under consideration, i.e., no other configuration provides a better performance in terms of one of the metrics without worsening the performance with respect to at least one of the other metrics. For each beyond-accuracy metric, we then perform a univariate linear regression on NDCG@10 of the values achieved by the hyperparameter configurations belonging to the Pareto front, in order to quantify the interplay of the two metrics for the optimal configurations. Second, we provide an analysis of the generalizability of the trade-offs among the optimization metrics across algorithms and domains of recommendation. More specifically, we consider four recommendation algorithms – `ItemkNN` [8], `BPR` [9], `MultVAE` [10], and `LightGCN` [11] – and three datasets from the domains of movie rating, music streaming, and food and household delivery. This allows us to compare the relationships between metrics across combinations of algorithms and domains. Our analysis shows that accuracy and at least one of the beyond-accuracy metrics taken into consideration are in a trade-off relationship in 8 out of 12 combinations of algorithms and domains. We hence highlight the potential harms of a hyperparameter optimization based only on accuracy and provide insight for the multiobjective hyperparameter optimization of RSs. In addition, we show that the relationship

between accuracy and beyond-accuracy metrics strongly depends on the algorithm and on the domain of recommendation and hence it is not generalizable. Throughout our analysis, we make use of the library `recsyslearn`[1], which we introduce in this work for the first time. `recsyslearn` is an open-source library for the evaluation of recommendation lists. The library does not rely on any external RS framework, therefore facilitating the comparison of model performance and the reproducibility of experiments. For more details on the functionalities of `recsyslearn` we refer the reader to Appendix B, to the official documentation,[2] and to the GitHub repository.[3]

In summary, this work assumes a novel perspective on the hyperparameter optimization of RSs by treating it as a multiobjective optimization task and applying the concept of Pareto optimality. We provide insight on the interplay between accuracy and beyond-accuracy metrics on the optimal hyperparameter configurations, highlighting the potential harms of a hyperparameter optimization based only on accuracy. We further show that the relationship between accuracy and beyond-accuracy metrics is not generalizable across algorithms and domains of recommendation.

2. Related Work

Evaluating RSs beyond accuracy is motivated by the observation that accuracy is not the only aspect defining the quality of recommendations [2]. Additionally, content consumers are not the only stakeholders affected by RSs [6, 7]. These considerations led to the design of several evaluation metrics going beyond accuracy [12], such as metrics measuring the novelty and diversity [13] or the popularity bias [14] of recommendations, and to the development of multiobjective RSs, i.e., RSs designed to optimize or balance more than one optimization objective [1]. For instance, Zhou et al. [15] use a late fusion approach to optimize diversity and accuracy of recommendations simultaneously, while Zhang et al. [16] and Jambor et al. [17] propose the use of hard constraints on the values of one or more optimization objectives. More recently, other works proposed scalarization for the multiobjective optimization of RSs. This approach consists in modeling each metric of recommendation with a loss term and compute the total loss as a weighted sum of the terms related to each metric. For instance, Coba et al. [18] propose a matrix factorization approach with a loss term for the novelty of recommendations. Moreira et al. [19] add regularization terms that model popularity and recency in the loss function of recurrent neural networks for session-based news recommendation. Isufi et al. [20] propose a RS based on graph convolutions that includes a regularization term for increasing the diversity of recommendations. Few works exploit the concept of Pareto optimality, which is fundamental in the domain of multiobjective optimization, for balancing the accuracy and beyond-accuracy aspects of recommendations. In the context of RSs, a model is Pareto-optimal if none of the metrics can be improved without negatively affecting at least one of the others. In addition to the metrics to be optimized, the current approaches differ in what is considered as independent variables the metrics depend on (e.g., parameters of the model, late fusion parameters, or weights in the scalarization of the multiobjectives), and on the strategy used to approximate the Pareto front, i.e., either with evolutionary algorithms or by means of scalarization. For instance, Ribeiro

[1] https://recsyslearn.readthedocs.io/en/latest/
[2] recsyslearn.readthedocs.io
[3] github.com/giuliowaitforitdavide/recsyslearn/

et al. [21] leverage the concept of Pareto optimality for the late fusion of RSs in a post-processing fashion using evolutionary algorithms. Other authors apply the concept of Pareto optimality to in-processing techniques based on scalarization, i.e., combining the objectives in a single value. For instance, Lin et al. [22] propose a two-step algorithm for automatically adjusting the scalarization weights and the model parameters simultaneously, in a way such that the scalarization weights converge to Pareto-optimal solutions. Similarly, Wu et al. [23] propose a framework in which fairness metrics are translated to loss terms with the use of smoothed versions of the ranked lists. The weights of the corresponding terms in the loss function are adapted during training in a way that guarantees Pareto optimality. Other works leverage reinforcement learning to approximate the Pareto-optimal solutions. For instance, the approach proposed by Ge et al. [24] generates the Pareto front in the space of accuracy and fairness of recommendations with a single training run by leveraging multiobjective reinforcement learning. Xie et al. [25] propose a multiobjective RS based on a set of single-objective models and a reinforcement learning module that tailors the weights of the loss terms reflecting the metrics to the profile of each user. The module leverages the Pareto stationarity to approximate Pareto-optimal combinations of weights. In the context of group recommendation, Xiao et al. [26] aim at balancing the group utility and the fairness of group recommendation, the latter being defined in terms of the differences between the utilities of the single users. For this purpose, the authors propose two approaches to generate group recommendations that approximate the Pareto front in terms of group utility and fairness. One algorithm is based on a greedy approach, while the other is based on integer programming.

The approaches discussed above address the issue of adapting the parameters of the model either with evolutionary algorithms or with loss functions designed by means of scalarization, which hence allow the use of training techniques such as stochastic gradient descent. However RSs only yield optimal performance when also the hyperparameters are properly tuned, and their tuning cannot be included in the training procedure. For instance, the hyperparameters defining the architecture of a neural-network-based recommendation algorithm affect the recommendation accuracy, but the architecture has to be set up prior to training. While the multiobjective optimization of the hyperparameters is a vivid topic of research in machine learning in general (see, for instance, [27, 28] for recent reviews), the hyperparameter optimization of RSs is typically performed considering accuracy aspects, only. To the best of our knowledge, only Quadrana et al. [29] model the *hyperparameter* optimization of RSs as a multiobjective optimization task, while the current work is the first that also leverages the concept of Pareto front, instead of using scalarization to reduce it to a single objective optimization task.

3. Methodology

In this section, we describe the methodology of our work. Section 3.1 introduces the formalism and notation used throughout the paper and adapts the concept of Pareto optimal and Pareto front to the hyperparameter optimization. In Section 3.2 we describe the datasets used in our experiments, as well as the data preparation. Section 3.3 provides a description of the recommendation algorithms used in our analysis, while Section 3.4 introduces the reader to the mathematical formulation of the accuracy and beyond-accuracy metrics we consider in this work. We then describe how we carried out the analysis of the interplay between the optimization objectives by means of univariate linear regression in Section 3.5.

Table 1
Summary of the notation used in the paper.

Notation	Description
$A = \{a_i\}_{i=1}^{N_a}$	Set of recommendation algorithms.
$M = \{m_j\}_{j=1}^{N_m}$	Set of evaluation metrics.
$H_i = \{h_c^i\}_{c=1}^{G_i}$	Set of hyperparameter configurations considered for algorithm a_i.
$P_i \subseteq H_i$	Subset of Pareto optimal configurations of a specific algorithm a_i.
IF	Fairness of exposure for items of different popularity.
UF	Fairness of effectiveness of recommendations for items of different activity.

3.1. Optimal Configurations

In this section we introduce the reader to the notation used throughout the paper, providing a summary in Table 1. We indicate the set of N_a recommendation algorithms (e.g., matrix factorization, item k-nearest neighbors, ...) as $A = \{a_i\}_{i=1}^{N_a}$ and the set of N_m evaluation metrics as $M = \{m_j\}_{j=1}^{N_m}$. For each algorithm $a_i \in A$, we consider a set of G_i hyperparameter configurations, $H_i = \{h_c^i\}_{c=1}^{G_i}$. The specific configurations h_c^i as well as the total number of configurations considered G_i depend on the algorithm a_i considered. A hyperparameter configuration $h_1^i \in H_i$ of a specific algorithm a_i *dominates* another configuration $h_2^i \in H_i$ if the values of all metrics $m_j \in M$ are equal or better[4] on h_1^i than on h_2^i, and if there exists at least one metric that is strictly better on h_1^i than on h_2^i. A model is considered *Pareto optimal* on the considered set H_i if it is not dominated by any other configuration in H_i. We indicate the subset of Pareto optimal configurations of a specific algorithm as $P_i \subseteq H_i$; this subset consists of one hyperparameter configuration if the objectives are not competing, and coincides with H_i if none of the configurations considered is dominated by the others.

For a given dataset and recommendation algorithm a_i, our goal is to investigate the interplay between pairs of evaluation objectives when the hyperparameters are varied. Therefore, to ensure that the hyperparameter configurations selected are the ones providing the best trade-off, we restrict our analysis to the configurations in P_i. In agreement with the typical (single-objective) hyperparameter optimization, the set of optimal hyperparameter configurations P_i is selected on the validation set, while the overall performance is computed and reported on the test set.

3.2. Datasets

We perform our experiments in three recommendation domains and on three corresponding datasets $d_k \in D$, namely movie (`MovieLens100K` [30]), e-commerce (`Amazon Pantry`[5]), and music (`LastFM` [31]). `MovieLens100K` and `Amazon Pantry` contain explicit feedback from users, given as item ratings, while `LastFM` contains implicit feedback, consisting of listening events, which indirectly reflect users' preferences. All datasets are converted to implicit and binarized feedbacks. For the explicit datasets (`MovieLens100K`, `Amazon Pantry`) we consider interactions to be positive if the rating is at least 3. For `LastFM`, we consider as positive

[4]We use the term *better* instead of *higher* since for some metrics, lower values are preferred.
[5]https://jmcauley.ucsd.edu/data/amazon/

Table 2
Characteristics of the datasets used in the experiments. Values refer to the datasets after conversion to implicit feedback and after user 5-core filtering. The Gini indices are defined as in Deldjoo [32].

Dataset	Domain	Interactions	Users	Items	Gini Items	Gini Users
MovieLens100K	Movie rating	82 520	943	1 574	0.64	0.47
LastFM	Music streaming	92 798	1 877	17 617	0.73	0.01
Amazon Pantry	Food and household delivery	115 920	11 981	8 401	0.67	0.33

interactions those user-item pairs for which the listening count is at least 2. After binarization, we apply 5-core filtering to users, restricting the dataset to the set of users that have positive interactions with at least 5 different items. The characteristics of the datasets after binarization and filtering are summarized in Table 2. The resulting dataset is split into a train, a validation, and a test set by randomly selecting 60%, 20%, and 20% of the positive interactions of each user. This ensures all users to be present in all the three sets. The selection of the non-dominated hyperparameter configurations P_i is performed on the validation, while the results refer to the test set.

3.3. Algorithms

We select for our study four types of recommendation algorithms: memory-based, matrix-factorization-based, neural-network-based, and graph-based algorithms. The representatives of each algorithm class are selected due to their frequent use in the RS community. Following the work of Melchiorre et al. [33], we select Item-k-nearest-neighbors (ItemkNN [8]) for memory-based algorithms, matrix factorization with Bayesian Personalized Ranking as optimization function (BPR [9]) for matrix-factorization-based algorithms, and variational autoencoders for collaborative filtering (MultVAE [10]) for neural-network-based algorithms. For graph-based approaches we select LightGCN [11]. We provide below a brief description of the investigated algorithms. The set of hyperparameter configurations H_i considered for each algorithm a_i is a grid obtained by linearly spacing the range of values considered for each hyperparameter. Details on the values considered for each hyperparameter can be found at https://github.com/mmosc/moho/.

ItemkNN [8] is a memory-based approach that recommends to a user items that are similar to the ones they interacted with. The similarity between items is defined on the consumption pattern of other users, and computed as cosine similarity between the interaction vectors of items (i.e., the corresponding columns in the user-item interaction matrix).

BPR [9] BPR is an optimization function that ranks the items by defining an implicit order between pairs thereof, and maximizing the difference between the scores of items that have been interacted with by the user, with respect to the other items. Rendle et al. [9] introduce BPR and apply it to matrix factorization. We use the same strategy and hence indicate with BPR a matrix factorization algorithm with BPR as optimization strategy.

MultVAE [10] is a variational autoencoder architecture that projects the users' interaction vectors to a latent distribution space. The model then samples the user's latent representation from the corresponding latent distribution and reconstructs the interaction vector by means of a decoder; the model is trained to minimize the reconstruction loss.

LightGCN [11] is a graph convolutional neural network (GCN) that models users and items as entities in a graph, and interactions as edges. The model learns user and item embeddings by linearly propagating them on the user-item interaction graph. The weighted sum of the embeddings of the neighbors of an entity contributes to the final embedding of the entity. Items are then assigned a recommendation score given by the scalar product between the embedding of the item and the embedding of the target user.

3.4. Evaluation Metrics

We consider accuracy, coverage, novelty, as well as user and item fairness as metrics for evaluating the quality of recommendations. In this section, we provide the mathematical formulation of these evaluation metrics. In our analysis, all metrics are computed for recommendation lists of $k = 10$ items.

Accuracy: We measure the accuracy of recommendations for a user u in terms of NDCG. For a list of k recommendations, $\text{NDCG@}k(u) = \frac{\text{DCG@}k(u)}{\text{IDCG@}k(u)}$ [13], where DCG is defined as $\text{DCG@}k(u) = \sum_{i=1}^{k} \frac{\text{rel}(i)}{\log_2(i+1)}$. Since we consider a scenario of implicit feedback, $\text{rel}(i)$ is an indicator function that signals whether the item recommended at rank i has been positively interacted with by user u. $\text{IDCG@}k(u)$ represents the ideal DCG for user u, obtained when the intersection between the set of recommended items and the items with which the user has positively interacted is the largest possible allowed by the user's profile. If u positively interacted with a sufficient number of items, IDCG corresponds to the value of DCG obtained when u positively interacted with all items in their recommendation list. We choose NDCG as a measure of accuracy since, compared to other metrics, it not only captures the ability of the system to recommend relevant items, but also gives more weight to accurate recommendations that appear higher in the recommendation list.

Coverage: We investigate coverage, which is defined as the proportion of items that appear in the recommendation lists at least once, $\text{Cov} = \frac{|\hat{T}|}{|T|} = \frac{\left|\bigcup_u \hat{T}_u\right|}{|T|}$ [13], where T is the set of items in the dataset, $\hat{T}_u$ is the set of items recommended to user u, while $\hat{T}$ the set of items that have been recommended to at least one user. A low coverage is hence an indication that the RS tends to always recommend a restricted set of items.

Novelty: To reflect the *freshness* of the recommended items in terms of global popularity we define novelty as $N@k = -\frac{1}{|U|}\sum_{u\in U}\sum_{i\in\hat{T}_u}\frac{\log_2 \text{pop}_i}{k}$ [13], where $\hat{T}_u$ is the list of k items recommended to user u, and pop_i is the popularity of item i, measured as percentage of interactions with i compared to the total number of interactions. Since we perform our experiments on binarized versions of the datasets, this definition of popularity corresponds to the percentage of unique positive user-item pairs involving i, compared to the total number of positive pairs.

Item Fairness (IF): We consider an *interaction-oriented* definition of item fairness [5] and therefore measure the extent to which a RS is able to spread item exposure across items of different popularity. Exposure is defined in terms of the appearances in the recommendation lists, while items are categorized according to their popularity. Following the work of Lesota et al. [34], we define three item categories: a *short-head*, a *mid-tail*, and a *distant-tail*, corresponding to the most popular, intermediate, and least popular items, respectively. These categories are defined in terms of percentiles of the total interactions in the dataset; *short-head*, *mid-tail*, and *distant-tail* account for 60%, 30%, and 10% of the number of interactions in the training set, respectively.

We then compute the distribution E of recommendations across popularity categories; $E(c)$ is therefore the proportion of items of popularity category c in the recommendations (i.e., over all users). Since we assume proportionality as a fairness criterion, we take the Kullback-Leibler (KL) divergence w.r.t. a target distribution Tar_i reflecting the distribution of the interactions in the training set across the same categories, as a measure of item fairness. Item fairness is therefore defined as $\text{IF} = \text{KL}(E, \text{Tar}_i) = \sum_c E(c) \log_2 \frac{E(c)}{\text{Tar}_i(c)}$.

User Fairness (UF): We are interested in a behavior-oriented definition of user fairness [5, 35]. Therefore we consider the extent to which effective recommendations are spread over user groups of different activity. We first assign each user to the group of *active*, *semi-active*, or *inactive* users, respectively representing the 60%, 30%, and 10% percentiles of the total number of interactions. We then compute the distribution R of *relevant* recommendations across activity groups and measure the overall fairness of recommendations as the KL divergence w.r.t. a target distribution Tar_u reflecting the distribution of the interactions in the training set across the same groups. User fairness is therefore defined as $\text{UF} = \text{KL}(R, \text{Tar}_u) = \sum_g R(g) \log_2 \frac{R(g)}{\text{Tar}_u(g)}$. We use the library `recsyslearn` to compute the metrics and to segment users and items into the categories and groups required by the definition of IF and UF.

3.5. Quantifying the interplay between metrics

We aim at modeling the variation in performance of a RS when its hyperparameters are varied across the optimal configurations. Since we consider four recommendation algorithms, three domains of recommendation, and five metrics for evaluation, analyzing the interplay between each pair of metrics would lead to 10 comparisons for each algorithm-dataset combination, for a total of 120 pair-wise analyses. Therefore, since accuracy is still the dominant optimization objective in the RS community, we simplify the discussion by considering the pair-wise interplay of NDCG@10 with each of the beyond-accuracy metrics described in Section 3.4. Furthermore, similar pair-wise analyses between beyond-accuracy metrics can be carried out by adapting the code we provide under https://github.com/mmosc/moho/.

For each algorithm and each domain of recommendation, the interplay between accuracy and each of the beyond-accuracy metrics is modeled as a linear dependence by means of a univariate linear regression: accuracy is taken as the independent variable x and each of the beyond-accuracy metrics as the dependent variable y. As it is well known, and in agreement with previous studies that analyze the impact of sampling strategies and dataset characteristics on the performance of RSs [36], we observe that depending on the domain of recommendation the values of the metrics range in very different intervals. This general and well-established observation is evident in our experiments by looking at the mean values of the metrics reported in Table 3, computed for each algorithm a_i and each dataset d_k, over the optimal hyperparameter configurations in P_i. We are, however, not primarily interested in the absolute performance of each recommendation algorithm, but rather in comparing the different amounts of trade-offs that each of them displays among the optimization metrics, and on whether these trade-offs depend on the domain of recommendation. Therefore, in order to facilitate the comparison across domains despite the different absolute values of the metrics, we first apply min-max scaling to the values achieved for each metric on the test set of a specific recommendation domain, and across all model configurations a_i in the optimal configurations P_i. In the case of IF and UF, lower values are to be preferred since they indicate a higher similarity between the

target distribution (i.e., the one observed over the interactions in the training set) and the one observed over recommendations. We therefore subtract the min-max rescaled values of IF and UF from 1 before performing the linear regression. In this way, also for the normalized values of IF and UF higher values of the slope of the line of best fit indicate a better interplay, while negative values indicate a trade-off relationship.

To summarize, we quantify the interplay between different metrics in the hyperparameter selection by modeling the dependence of each of the beyond-accuracy metric on NDCG@10 as a linear relationship. We perform a univariate linear regression on the pairs of min-max rescaled values (NDCG@10, m_j), with $m_j \in M \setminus \{\text{NDCG@10}\}$. We then take the slope of the line of best fit as a measure of the overall trend for the trade-off for a specific recommendation algorithm on a specific dataset. A steeper positive slope of the line indicates a better overall trade-off for that recommendation algorithm on that dataset.[6] To better visualize our approach, we display the values of the min-max scaled metrics, as well as the line of best fit, in 2-dimensional planes with NDCG@10 on the x and $m_j \in M \setminus \{\text{NDCG@10}\}$ on the y axis. Each recommendation algorithm a_i is represented by a different color, and points represent different optimal hyperparameter configurations $h_c^i \in P_i$. Figure 1 provides an example of these plots on the `Amazon Pantry` dataset. Similar plots for the other domains can be found in Appendix A and obtained by adapting the code available at https://github.com/mmosc/moho/. These plots represent visually our way of modeling the dependency of a beyond-accuracy metric on NDCG: If moving from left to right, i.e., increasing NDCG@10, the value of the beyond-accuracy metric depicted on the y-axis tends to decrease, this is an indication of a trade-off between accuracy and the beyond-accuracy metric represented on the y axis.

4. Results

Table 3 displays the average value of each evaluation metric before min-max rescaling, as well as the slope of the line of best fit between the min-max rescaled values of NDCG@10 and each of the beyond-accuracy metrics $m_j \in M \setminus \{\text{NDCG@10}\}$, for each dataset $d_k \in D$ and recommendation algorithm $a_m \in A$. By looking at the slopes for each recommendation algorithm and across all datasets, we notice that none of the algorithms $a_m \in A$ displays a positive interplay (slope) for all $m_j \in M \setminus \{\text{NDCG@10}\}$ and across all $d_k \in D$. Even considering all algorithm-dataset combinations (a_m, d_k), only for a few of them, highlighted in green, all beyond-accuracy metrics have a positive interplay with accuracy. This is an indication that for many combinations, NDCG@10 and at least one of the beyond-accuracy metrics $m_j \in M \setminus \{\text{NDCG@10}\}$ show a trade-off relationship: as NDCG@10 increases, m_j tends to decrease, indicating a worse performance of the model w.r.t. this aspect of recommendation.

Insights: In the hyperparameter optimization of RSs, accuracy and beyond-accuracy metrics do not always show a positive interplay, indicating a trade-off relationship. This highlights the need for multiobjective hyperparameter optimization to balance competing objectives, since in general there exists no optimal hyperparameter configuration for all evaluation metrics.

[6] We focus on the slope instead of the Pearson's correlation coefficient since the slope represents the (average) change in the metric for each percentage increase in NDCG@10, and hence better captures the trade-offs we want to analyze.

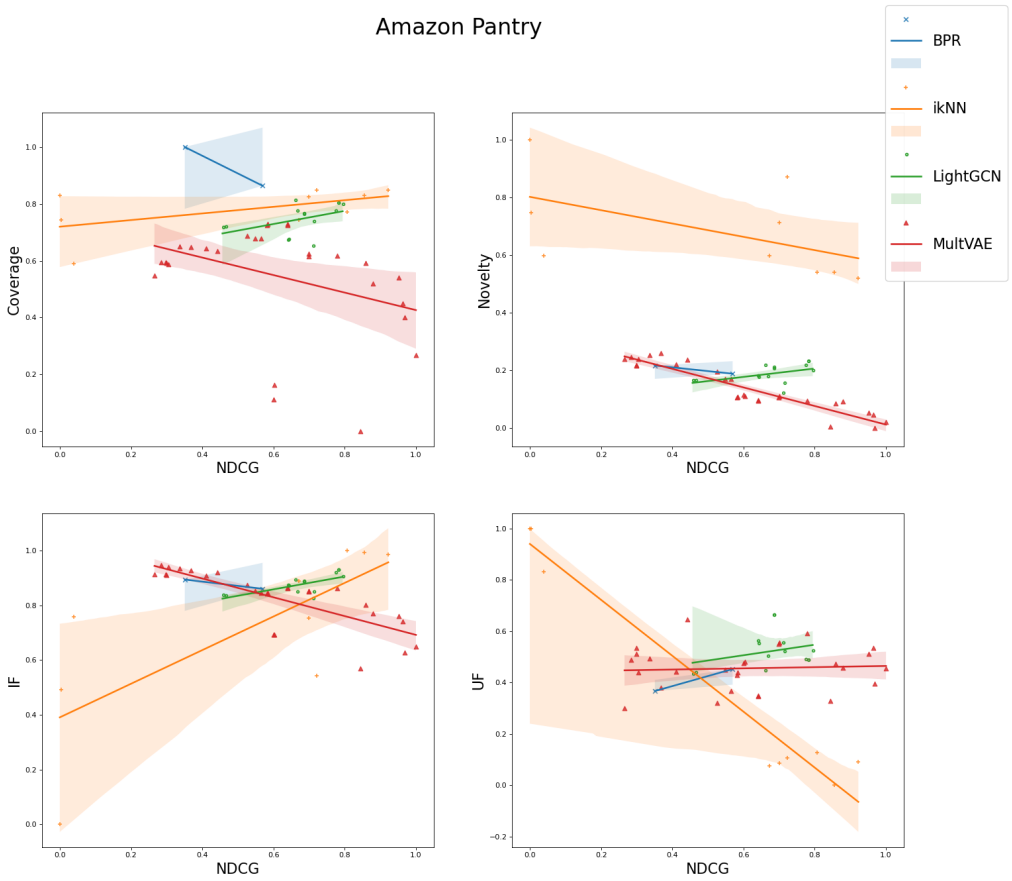

Figure 1: Performance of the Pareto optimal configurations P_i in terms of accuracy (x axis) and beyond-accuracy (y axis) metrics on the test set of `Amazon Pantry`. All metrics are min-max scaled. Each color represents a different recommendation algorithm. Shaded areas correspond to the 1σ confidence intervals of the linear regression.

Additionally, in agreement with previous studies [36], we observe that the ranking of different recommendation algorithms according to their performance in terms of accuracy of recommendation varies across datasets. In other words, not only the absolute values of NDCG@10 for a specific recommendation algorithm, but also the relative performance compared to other algorithms depends on the characteristics of the dataset. The novelty of our analysis is in showing that these considerations hold for the interplay between accuracy and beyond-accuracy metrics: Comparing the slopes of the lines of best fit for a single algorithm across different datasets, we observe that the interplay between accuracy and other performance metrics heavily depends on the domain of recommendation. Additionally, if the interplay between NDCG@10 and $m_j \in M \setminus \{\text{NDCG@10}\}$ on a dataset $d_k \in D$ is better for algorithm $a_m \in A$ than for algorithm $a_n \in A$, the same does not necesarily hold on a different dataset $d_l \in D \setminus \{d_k\}$.

Table 3
Mean (before min-max rescaling) and slope of the line of best fit for the dependence of the min-max scaled metrics $m_j \in M \setminus \{\text{NDCG@10}\}$ on NDCG@10. Green cells indicate a model-dataset combination for which all slopes are positive. The best interplays between NDCG@10 and m_j are highlighted in **bold**, the worst in *italic*.

Dataset	Model		NDCG@10	Coverage	Novelty	IF	UF
LastFM	BPR	mean	0.169	0.190	9.588	0.508	2.726e-04
		slope	—	**0.923**	**0.285**	0.998	1.973e-01
	ItemkNN	mean	0.156	0.269	11.192	0.220	3.251e-04
		slope	—	-1.058	-2.044	*-1.577*	1.055e-02
	LightGCN	mean	0.007	0.255	12.859	0.051	2.466e-03
		slope	—	*-37.666*	*-15.634*	**7.687**	**1.591e+01**
	MultVAE	mean	0.210	0.346	10.944	0.144	6.870e-04
		slope	—	-0.086	-0.458	-0.856	*-5.607e-02*
MovieLens100K	BPR	mean	0.291	0.535	8.987	0.183	1.646e-01
		slope	—	*-0.623*	-0.131	-0.154	**3.914e-01**
	ItemkNN	mean	0.252	0.697	9.385	0.087	1.711e-01
		slope	—	0.117	-0.231	**0.614**	*-1.089*
	LightGCN	mean	0.063	0.608	9.140	0.116	1.478e-01
		slope	—	**0.231**	**0.145**	0.235	2.050e-01
	MultVAE	mean	0.300	0.527	9.079	0.125	1.714e-01
		slope	—	-0.308	*-0.322*	*-0.343*	2.317e-02
Amazon Pantry	BPR	mean	0.025	0.939	10.701	0.186	7.100e-02
		slope	—	0.176	-0.151	0.107	*-8.647e-02*
	ItemkNN	mean	0.027	0.856	12.783	0.400	7.390e-02
		slope	—	*-1.287*	*-3.363*	*-3.116*	4.368e-01
	LightGCN	mean	0.032	0.837	10.638	0.187	6.307e-02
		slope	—	**4.416**	**4.416**	**10.369**	2.432e-01
	MultVAE	mean	0.030	0.727	10.424	0.250	6.776e-02
		slope	—	0.976	0.049	0.107	**5.516e-01**

As a consequence, also the algorithm achieving the best trade-off in terms of NDCG@10 and one of the beyond-accuracy metrics $m_j \in M \setminus \{\text{NDCG@10}\}$ can vary across recommendation domains. This pattern can be observed in Table 3 where for each domain and for each beyond-accuracy metric $m_j \in M \setminus \{\text{NDCG@10}\}$, the highest value of the slope of the line of best fit is highlighted in bold, while the lowest is displayed in italic.

Insights: The impact of algorithm selection on the interplay between metrics varies depending on the dataset. Along with hyperparameter selection, model selection is a process that cannot be translated nor generalized across datasets, neither in terms of individual metrics, nor in terms of their interplay.

5. Conclusions and Future Work

The hyperparameter optimization of RSs is typically regarded as a single-objective optimization problem. However, many aspects of recommendations have to be considered when

evaluating their quality, both because several stakeholders are involved, and because each stakeholder values more than one aspect of recommendations. In this paper we analyzed the interplay between NDCG@10, coverage, novelty and fairness of recommendation towards items of different popularity and users of different activity, for four algorithms and three domains of recommendation. We showed that the aspects of recommendation included in our analysis are often competing and we therefore provided evidence in support of the multiobjective hyperparameter optimization of RSs. Our analysis was carried out adapting the concepts of Pareto optimality and of Pareto front from the multiobjective optimization domain: We selected the set of optimal hyperparameter configurations and quantified the interplay between accuracy and beyond-accuracy metrics by means of univariate linear regressions. This allowed us to show that the accuracy of recommendations is often in a trade-off relationship with at least one of the beyond-accuracy metrics considered, hence supporting our proposal for multiobjective hyperparameter optimization. Furthermore, by comparing the results across algorithms and domains of recommendation, we showed that the interplay between evaluation metrics depends both on the RS and on the dataset, hence showing that the results of the evaluation of the amount of trade-off cannot be generalized across algorithms and domains. In performing our analysis we developed `recsyslearn`, which is a library for the evaluation of accuracy and beyond-accuracy aspects of recommendations. Since `recsyslearn` does not rely on any library used to generate the recommendation lists, we hope that its release can contribute to the reproducibility of scientific results in the domain of RSs.

Although in this work we evaluated the performance of RSs in terms of accuracy, coverage, novelty, and fairness of recommendations towards items of different popularity and users of different activity, the analysis can be extended to other measures as well [35]. For instance, one could consider user group fairness with respect to users of different gender, item group fairness with respect to items of different genres, or include emerging evaluation metrics such as glocalization [37]. Additionally, the dependence of the relationship between accuracy and beyond-accuracy metrics on the domain of recommendation could be attributed to the different dataset characteristics. Therefore, it would be interesting to also analyze how the interplay between evaluation metrics varies depending on dataset characteristics, such as sparsity and Gini index with respect to users and to items [32], or on sampling strategies used to reduce the size of the dataset. We also considered only a cutoff of $k = 10$ in our evaluation, although the metrics considered, and therefore also their interplay, might depend on this parameter. Our analysis relied on multiple univariate linear regressions of beyond-accuracy metrics on NDCG and did not include a multivariate linear regression of all beyond-accuracy metrics, nor the use of statistical tools such as mixed-effects models, which allow to also model groupings of datapoints (e.g., in terms of the dataset or of the underlying recommendation algorithm). We leave these extensions of our analysis for future work.

This work raises the question of which techniques from the multiobjective optimization domain, e.g., scalarization or heuristic approaches such as evolutionary algorithms, can be effectively translated to the hyperparameter optimization of RSs. This challenge still remains open and we hope it will be addressed by the RS community in the next years.

References

[1] D. Jannach, Multi-objective recommendation: Overview and challenges, in: In CEUR Proc. of MORS Workshop co-located with RecSys, volume 3268, 2022.

[2] S. M. McNee, J. Riedl, J. A. Konstan, Being accurate is not enough: How accuracy metrics have hurt recommender systems, in: CHI Extended Abstracts, 2006, pp. 1097–1101.

[3] M. Ge, C. Delgado-Battenfeld, D. Jannach, Beyond accuracy: Evaluating recommender systems by coverage and serendipity, in: Proc. of ACM RecSys, 2010, pp. 257–260.

[4] J. Zhang, G. Adomavicius, A. Gupta, W. Ketter, Consumption and performance: Understanding longitudinal dynamics of recommender systems via an agent-based simulation framework, Information System Research 31 (2020) 76–101.

[5] Y. Deldjoo, D. Jannach, A. Bellogin, A. Difonzo, D. Zanzonelli, Fairness in recommender systems: Research landscape and future directions, User Modeling and User-Adapted Interaction (2023) 1–50.

[6] H. Abdollahpouri, G. Adomavicius, R. Burke, I. Guy, D. Jannach, T. Kamishima, J. Krasnodebski, L. Pizzato, Multistakeholder recommendation: Survey and research directions, User Modeling and User-Adapted Interaction 30 (2020) 127–158.

[7] H. Abdollahpouri, R. Burke, Multistakeholder recommender systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, 2022, pp. 647–677.

[8] M. Deshpande, G. Karypis, Item-based top-n recommendation algorithms, ACM Transactions on Information Systems 22 (2004) 143–177.

[9] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proc. of UAI, 2009, pp. 452–461.

[10] D. Liang, R. G. Krishnan, M. D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proc. of ACM WWW, 2018, pp. 689–698.

[11] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: Proc. of SIGIR, 2020, pp. 639–648.

[12] A. Gunawardana, G. Shani, S. Yogev, Evaluating recommender systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, 2022, pp. 547–602.

[13] M. Schedl, H. Zamani, C. Chen, Y. Deldjoo, M. Elahi, Current challenges and visions in music recommender systems research, International Journal of Multimedia Information Retrieval 7 (2018) 95–116.

[14] O. Lesota, A. Melchiorre, N. Rekabsaz, S. Brandl, D. Kowald, E. Lex, M. Schedl, Analyzing item popularity bias of music recommender systems: Are different genders equally affected?, in: Proc. of ACM RecSys, 2021, pp. 601–606.

[15] T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, Y.-C. Zhang, Solving the apparent diversity-accuracy dilemma of recommender systems, Proceedings of the National Academy of Sciences 107 (2010) 4511–4515.

[16] M. Zhang, N. Hurley, Avoiding monotony: improving the diversity of recommendation lists, in: Proc. of ACM RecSys, 2008, pp. 123–130.

[17] T. Jambor, J. Wang, Optimizing multiple objectives in collaborative filtering, in: Proc. of ACM RecSys, 2010, pp. 55–62.

[18] L. Coba, P. Symeonidis, M. Zanker, Novelty-aware matrix factorization based on items' popularity, in: Proc. of AIxIA, volume 11298, 2018.

[19] G. de Souza Pereira Moreira, D. Jannach, A. M. da Cunha, Contextual hybrid session-based news recommendation with recurrent neural networks, IEEE Access 7 (2019) 169185–169203.
[20] E. Isufi, M. Pocchiari, A. Hanjalic, Accuracy-diversity trade-off in recommender systems via graph convolutions, Information Processing & Management 58 (2021) 102459.
[21] M. T. Ribeiro, N. Ziviani, E. S. D. Moura, I. Hata, A. Lacerda, A. Veloso, Multiobjective pareto-efficient approaches for recommender systems, ACM Transactions on Intelligent Systems and Technology 5 (2015).
[22] X. Lin, H. Chen, C. Pei, F. Sun, X. Xiao, H. Sun, Y. Zhang, W. Ou, P. Jiang, A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation, in: Proc. of ACM RecSys, RecSys '19, 2019, pp. 20–28.
[23] H. Wu, C. Ma, B. Mitra, F. Diaz, X. Liu, A multi-objective optimization framework for multi-stakeholder fairness-aware recommendation, ACM Transactions on Information Systems 41 (2022).
[24] Y. Ge, X. Zhao, L. Yu, S. Paul, D. Hu, C.-C. Hsieh, Y. Zhang, Toward pareto efficient fairness-utility trade-off in recommendation through reinforcement learning, in: Proc. of ACM WSDM, 2022, pp. 316–324.
[25] R. Xie, Y. Liu, S. Zhang, R. Wang, F. Xia, L. Lin, Personalized approximate pareto-efficient recommendation, in: Proc. of ACM WWW 2021, New York, NY, USA, 2021, pp. 3839–3849.
[26] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, M. Shaoping, Fairness-aware group recommendation with pareto-efficiency, in: Proc. of ACM RecSys, 2017, pp. 107–115.
[27] F. Karl, T. Pielok, J. Moosbauer, F. Pfisterer, S. Coors, M. Binder, L. Schneider, J. Thomas, J. Richter, M. Lang, E. C. Garrido-Merchán, J. Branke, B. Bischl, Multi-objective hyperparameter optimization - an overview, 2022.
[28] A. M. Hernández, I. V. Nieuwenhuyse, S. Rojas-Gonzalez, A survey on multi-objective hyperparameter optimization algorithms for machine learning, Artificial Intelligence Review 56 (2023) 8043–8093.
[29] M. Quadrana, A. Larreche-Mouly, M. Mauch, Multi-objective hyper-parameter optimization of behavioral song embeddings, in: Proc. of ISMIR, 2022, pp. 437–445.
[30] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, ACM Transactions on Interactive Intelligent Systems 5 (2015).
[31] I. Cantador, P. Brusilovsky, T. Kuflik, Second workshop on information heterogeneity and fusion in recommender systems, in: Proc. of ACM RecSys, 2011, p. 387–388.
[32] Y. Deldjoo, T. Di Noia, E. Di Sciascio, F. A. Merra, How dataset characteristics affect the robustness of collaborative recommendation models, in: Proc. of ACM SIGIR, 2020, pp. 951–960.
[33] A. B. Melchiorre, N. Rekabsaz, E. Parada-Cabaleiro, S. Brandl, O. Lesota, M. Schedl, Investigating gender fairness of recommendation algorithms in the music domain, Information Processing & Management 58 (2021) 102666.
[34] O. Lesota, S. Brandl, M. Wenzel, A. Melchiorre, E. Lex, N. Rekabsaz, M. Schedl, Exploring cross-group discrepancies in calibrated popularity for accuracy/fairness trade-off optimization, in: In CEUR Proc. of MORS Workshop co-located with RecSys, 2022.
[35] E. Amigó, Y. Deldjoo, S. Mizzaro, A. Bellogín, A unifying and general account of fairness measurement in recommender systems, Information Processing & Management 60 (2023)

103115.
[36] N. Sachdeva, C.-J. Wu, J. McAuley, On sampling collaborative filtering datasets, in: Proc. of ACM WSDM, 2022, pp. 842–850.
[37] O. Lesota, E. Parada-Cabaleiro, S. Brandl, E. Lex, N. Rekab-saz, M. Schedl, Traces of globalization in online music consumption patterns and results of recommendation algorithms, in: Proc. of ISMIR, 2022, pp. 291–297.

A. Additional Figures

Figure 2 displays the scatterplots of the values of the min-max scaled metrics, as well as the line of best fit, in 2-dimensional planes with NDCG@10 on the x and $m_j \in M \setminus \{\text{NDCG@10}\}$ on the y axis, for `MovieLens100K` and `LastFM`. Each recommendation algorithm a_i is represented by a different color, and points represent different optimal hyperparameter configurations $h_c^i \in P_i$. These plots extend Figure 1, which refer to the `Amazon Pantry` dataset, instead.

B. The `recsyslearn` library

This Appendix provides additional information on the functionalities of `recsyslearn`, which is a Python library to preprocess RS datsets and evaluate recommendation lists. The following subsections describe the main classes of the library, categorizing them in those for dataset preprocessing and for evaluation of the recommendation lists.

B.1. Dataset

`recsyslearn` simplifies the process of calculating item popularity and user activity, and of segmenting (i.e., categorizing) users and items into groups. The users and items can be segmented based on various criteria, hence providing the basis for group fairness analyses on several dimensions.

`PopularityPercentage` assigns items a popularity value, or user an activity value, corresponding to the percentage of user-item interactions.

`DiscreteFeatureSegmentation` segments the users or items into groups based on one of their categorical features (e.g., user gender, or item genre).

`InteractionSegmentation` segments the items or users after sorting them according to the number of interactions and based on the cumulative number of interactions in each group. For instance, segmenting the items keeping the function's argument for the split to the default value of $[0.8, 0.2]$, the most popular items corresponding to the first group account for 80% of the interactions, and the items in the second group account for the remaining 20%.

`ActivitySegmentation` segments the items or users after sorting them according to the number of interactions and based on the number of items or users in each group. For instance, segmenting the users keeping the function's argument for the split to the default value of $[0.8, 0.2]$, the most active 80% users will belong to the first group, and the least active 20% users to the second.

The methods of the classes for dataset preprocessing take as argument the dataset in the form of a pandas[7] DataFrame.

[7] https://pandas.pydata.org/

B.2. Evaluation

For the evaluation of recommendation lists, `recsyslearn` provides classes with methods that accept as arguments the pandas DataFrames representing the recommendation lists and the target user-item interactions (i.e., the validation or test set).

`NDCG` computes the NDCG at a specific cutoff k. A list of values of k can also be passed as argument to compute the corresponding values of NDCG@k on the same recommendation list.

`Coverage` evaluates the proportion of items that appear in the recommendation lists at least once.

`Novelty` implements the definition of novelty provided by Schedl et al. [13] and extends it allowing for a definition of item popularity in terms of percentage of interactions (as in the original definition) or in terms of popularity class (e.g., as defined by `InteractionSegmentation`).

`Entropy` [35] computes the Shannon's entropy of utility (i.e., recommendations or accurate recommendations) over user or item groups.

`KullbackLeibler` [35] measures the KL divergence between the distribution of utility over user or item groups computed on the list of recommendations, and a target distribution passed to the function as additional argument.

`MutualInformation` implements the definition of mutual information provided by Amigó et al. [35], which measures to what extent the information on the user's group provides information about the item groups to which the recommendations provided to the user belong.

For more details on the functionalities of `recsyslearn` we refer the reader to the official documentation[8] and to the GitHub repository,[9] which also provides several examples on how to use the library.

[8] recsyslearn.readthedocs.io

[9] github.com/giuliowaitforitdavide/recsyslearn/

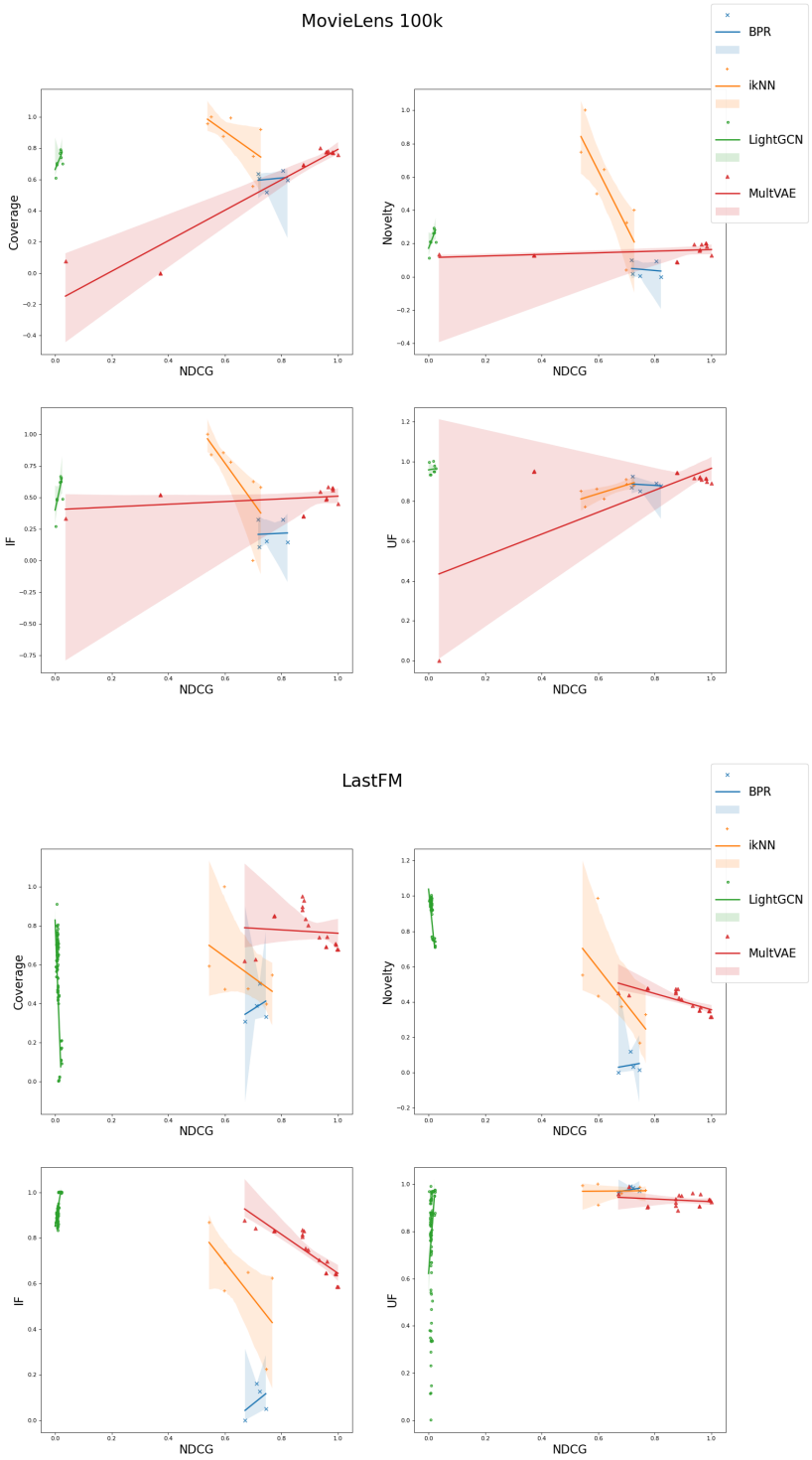

Figure 2: Performance of the Pareto optimal configurations P_i in terms of accuracy (x axis) and beyond-accuracy (y axis) metrics on the test set of `MovieLens100K` and `LastFM`. Each color represents a different recommendation algorithm. Shaded areas correspond to the 1σ confidence intervals of the linear regression.