User Modeling and User-Adapted Interaction (2023) 33:43–79
https://doi.org/10.1007/s11257-022-09334-x

# A bias detection tree approach for detecting disparities in a recommendation model's errors

**Joanna Misztal-Radecka**[1] · **Bipin Indurkhya**[2]

Received: 28 May 2021 / Accepted in revised form: 29 May 2022 / Published online: 11 July 2022
© The Author(s), under exclusive licence to Springer Nature B.V. 2022

**Abstract**
Many of the current recommendation systems are considered to be blackboxes that are tuned to optimize some global objective function. However, their error distribution may differ dramatically among different combinations of attributes, and such algorithms may lead to propagating hidden data biases. Identifying potential disparities in an algorithm's functioning is essential for building recommendation systems in a fair and responsible way. In this work, we propose a model-agnostic technique to automatically detect the combinations of user and item attributes correlated with unequal treatment by the recommendation model. We refer to this technique as the *Bias Detection Tree*. In contrast to the existing works in this field, our method automatically detects disparities related to combinations of attributes without any a priori knowledge about protected attributes, assuming that relevant metadata is available. Our results on five public recommendation datasets show that the proposed technique can identify hidden biases in terms of four kinds of metrics for multiple collaborative filtering models. Moreover, we adapt a minimax model selection technique to control the trade-off between the global and the worst-case optimizations and improve the recommendation model's performance for biased attributes.

**Keywords** Recommender systems · System fairness · Bias detection

## 1 Introduction

Modern recommendation systems are often designed as complex hybrid architectures that deal with heterogeneous input features and combine diverse types of algorithms

✉ Joanna Misztal-Radecka
joanna.misztal-radecka@ringieraxelspringer.pl

Bipin Indurkhya
bipin.indurkhya@uj.edu.pl

[1] Ringier Axel Springer Polska, Warsaw, Poland

[2] Jagiellonian University, Kraków, Poland

(Ricci et al. 2015). In many such systems, applying a global objective results in optimizing the mainstream trends while minority preference groups, as well as those interested in niche products, are not represented well (Beutel et al. 2017; Chen et al. 2020). Given a lack of understanding of the dataset characteristics and insufficient diversity of represented individuals, such approaches inevitably lead to amplifying hidden data biases and existing disparities (Chen et al. 2020). Chen et al. (2020) summarize different situations when a recommendation system may propagate hidden biases and unfairness. They distinguish seven distinct types of biases, including bias resulting from implicit and explicit feedback, bias in the model and in the results, and unfairness toward certain users or societal groups based on their sensitive attributes. Similarly, various types of biases related to social data were distinguished by Olteanu et al. (2019), resulting from the differences in population characteristics, behavioral and content production biases, temporal variations, and content redundancy. Moreover, bias is often caused by certain combinations of circumstances rather than a single feature. For instance, a recommender may propagate an unintended gender bias from the training data by underestimating the preferences of female students for technical courses (Yao and Huang 2017).

While designing a recommender as a transparent box (Guidotti et al. 2018; Rudin 2019; Tintarev and Masthoff 2011) may help in identifying such hidden disparities, it is often not applicable for many practical use cases. As observed by Ribeiro et al. (2016), *restricting machine learning to interpretable models is often a severe limitation*. For instance, due to the high costs of model training and serving infrastructure, many companies decide to use machine learning as a service (MLaaS) solutions where the recommendations are generated by blackbox services such as Amazon Personalize,[1] Google Recommendations AI,[2] Azure ML,[3] and other external tools. In such cases, the exact functioning of the algorithm remains unknown, and it is not possible to use a transparent-box design approach to provide explainability (Guidotti et al. 2018). Moreover, it is often not possible to build explanations of the bias by model inspection (Guidotti et al. 2018), as the attributes related to disparities are not directly fed into the model and hence their direct impact on predictions remains unknown. For instance, a collaborative filtering recommender is trained with user-item interactions, yet it may propagate latent biases for particular demography groups even though this information is not directly observable (Yao and Huang 2017). In such situations, a post hoc analysis and model-agnostic detection (Ribeiro et al. 2016) of disparities in error distribution among distinct user and item attributes that are available may indicate potential biases and unfairness to enable taking further remedial actions.

A majority of existing approaches to detecting disparities are based on analyzing a single source of bias based on dimensions selected a priori, such as a sensitive user attribute or a protected category of products (Sánchez and Bellogín 2019; Beutel et al. 2017, 2019; Wan et al. 2020; Yao and Huang 2017; Kershaw et al. 2021; Wei et al. 2021; Zhu et al. 2021). However, it is not clear how to identify groups that

[1] https://aws.amazon.com/personalize/.

[2] https://cloud.google.com/recommendations.

[3] https://azure.microsoft.com/pl-pl/blog/building-recommender-systems-with-azure-machine-learning-service/.

should be protected, and different types of recommendation algorithms are prone to different vulnerabilities. For instance, while it was found in Yao and Huang (2017) that a CF recommender underestimates the preferences of female students for technical courses, it is possible that there exist other types of stereotypical biases that the authors were not aware of (for instance, for male students and arts classes). In our research, we address the problem of identifying such hidden sources of disparities by automatically identifying the particular circumstances when a recommender system's predictions are less accurate, without making any a priori assumptions. Similar to existing works in this field (Sánchez and Bellogín 2019; Beutel et al. 2017, 2019; Wan et al. 2020; Yao and Huang 2017; Kershaw et al. 2021; Wei et al. 2021; Zhu et al. 2021), we assume the availability of user and item metadata and activity features. However, in contrast to other research, our approach does not require information about which attributes are protected.

While handling different types of bias usually requires an in-depth analysis, and selecting an adequate debiasing technique, some basic approaches can be applied to reduce the potential bias during the model selection stage (Beutel et al. 2017; Diana et al. 2020). In our work, we propose the Lambda-Minimax debiasing model selection method which adapts the Minimax Group Fairness approach (Diana et al. 2020). This method enables a balance between the global optimization and the worst-case error metrics for the previously detected combination of attributes. The model selection process optimizes the objective for the most biased case apart from the global optimization, and therefore, error for the most discriminated cases should be reduced in comparison with a pure global optimization.

### 1.1 Contributions

This paper makes the following contributions:

1. A model-agnostic Bias Detection Tree (BDT) approach is proposed to detect any systematic or unintentional bias in the recommendation model's performance. Our method is based on a post hoc tree-based model, and, in contrast to other existing works on recommendation bias detection, it enables identifying those combinations of attributes for which the recommendations are significantly less accurate than in other cases without a priori knowledge about the protected categories.
2. We show that our proposed method can detect different types of disparities by analyzing the disparities in four kinds of metrics for collaborative filtering algorithms on five public datasets. Moreover, we demonstrate that the biases detected for the feature combinations are more severe than for single attributes. We consider here attributes associated with different types of biases and unfairness situations to identify complex sources of inequalities.
3. We propose a Lambda-Minimax debiasing model selection technique to improve the functioning for the worst-case combinations of attributes, while controlling the global optimization trade-off.

Our approach is based on a well-established tree-based technique (Kass 1980), while its novelty lies in applying this method to detect algorithmic biases in the predictions of a recommendation model. The preliminary results of our proposed method were

presented in a workshop paper (Misztal-Radecka and Indurkhya 2021). This work has been significantly extended now with an in-depth evaluation on multiple datasets from different domains, concerning different types of error metrics. Moreover, we have carried out an extensive theoretical analysis and developed a debiasing technique based on our approach. All these results are reported in the current paper. The code is available on Github.[4]

The rest of this paper is organized as follows. Section 2 summarizes the current challenges of recommendation systems and state of the art in the field of algorithmic fairness and explainability. The research problem is articulated in Sect. 3 and the proposed approach is described in Sect. 4. The experimental evaluation is presented in Sect. 5. Finally, we summarize the conclusions and future work directions in Sect. 7. We provide an analysis of computational complexity along with practical considerations in "Appendix A".

## 2 Background and related work

In this section, we summarize the current challenges related to biases in recommendation systems (Sect. 2.1). Then we describe the state of the art in algorithmic fairness (Sects. 2.2 and 2.3) and debiasing techniques (Sect. 2.4). Finally, we compare our proposed approach with the existing works from this field (Sect. 2.5).

### 2.1 Bias in recommendations

Current recommendation systems face many tough problems when dealing with real-world data, thereby degrading their performance in various situations. A summary of biases existing on the Web was presented by Baeza-Yates (2018) whereas various types of biases related to social data are distinguished by Olteanu et al. (2019). Similarly, in Chen et al. (2020), the authors distinguish seven distinct types of bias in recommendations. Bias in data is caused by the observational rather than experimental data collection process in recommendation systems (Chen et al. 2020) and may occur during the collection or processing of data (Olteanu et al. 2019). *Data bias* happens when the distributions in training and test sets are different. In particular, the *population bias* (Olteanu et al. 2019) results from the differences in user characteristics and demography between the training and the target data. This category also includes *selection bias,* which refers to the *missing not at random* problem of user ratings, and *exposure bias,* which refers to the situations when negative interactions are caused by a lack of exposure to some items. *Conformity bias* is caused by the users' behavior according to their group preferences. Conversely, the *grey and black sheep* users have unusual tastes and so have no or few similar users, therefore the CF methods fail to find adequate recommendations for them (Su and Khoshgoftaar 2009; McCrae et al. 2004). As observed by Baeza-Yates (2018), the *presentation* and the *position* biases are related to how particular items are displayed to a user on a website: Better-positioned products have a higher probability of being selected than the ones not so prominently visible.

[4] https://github.com/JoannaMisztalRadecka/recommendation-bias-detection.

In contrast to data bias, as defined by Baeza-Yates (2018), *algorithmic bias* (Boratto et al. 2019) is added by the algorithm itself. Bias and unfairness in results include the popularity bias (Abdollahpouri et al. 2017; Wei et al. 2021; Zhu et al. 2021; Boratto et al. 2021) (when *"popular items are recommended even more frequently than their popularity would warrant"* Chen et al. 2020) and unfairness in terms of discrimination of some users (groups or individuals) (Kershaw et al. 2021). For instance, Tsintzou et al. (2018) notes that a popular KNN recommendation algorithm tends to amplify the gender bias related to movie genres. The language bias is another significant phenomenon on the Internet: As observed by Baeza-Yates (2018), more than half of the web content is in English, whereas only 27% of the web users speak English. More generally, a small number of influential users may have a large impact on the recommendations of other users, which is also referred to as an *activity bias* (Li et al. 2021; Eskandanian et al. 2019; Baeza-Yates 2018; Olteanu et al. 2019). Additionally, offline recommendation systems face many challenges in dynamic domains such as online services due to user interest shift and dynamic popularity trends. Such temporal variations may impact the quality of recommendation for some groups of items or users (Boratto et al. 2019; Anelli et al. 2019; Olteanu et al. 2019). For example, a news recommendation system may consider that certain football-related articles were popular last week and may assign similar articles appearing this week high probability scores. However, the popularity could be due to a match that took place last week, and so the probabilities for this week will be overestimated. All these situations can lead to generating irrelevant recommendations due to systematic biases and disparities. In our research, we aim to automatically detect such disparities to facilitate taking corrective actions.

### 2.2 Fairness evaluation for ML models

The problem of algorithmic fairness has recently attracted a lot of research interest, and it is central to the research presented in this paper. From the legal perspective, discrimination is a situation where groups are treated *less favorably* by an algorithm (Legislation 2009). A generic definition of discrimination was proposed by Lippert-Rasmussen (2013):

**Definition 1** An agent, $X$, discriminates against someone, $Y$, in relation to another, $Z$, by $\Phi$-ing (e.g., hiring $Z$ rather than $Y$) if, and only if:

1. There is a property, $P$, such that $Y$ has $P$ or $X$ believes that $Y$ has $P$, and $Z$ does not have $P$ or $X$ believes that $Z$ does not have $P$,
2. $X$ treats $Y$ worse than he treats or would treat $Z$ by $\Phi$-ing, and
3. It is because ($X$ believes that) $Y$ has $P$ and ($X$ believes that) $Z$ does not have $P$ that $X$ treats $Y$ worse than $Z$ by $\Phi$-ing.

The property $P$ usually refers to a set of *protected* attributes, as defined by the UK Equality Act,[5] which may be related to the user's demography, religion, sexual orientation, and so on. For ML systems, the action $\Phi$ may be defined in terms of the

[5] https://www.legislation.gov.uk/ukpga/2010/15/contents.

classifier output (for instance, a decision about hiring someone or their credit score) or an evaluation measure (such as the number of false positives). While, in general, fairness may be defined as the lack of discrimination against individuals or groups, it has been explored in the literature concerning distinct aspects. As defined by Gajane and Pechenizkiy (2017), *fairness by treatment* happens when the protected attributes are not used explicitly by a predictor. However, it was shown that removing the sensitive attributes from an input may lead to propagating hidden biases encoded by other non-protected attributes (for instance for race-blind approaches Fryer et al. 2008). *Parity by impact* happens when the outcomes (for instance, the true positive rate) of a predictor are balanced across groups. To formalize the non-discrimination criteria, let us first introduce the definition of conditional independence from the probability theory:

**Definition 2** Two random variables X and Y are independent (denoted as $X \perp Y$) if their probability distribution can be expressed as a product:

$$\forall x \in X, y \in Y : P(X = x, Y = y) = P(X = x)P(Y = y) \tag{1}$$

**Definition 3** Two random variables $X$ and $Y$ are conditionally independent given a random variable $Z$ (denoted as $X \perp Y \mid Z$) if:

$$\begin{aligned} \forall x \in X, y \in Y, z \in Z : P(X = x, Y = y \mid Z = z) \\ = P(X = x \mid Z = z)P(Y = y \mid Z = z) \end{aligned} \tag{2}$$

Accordingly, in Barocas et al. (2019), three fundamental non-discrimination observational criteria for model fairness are defined as properties of the joint distribution of the sensitive attribute $A$ (which is the property $P$ from Definition 1), the target variable $Y$, and the classifier or score $R$ for the training examples $X$. The criteria are defined based on the conditional independence between those variables:

**Definition 4** The random variables $A$, $R$ satisfy independence if $R \perp A$.

The first criterion requires the sensitive characteristic to be statistically independent of the score. It is also referred to by other terms such as demographic parity, statistical parity, group fairness, and disparate impact (Gajane and Pechenizkiy 2017). For a binary problem, this condition means that the *acceptance* rate (number of positive predictions, for instance, for the credit acceptance decision) to equal among the groups, or the difference should be below a certain threshold $\epsilon$ in the relaxed variant.

**Definition 5** The random variables $R$, $A$, $Y$ satisfy separation if $R \perp A \mid Y$.

In the second criterion, the correlation between the score and the sensitive attribute is allowed as long as it is justified by the target variable. In the case of classification, the false-negative rate and false-positive rate should be equal across the groups. (For instance, the number of innocent people classified as criminals should be equal across races.)

**Definition 6** The random variables $R$, $A$, $Y$ satisfy sufficiency if $Y \perp A \mid R$.

The third criterion is satisfied when the sensitive attribute and target variable are clear from the context. In a binary case, this condition implies a parity of positive and negative predictive values across all groups.

As further noticed by Barocas et al. (2019), these criteria cannot be satisfied at the same time, which leads to trade-offs when selecting the fairness goals for a given problem. For instance, it can be shown by applying the contraction rule for conditional independence that the first two rules (independence and separation) are mutually exclusive when the attribute $A$ and $Y$ are not independent:

$$A \perp R \wedge A \perp Y \mid R \implies A \perp (Y, R) \implies A \perp Y \tag{3}$$

More generally, these observational fairness criteria may refer to the features $X$ rather than a single attribute $A$ and may be verified given samples from the joint distribution of these variables.

### 2.3 Fairness evaluation for recommendation systems

While the fairness criteria are well-defined for the standard classification problems in the decision-making systems, their application to the recommendation systems is less straightforward. As pointed out by Burke et al. (2018), user equity may have goals that conflict with personalization for recommendation systems. Kamishima et al. (2011), the model fairness is measured based on the demographic parity; however, as pointed out by Yao and Huang (2017), such an approach is appropriate only when the user behavior does not correlate with their demography, which is often not true for recommender systems. As further noticed by Barocas et al. (2019), there are multiple ways in which a recommendation system may show unfair functioning. For instance, some consumers' informational needs may be addressed better than others, or the content of some producers may be privileged. While the fairness evaluation methods are typically applied to analyze the *unfairness* toward a selected protected attribute, in this work, we generalize the above-mentioned criteria to detect disparities in different biased scenarios, including unfairness and other types of disparities (Chen et al. 2020; Olteanu et al. 2019).

Moreover, system fairness may be evaluated with respect to rankings (Zehlike et al. 2017; Zehlike and Castillo 2020; Singh and Anand 2018), pairwise metrics (Beutel et al. 2019), or prediction errors (Yao and Huang 2017; Beutel et al. 2017; Wan et al. 2020). While ensuring fairness in ranking is an important problem, we focus here on detecting disparities in rating predictions during training that may be further propagated to the rankings.

### 2.4 Debiasing techniques for recommendations

A summary of recent debiasing approaches for recommendation systems is presented in Chen et al. (2020). The authors classify these approaches according to the types of bias addressed. It is also noted that a majority of debiasing techniques address only one or two types of bias, which may not be sufficient for multiple real-world scenarios. In

Beutel et al. (2017), a *focused learning* approach is proposed to learn additional models to improve the recommendation quality for a specified subset of items. However, this method requires training and serving multiple models which may increase the overall costs of the system. In Yao and Huang (2017), the fairness metrics are optimized by including a fairness term in the learning objective. The Minimax Group Fairness technique was proposed by Diana et al. (2020) for a fair model selection in classification and regression problems. In their approach, the information about the worst-case error value is incorporated in the model selection objective. In our research, we generalize this approach to enable controlling the trade-off between the global accuracy and the worst-case optimization of recommendation models.

### 2.5 Related research

The definition of bias in this work is based on the fairness objectives defined by Yao and Huang (2017) for measuring the disparities in rating prediction errors between the advantaged and disadvantaged users. Similarly, in Wan et al. (2020) the rating prediction error of collaborative filtering algorithms is compared for distinct consumer-product market segments revealing certain biases propagated by the recommender systems. The authors assume that *a fair algorithm is supposed not to worsen the market imbalance in interactions* and they focus on enhancing the parity of prediction errors in recommendations for combinations defined by the product image and the consumer's identity. Similarly, in Sánchez and Bellogín (2019), the evaluation metric for recommendations is aggregated according to the groups defined by the user attributes to detect if an algorithm makes more relevant recommendations for users belonging to some specific groups. As shown in Beutel et al. (2017), a recommender tuned to improve a global prediction objective leaves many items badly modeled in terms of prediction error, and thus these products are under-served.

We would like to point out that in most of these approaches, recommendation fairness evaluation is limited to a single pre-defined attribute (such as a demography feature Sánchez and Bellogín 2019 or an item category Beutel et al. 2017, 2019), and the combinations of potentially discriminated attributes are defined a priori (Wan et al. 2020; Yao and Huang 2017). In our work, we consider disparity related to a combination of factors and aim at generating these combinations automatically without any a priori assumptions about protected groups.

As observed by Diana et al. (2020), "*(...)equalizing error rates and similar notions may require artificially inflating error on easier-to-predict groups - without necessarily decreasing the error for the harder to predict groups(...)*". Accordingly, they propose *minimax group fairness* for minimizing the maximum group loss instead of equalizing the results across groups. Following this observation, we extend their technique by adding a $\lambda$ coefficient to control the global and the worst-case optimization trade-off.

We use a tree-based model for explaining the model errors; thus, our approach may be formally characterized as an *Explanation Through Interpretable Model via Single Tree Approximation* (Guidotti et al. 2018). Similar approaches were previously applied for building explainable recommenders (Zhang and Chen 2018; Tintarev and Masthoff 2011; Singh and Anand 2018). For example, a tree-based model is trained by

Singh and Anand (2018) on a set of interpretable labels to provide explanations for a black-box learning-to-rank algorithm on web search. Though our work also applies a tree-based post hoc model, we aim at approximating the model errors, whereas (Singh and Anand 2018) focuses on evaluating how well the post hoc model approximates the original ranking. Thus, we use a similar model to solve a different type of problem, and therefore these approaches are not comparable. The concept of training new models on the errors of previous ones is reminiscent of the gradient boosting technique, which is commonly applied in ensemble learning approaches. However, the goal of gradient boosting is to improve the accuracy of weak learners while we use this technique for identifying error-prone situations.

## 3 Problem definition

### 3.1 Motivation

In most standard recommendation algorithms, the model parameters are adjusted based on a global optimization function. For instance, for the popular model-based matrix factorization algorithms (Koren et al. 2009), the predicted ratings are calculated as:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \tag{4}$$

where $\mu$ is the overall average of the ratings, $q_i$ and $p_u$ represent the latent vector representations of user $u$ and item $i$ respectively and $b$ denotes the *bias intercept* assigned to a user $u$ or an item $i$ which is independent of the user-item interaction. (For instance, some users tend to give higher ratings and some items are more popular than others.) The learning is performed by minimizing a global objective, which is defined as the sum of the differences between the real $r$ and the predicted $\hat{r}$ ratings in training set for all the data samples (with $\lambda$ regularization coefficient):

$$\mathcal{L} = \sum_{r_{ui} \in R_{train}} \left(r_{ui} - \hat{r}_{ui}\right)^2 + \lambda \left(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2\right) \tag{5}$$

Hence, the model weights are adjusted to optimize the predictions for the majority of the cases or the most impactful instances. As shown by Beutel et al. (2017), this leads to underfitting for some unusual situations such as specific interest groups and niche products and results in a long-tailed distribution of the prediction errors.

Another example of unequal error distribution is the standard k-nearest neighbors collaborative filtering algorithm. In the kNN methods, the rating prediction for the given user and item is calculated by taking the weighted average of all the ratings of the $k$ most similar users (user-user approach):

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot r_{vi}}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)} \tag{6}$$

or items (item–item approach):

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} \text{sim}(i, j) \cdot r_{uj}}{\sum_{j \in N_u^k(i)} \text{sim}(i, j)} \tag{7}$$

where $N_i^k(u)$ denotes $k$ nearest neighbors of $u$ who have ratings for item $i$ and $\text{sim}(u, v)$ is a similarity measure between vectors $u$ and $v$. To generate a top-$N$ recommendation, the list of $N$ most frequent items from $k$ closest neighbors is aggregated as the recommendation. Hence, the predictions may be inaccurate for users who have low similarity to others (*grey sheep*) or those with very few ratings (the *cold-start* problem).

Moreover, when tuning the model hyper-parameters (such as the number of neighbors for KNN or the vector size for MF), the performance of the model on the validation set is estimated with an average error measure, such as MSE, or a performance metric such as the precision or NDCG of all the examples. Though the optimization is performed globally, the model may underperform in certain situations, such as for particular types of users or items. As the average error does not carry any information about the metric's distribution, these hidden biases may remain unnoticed.

### 3.2 Research problem

To define the research problem formally, we introduce the following notations. Let $r_{u,i} \in \mathbb{R}$ denote a rating (interaction) of user $u$ for item $i$. Then, $\hat{r}_{u,i}$ is the predicted rating (score) for user $u$ for item $i$. We refer to a pointwise performance measure defined as an error function applied to the real and predicted ratings for user $u$ and item $i$ $\mathcal{E}(r_{u,i}, \hat{r}_{u,i}) \in \mathbb{R}$.

Then, $X$ is the set of the user and the item attribute features, $x^{ui} = [x_1^{ui}, \ldots, x_N^{ui}] \in X$ is a vector of user and item attributes associated with rating $r_{u,i}$ (for instance $x_1$ is the item's category, $x_2$ is the production year and $x_3$ is the user's age, $N = 3$) and $T_k = \{k_1, \ldots k_K\}$ is a set of possible values for an attribute $x_k$. (For instance, for feature *genre* the possible values are $T_{genre} = \{crime, comedy, thriller\}$, $K_{genre} = 3$.) We define the *recommendation context* as a subset of instances with specific combinations of attributes:

$$\begin{aligned} &X_{k_n,l_m,\ldots} = X_{k_n} \cap X_{l_m} \cap \ldots = \{x^{ui} : x_k^{ui} = k_n \wedge x_l^{ui} = l_m \wedge \ldots\}; \\ &X_{k_n,l_m,\ldots} \subset X \\ &X_{\sim k_n,l_m,\ldots} = X \backslash X_{k_n,l_m,\ldots} \end{aligned} \tag{8}$$

For instance, a context $X_{gender_{female}, genre_{crime}}$ refers to all the cases $x$ where $gender = female$ and $genre = crime$, while $X_{\sim gender_{female}, genre_{crime}}$ denotes the remaining cases (where $gender \neq female$ or $genre \neq crime$).

The average error for ratings with attributes from $X_{k_n,l_m,\ldots}$ (for instance, the average error between the real and the predicted ratings of female users for crime movies) is denoted by:

$$\overline{\mathcal{E}}_{X_{k_n,l_m,\ldots}} = \sum_{n=1}^{N} \frac{\mathcal{E}(r_{u,i}, \hat{r}_{u,i})}{N}, \quad x^{ui} \in X_{k_n,l_m,\ldots} \tag{9}$$

where $N$ is the number of instances in $X_{k_n,l_m,\ldots}$.

Then, let us formulate the generic definition of a fair recommendation model:

**Definition 7** Given a rating prediction error function $\mathcal{E}$ and a set of attribute combinations $X$ that describe the recommendation context, a recommendation model is considered fair if $\mathcal{E} \perp X$.

which means that the performance of a fair recommender in terms of the error metric $\mathcal{E}$ does not depend on the recommendation context. In case when $\mathcal{E}$ is a pointwise error metric between $r_{u,i}$ and $\hat{r}_{u,i}$, this can be written by adapting the *separation* criterion for algorithmic fairness (Definition 5): $\hat{R} \perp X \mid R$.

It should be noted that this criterion does not imply the parity of predictions ($\hat{R} \mid X$ from Definition 4)—for instance, one group of users $k_n$ may be more likely to highly rate particular type of items $l_m$ and the predictions for these instances will be higher, yet the error rate should be equal for all cases. Moreover, while the term *fairness* usually refers to a set of sensitive user attributes, in this research, we focus on the equality of error rates for any user or item attribute.

**Definition 8** If the performance $\mathcal{E}(r_{u,i}, \hat{r}_{u,i})$ of a recommendation model depends on a recommendation context $X_k$, the recommendations are considered biased with respect to $k$.

The bias in this definition may be viewed as a specification of Definition 1 (Lippert-Rasmussen 2013) where the action $\Phi$ is evaluated with the error metric $\mathcal{E}$, and the property $P$ is the recommendation context $X$. However, since the dependence may be due to some latent factors extracted from the behavioral patterns that are not given explicitly, we consider the *parity by impact* rather than *fairness by treatment* (Gajane and Pechenizkiy 2017). Hence, the dependence between the error and attributes is inferred from the observable samples from the distribution of $\mathcal{E}(r_{u,i}, \hat{r}_{u,i})$ and attributes $X$.

**Definition 9** If the average error $\overline{\mathcal{E}}$ for a recommendation context $X_{k_n,l_m,\ldots}$ is significantly higher than the average error for the rest of cases

$$B = \{(k_n, l_m, \ldots) : \overline{\mathcal{E}}_{X_{k_n,l_m,\ldots}} > \overline{\mathcal{E}}_{X_{\sim k_n,l_m,\ldots}}\} \tag{10}$$

the recommendation algorithm is *biased* against $X_{k_n,l_m,\ldots}$ in terms of the error metric $\mathcal{E}$.

This research aims to detect the circumstances $X_{k_n,l_m,\ldots}$ that lead to a systematic bias by a given recommendation algorithm concerning a pointwise error metric $\mathcal{E}$. This information may contain user and item attributes—either explicit (such as user demography or category of an item) or implicit (such as the activity rate), as well as latent interest patterns learned by the model.

Additionally, the information about potential biases must be understandable so that adequate remedial actions may be taken to address the disparity problem (such as increasing the dataset diversity or tuning the algorithm parameters). Hence, the explainability of the model must be ensured for the bias detection method.

### 3.3 Measuring disparities

We adapt here four unfairness criteria for recommendation systems that were proposed by Yao and Huang (2017) based on distinct types of errors to apply them to different types of biases. The average of each error is calculated for the groups defined by the attributes $X_{k_n,l_m,\ldots}$ (as defined in Eq. 9), where $\mathcal{E}(r_{u,i}, \hat{r}_{u,i})$ is defined below for each of these metrics.

**Definition 10** Value error measures inconsistency in signed estimation error:

$$\mathcal{E}_{\mathcal{V}}(r_{u,i}, \hat{r}_{u,i}) = r_{u,i} - \hat{r}_{u,i} \tag{11}$$

As defined by Yao and Huang (2017), the value error occurs when one group $X_{k_n,l_m,\ldots}$ *is consistently given higher or lower predictions than their true preferences*. For instance, this could mean that the ratings for the last year's Oscar movies are consistently too high or the ratings of female users for horror movies are consistently too low. However, if the predicted ratings are sometimes too low and sometimes too high, this error will be close to zero. This measure combines the underestimates and the overestimates of the error (defined below) and is related to the prediction bias (in contrast to its variance or noise).

**Definition 11** Absolute error measures inconsistency in absolute estimation error:

$$\mathcal{E}_{\mathcal{A}}(r_{u,i}, \hat{r}_{u,i}) = |r_{u,i} - \hat{r}_{u,i}| \tag{12}$$

Following Yao and Huang (2017), the absolute error occurs *"if one user type has a small reconstruction error and the other user type has a large reconstruction error; one type of user has an unfair advantage of a good recommendation, while the other user type has a poor recommendation"*. The same definition applies to the attributes of items or a combination of user and item attributes. For instance, if the predicted ratings of young users for documentary movies have a high absolute error, it indicates an inaccuracy without defining the sign of the difference. This type of error measures both bias and variance in predictions for a particular group.

**Definition 12** The underestimate error measures inconsistency in how much the predictions underestimate the true ratings:

$$\mathcal{E}_{\mathcal{U}}(r_{u,i}, \hat{r}_{u,i}) = max(r_{u,i} - \hat{r}_{u,i}, 0) \tag{13}$$

This type of error happens when the ratings for one group $X_{k_n,l_m,\ldots}$ are consistently too low (for instance, the ratings of female users for horror movies).

**Definition 13** The overestimate error measures inconsistency in how much the predictions overestimate the true ratings:

$$\mathcal{E}_{\mathcal{O}}(r_{u,i}, \hat{r}_{u,i}) = max(\hat{r}_{u,i} - r_{u,i}, 0) \tag{14}$$

Contrary to the *underestimate error*, an overestimation occurs when the predictions for $X_{k_n,l_m,\ldots}$ are consistently too high (for instance, for the last year's Oscar movie).

Additionally, for evaluating the disparities in pointwise rating prediction error, we define a *total bias* measure for the whole dataset, which measures the difference between the two most extreme cases:

$$\Delta^{total}_{bias}\overline{\mathcal{E}}(X_{max}, X_{min}) = |\overline{\mathcal{E}}_{X_{max}} - \overline{\mathcal{E}}_{X_{min}}| \tag{15}$$

where $X_{min}$, $X_{max}$ denote the combinations of attributes with the smallest and the largest error values, respectively, and $\overline{\mathcal{E}}$ is calculated according to Eq. (9). Note that this metric measures the equality of results among the groups regardless of the global average.

## 4 Proposed algorithm

This section proposes a bias detection tree algorithm to detect the combinations of user and item attributes that may be associated with a less accurate performance of recommendations.

### 4.1 Algorithm description

In the proposed approach (Algorithm 1), a post hoc decision tree model is applied to identify the combinations of explicit user and item attributes for which the recommendations are significantly worse than in other cases.

The problem of finding the optimal tree structure for a given dataset is known to be NP-complete. Thus, the BDT algorithm is based on chi-square automatic interaction detection (CHAID) tree for regression, which is a well-established statistical method proposed by Kass (1980). CHAID decision tree method is a greedy heuristic algorithm that usually finds a reasonable solution, yet it is not guaranteed to be optimal.

The tree model is fitted with the inputs of $x^{ui}$ as independent variables and the target $\mathcal{E}$, that may be one of the pointwise error metrics $\mathcal{E}(r_{u,i}, \hat{r}_{u,i})$ (Yao and Huang 2017), as the predicted dependent variable. In each node of the decision tree, the split is performed based on the significant difference in the error distribution between the instances. The algorithm consists of two stages:

1. *Merge phase* (Algorithm 1, Steps 2–4)
   The goal of this stage is to find the set $T_k^*$ of combinations of attribute values from $T_k$ for each attribute $k$ such that the values with no significant difference in the error distribution are merged together. However, to find an exact solution, all possible subsets of the set $|T_k| = n$ should be explored, and the number of such subsets

can be calculated as the Bell's number, which is large even for small $n$. Hence, in the merge stage, the CHAID algorithm applies a heuristic method for selecting the splits that iteratively merges pairs of categories with insignificant differences in a greedy way.

2. *Split phase* (Algorithm 1, Step 5)
   The second stage of the CHAID method aims at finding the attribute $k$ that will be split as the next node in the tree. The node selection is determined by the most significant difference (the smallest adjusted $p$-value) concerning the predicted error value.

This procedure is repeated recursively until no further splits can be done. ($p > \alpha_{split}$ for all nodes or the max tree depth was achieved.) An additional step may be added after the merging phase to split compound categories consisting of over two single categories into two significantly different subgroups (Kass 1980).

**Algorithm 1** Bias Detection Tree

1: **for** $k = 1 \ldots K$ **do**
2: Calculate the pairwise $p$-value for the target error metric $\overline{\mathcal{E}}_{X_{k_n}}$ and $\overline{\mathcal{E}}_{X_{k_m}}$ for all pairs of attribute values $k_m, k_n \in T_k$ and select the pair $k_m, k_n$ with the least significant difference.
3: If this difference is not significant $p > \alpha$, merge these categories into one compound category $k_{m,n}$ and add the new category to $T_k$.
4: Repeat steps 2–3 until there are no more attribute values to be merged in this way.
5: Select attribute $k$ with the smallest adjusted $p$-value (using Bonferroni correction) as the next node in the tree and perform split into categories from $T_k$ if $p < \alpha_{split}$.
6: **end for**
7: Repeat steps 1–5 for each node recursively until there are no attributes with a significant difference in the error metric distribution or the maximum tree depth was achieved.

The detected biases are visualized as the decision tree rules that lead to the significant differences for the target error metric.

### 4.2 Hypothesis testing

Since the independence in Definition 9 is inferred from the observable samples from population, it is prone to sampling errors. Accordingly, an adequate hypothesis test should be applied to verify the significance of the obtained results. In a general case, we define the following null-hypothesis $H_0$ and an alternative hypothesis $H_1$:

$H_0$ : the mean value of error metric $\mathcal{E}$ among different recommendation contexts $X$ is equal.
$H_1$ : there exists a context $k_n, l_m, \ldots$ such that:

$$\overline{\mathcal{E}}_{X_{k_n,l_m,\ldots}} > \overline{\mathcal{E}}_{X_{\sim k_n,l_m,\ldots}} \tag{16}$$

If the resulting $p$-value is less than the significance level $\alpha$, the obtained variance differences are unlikely to have occurred in random samples taken from a population with equal variances, and the null hypothesis is rejected. Importantly, if the null hypothesis

is not rejected, it does not imply that it is necessarily true but that we do not have enough evidence to reject it. (For instance, there may additional latent variables that are not given in $X$ or the sample size is too small.) The alternative hypothesis can also be written for a two-tailed comparison so that both the disadvantaged and advantaged cases will be detected:

$$\overline{\mathcal{E}}_{X_{k_n,l_m,\ldots}} \neq \overline{\mathcal{E}}_{X_{\sim k_n,l_m,\ldots}} \tag{17}$$

Since we do not make any assumptions about the distribution of errors $\mathcal{E}$, an adequate statistical test should be selected to verify the significance of the difference depending on its distribution. In the experiments with prediction errors that have continuous values and do not necessarily have a normal distribution, the median Levene's test (Brown and Forsythe 1974) is used for assigning the statistical significance of particular splits by evaluating the homogeneity of their variances that provides robustness for non-normal data while retaining good statistical power.

### 4.3 Example

To generate an illustrative example of the bias detection process, we simulated predictions from a biased recommender and analyzed the detected disparities. A synthetic recommendation dataset is generated for 10,000 user-item ratings generated from a uniform distribution between $r_{u,i} \sim \mathcal{U}(1, 5)$. Additionally, each rating is accompanied by a vector of user and item attributes $x^{ui}$:

- gender $x^{gender} \sim \mathcal{U}(\{female, male, other\})$,
- age $x^{age} \sim \mathcal{N}(\mu = 40, \sigma = 15)$ (bucketized into 3 bins based on the percentiles),
- movie genre $x^{genre} \sim \mathcal{U}(\{thriller, comedy, crime, action\})$,
- movie year $x^{year} \sim \mathcal{N}(\mu = 2000, \sigma = 15)$ (bucketized into 3 bins based on the percentiles).

The synthetic model predictions were generated by adding random Gaussian noise to the real target values from the test set:

$$\hat{r}_{u,i} = r_{u,i} + \epsilon, \epsilon \sim \mathcal{N}(\mu, \sigma) \tag{18}$$

where the distribution parameters of this noise determine the error characteristics. A perfect theoretical predictor has $\mu = 0, \sigma = 0$ (outputs the real ratings in all cases). The value of $\mu$ defines how much the predicted ratings are different from the true value, while $\sigma$ determines how the results vary among the samples set. We consider here the following simulated cases with different types of systematic biases that correspond to the fairness objectives defined by Yao and Huang (2017):

1. *Fair recommendations* (Fig. 1a)—$\mu = 0$ and $\sigma = 0.1$ are equal for all subsets of attributes.
2. *Underestimation disparity* (Fig. 1b)—the predictions for certain situations are systematically underestimated: $\mu_b < 0$. In this example, the predictions for female

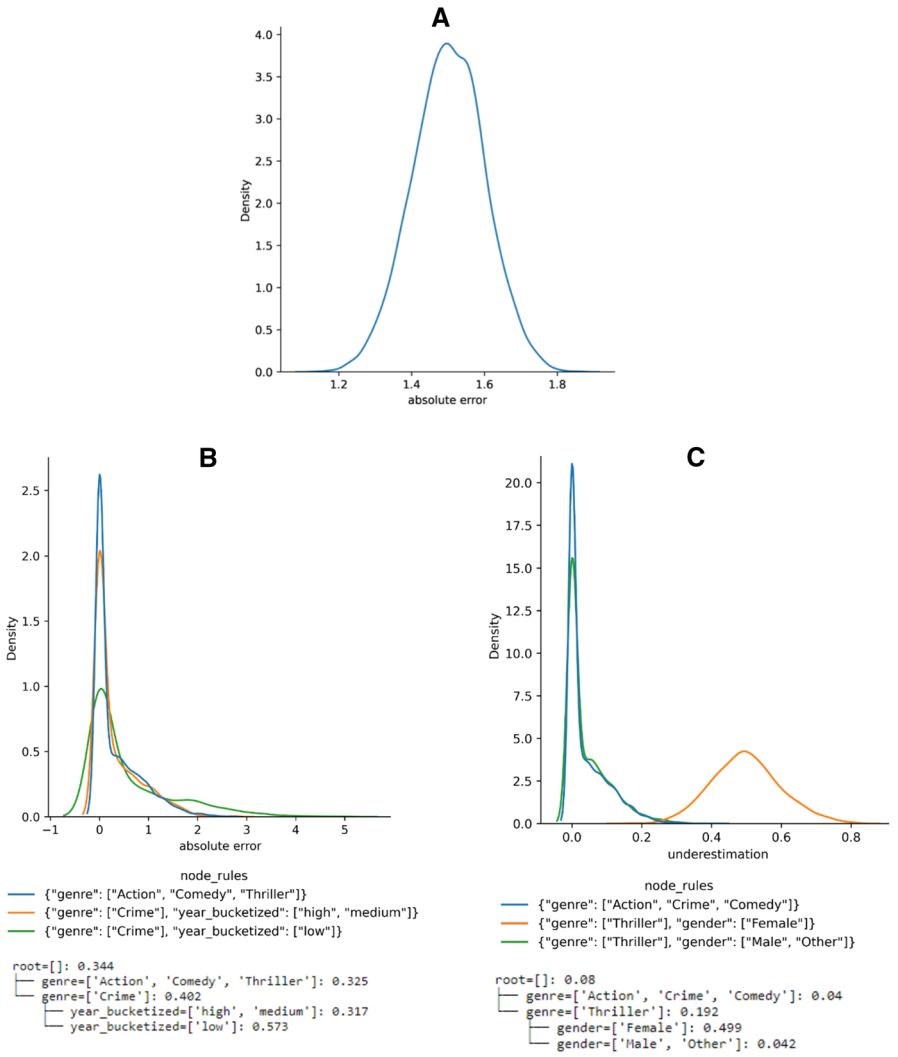


**Fig. 1** Examples of the distribution and conditions of the detected recommendation biases for the simulated cases: **a** A fair recommendation model. **b** A recommender makes systematically inaccurate predictions for old crime movies (higher $\sigma$ of the generated rating distribution). **c** A recommender makes systematically lower predictions of thriller movie ratings for female users (lower $\mu$ of the generated rating distribution). The probability density ($Y$-axis) of the absolute error of the model ($X$-axis) is estimated with KDE

users and thriller movies are systematically lower than the real ratings, which leads to a higher error on the test set ($\sigma = 0.1$ for all cases).

$$\mu = \begin{cases} -0.5 & \text{if } gender = F \text{ and } genre = thriller \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

**Table 1** *P*-values (below the diagonal) and test statistics (above the diagonal) for genres in the absolute unfairness example

| *p*-value/test stat. | Thriller | Comedy | Crime | Action |
|---|---|---|---|---|
| Thriller | – | 1.027 | 10.782 | 0.086 |
| Comedy | 0.311 | – | 18.537 | 1.788 |
| Crime | 0.001* | 1.67e−5* | – | 9.552 |
| Action | **0.768** | 0.181 | 0.002* | – |

The *p*-value for categories action and thriller is the highest (highlighted in bold); hence, these values will be merged

**Table 2** *P*-values (below the diagonal) and test statistics (above the diagonal) for genres in the absolute unfairness example after the first merge

| *p*-value/test stat. | Thriller\|action | Comedy | Crime |
|---|---|---|---|
| Thriller\|action | – | 0.116 | 52.151 |
| Comedy | **0.732** | – | 18.537 |
| Crime | 5.64e−13* | 1.67e−5* | – |

The *p*-value for categories thriller action and comedy is the highest (highlighted in bold); hence, these values will be merged

3. *Absolute disparity* (Fig. 1c)—the prediction accuracy is lower for certain combination of attributes (the model underfits for some situations): $\sigma_b > \sigma$. In this example, the estimations for a category of old crime movies are less accurate and $\mu = 0$ for all cases.

$$\sigma = \begin{cases} 1.5 & \text{if } genre = crime \text{ and } year = old \\ 0.8 & \text{otherwise} \end{cases} \tag{20}$$

In cases 2 and 3, the error of the samples from the biased groups is higher than for the remaining instances. Below we describe the first steps of the tree generation process, for example 3 (*absolute unfairness*).

1. *Merge phase*
   First, the pairwise statistical test is calculated for each pair of values in each category. Resulting *p*-values and test statistics for genre categories are summarized in Table 1. The *p*-value for categories *action* and *thriller* is the highest ($p > \alpha_{merge}$); hence, these values will be merged in the next iteration. Then, the same operation is repeated for the new set of genre values, including the merged category *thriller|action*. The statistics and *p*-values for the next iteration of merging for genres is presented in Table 2 showing that categories *thriller|action* and *comedy* will be merged as their *p*-value is the highest ($p > \alpha_{merge}$). Then, the same procedure is repeated until no category values can be merged in this way and thus the final set of values for *genre* will be $T^*_{genre} = \{\{thriller, action, comedy\}, \{crime\}\}$. The merge phase for other attributes (*year*, *age* and *gender*) is performed in an analogical way.
2. *Split phase*
   Next, in the split stage, the *p*-values and statistics for all attributes are compared (Table 3). The *p*-value of *genre* is the lowest ($p < \alpha_{split}$); hence, this attribute will be selected as the next node split in the tree.

**Table 3** *P*-values and test statistics for all categories in the absolute unfairness example

| | Genre | Year | Gender | Age |
|---|---|---|---|---|
| Statistics | 21.396 | 19.941 | 1.528 | 1.764 |
| *p*-value | **8.25e−14*** | 2.27e−09* | 0.216 | 0.171 |

The *p*-value for genre is the lowest (highlighted in bold); hence, this attribute will be selected for the next split

Stages 1–2 are repeated recursively for each node until no further splits can be done. The resulting bias trees and corresponding distributions of error in each leaf node are presented in Fig. 1. We use kernel density estimate (KDE) to visualize the distribution of the error metric for the detected tree nodes as a continuous probability density curve. The distribution for each node is normalized separately such that the area under each of the density curves sums to 1.

Figure 1 shows the detected bias trees for all simulated cases (1, 2, 3) and corresponding error distributions for each of the branches. In the first example (Fig. 1a), there is no difference in error distribution for any combinations of attributes; hence, no splits were performed by the BDT algorithm. In the second case (Fig. 1b), the average error on the whole dataset is 0.344 (value in the tree root). The split is first performed for the crime movies (error 0.402 vs. 0.325), and then the ratings are divided depending on the movie production year, with the highest error detected for the oldest movies (0.573).

Similarly, in the third example (Fig. 1c), the first split is performed based on the thriller genre (error 0.192 for thrillers and 0.04 for others), and the second split is based on the user's gender (0.499 for female and 0.042 for other users). The average values and distributions of errors for non-thrillers and thrillers for non-female users are equal, and the attributes corresponding to the significant biases were correctly identified. In both these examples (B and C), the average error of the whole sample clearly does not reflect these differences, and the detected tree nodes correspond with the pre-defined biases.

### 4.4 Comparison with other tree-based algorithms

We use the CHAID tree architecture in which the stopping conditions are determined by a statistically significant difference between the children nodes, in contrast to other tree-based models (such as CART) that require post-pruning to remove insignificant splits. Since the more popular decision tree methods such as CART or C4.5 are primarily oriented toward the classification and prediction problems, they use homogeneity measures for determining the splits. In contrast to this, the goal of the CHAID method is mostly to differentiate the groups in terms of the target distribution while the prediction or classification accuracy is rather implicit (Ritschard 2013). Moreover, the CHAID algorithm enables performing non-binary splits, resulting in less complicated and easier to interpret tree structures.

### 4.5 Lambda-Minimax debiasing for model selection

We adapt here the Minimax Group Fairness method, which was proposed by Diana et al. (2020) for optimizing the classification and regression problems, to the recommendation model setting. Accordingly, we incorporate the information about the worst-case loss across detected groups (denoted by $\overline{\mathcal{E}}_{X_{max}}$) into the model selection objective. In the basic minimax model selection process (Diana et al. 2020), the goal is to find a model $h*$ that minimizes the maximum error rate over all groups:

$$h^* = argmin_{h \in H} \{\overline{\mathcal{E}}_{X_{max}}\} \tag{21}$$

To control the trade-off between the minimax and the global optimizations, a relaxed version of this method was proposed by Diana et al. (2020):

> (…) given a target maximum group error bound $\gamma \geq OPT1$, the goal is to find a randomized model that minimizes overall population error while staying below the specified maximum group error threshold.

However, while this definition was proposed for regression and classification problems, we note that the selection of $\gamma$ for the recommendation system evaluation metrics is highly dependent on the dataset characteristics. Accordingly, instead of applying this relaxed variant, we propose here a generalization of the basic minimax definition, Lambda-Minimax, by applying a relaxation coefficient $\lambda_{bias}$, where $\mathcal{E}_{\lambda_{bias}}$ is calculated as a weighted sum of the global error metric $\mathcal{E}$ and the most discriminated subset $\mathcal{E}_{X_{max}}$:

$$\overline{\mathcal{E}}_{\lambda_{bias}} = (1 - \lambda_{bias})\bar{\mathcal{E}} + \lambda_{bias}\overline{\mathcal{E}}_{X_{max}} \tag{22}$$

where $0 \leq \lambda_{bias} \leq 1$ is the weight of the bias metric value. In this approach, the impact of the bias term on the final objective may be controlled by the parameter $\lambda_{bias}$—in particular, when $\lambda_{bias} = 0$, $\mathcal{E}_{\lambda_{bias}}$ is equal to the global measure, and when $\lambda_{bias} = 1$, only the worst-case error is considered (pure minimax approach).

## 5 Experimental evaluation

The preliminary results of the BDT method on a single dataset (Movielens 100K) were presented in Misztal-Radecka and Indurkhya (2021). Here, we extend the experimental evaluation by verifying if the proposed method is capable of detecting disparities that are more severe than for single user or item attributes for different recommendation scenarios with each of the unfairness metrics defined in Sect. 3.3.

First, we analyze the detected disparities for different measures and a deep neural network model trained on five public datasets. Next, we test the proposed approach to bias reduction by introducing the Lambda-Minimax debiasing (Sect. 4.5) to verify if the worst-case error value can be improved in the CF model selection process.

**Table 4** Attributes used in the experiments and potential biases related to each attribute (as available for each dataset)

| Attribute type | Attributes | Potential biases |
|---|---|---|
| User attributes | Gender, age, country, body shape | Model unfairness |
| Item attributes | Item category, production year, size, brand | Disparities in performance among distinct item categories |
| Context | Date of the rating | Interest shift, data drift |
| Activity | A number of rated items for a user | Activity bias |
| Popularity | Number of users with a rating for an item | Popularity bias |

### 5.1 Experimental data

We perform evaluations on five public datasets—MovieLens 100K and 1M (movies) (Harper and Konstan 2015), BookCrossing (books) and two datasets published by Wan et al. (2020) for analyzing bias in marketing data for ModCloth (clothes) and electronics (based on Amazon review data). A summary of these datasets is presented in Table 5. We use explicit information about the user-item ratings for all datasets. For MovieLens, we use older versions of the dataset due to availability of both user and item features. We use the user, item, and context features to search for the potential biases as summarized in Table 4.

The continuous variables are discretized into five buckets based on the sample quantiles (the intervals for each feature are selected so that each bin contains 20% of examples). We set the significance level $\alpha = 0.01$ to detect the significant differences in error distribution. To avoid overfitting and selecting too specific rules that would be hard to interpret, we set the minimum number of samples in a leaf node to 1% of the sample size.

### 5.2 Detecting bias for different error metrics

In the first experiment, we apply the proposed method for evaluating a deep learning two-tower collaborative filtering recommendation algorithm (Yi et al. 2019) on five recommendation datasets from distinct domains with different error measures. The model consists of user and item embedding layers of size 64, a dense layer of 16 neurons with ReLU activation and a single output layer for predicting the user-item ratings. For each dataset, we randomly split the datasets into training and test subsets with 20% of ratings in the test set, and we train the model for 20 epochs with Adam optimizer and L2 regularization. Next, we build the bias detection tree model for each type of error and calculate the total bias (Eq. 15) for each dataset on the training and test sets. Next, we analyze the tree structure for the detected disparities to verify which attributes are associated with the biased results.

**Table 5** Datasets summary

| | MovieLens 100K | MovieLens 1M | BookCrossing | ModCloth | Electronics |
|---|---|---|---|---|---|
| Type of items | Movies | Movies | Books | Clothes | Electronics |
| #users | 943 | 6040 | 278,858 | 44,783 | 1,157,633 |
| #items | 1682 | 3900 | 271,379 | 1020 | 9560 |
| #ratings | 100,000 | 1,000,209 | 1,149,780 | 99,893 | 1,292,954 |
| Time span | 1997–1998 | 2000–2003 | 08-09.2004 | 2010–2019 | 1999–2018 |
| User metadata | Gender, age, occupation | | Age, location | Body shape | Gender |
| Item metadata | Title, genres, year | | Title, author, publisher, year | Category, size, brand | Category, gender in image |

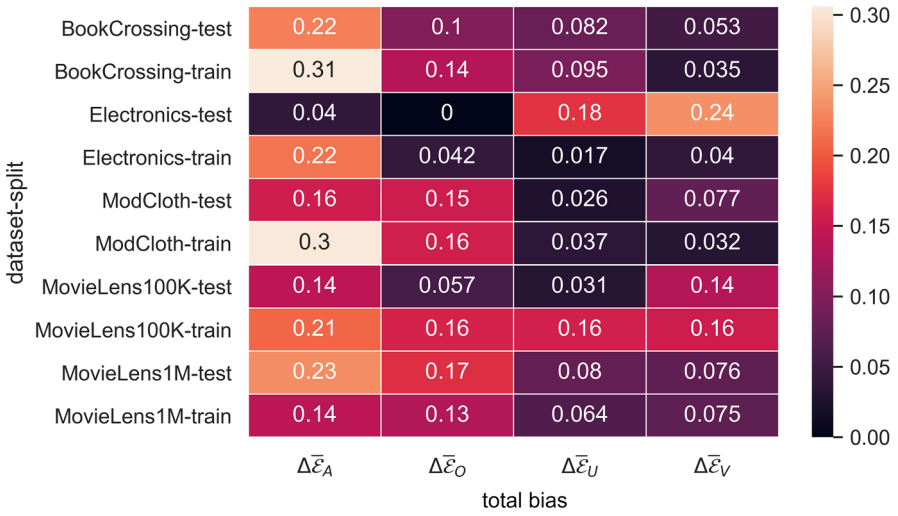


**Fig. 2** Results of total bias measures (*absolute, overestimation, underestimation and value* error) for each dataset split

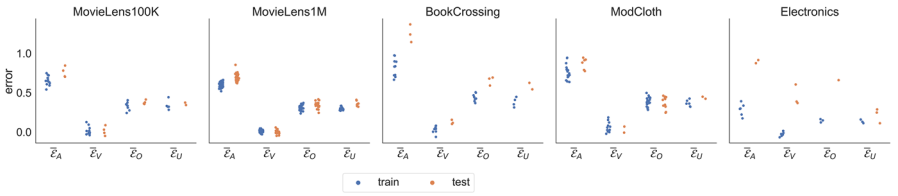


**Fig. 3** Error values in each detected leaf node for different error metrics and datasets

### 5.2.1 Results

Figure 2 presents the total bias for each metric and dataset calculated as an absolute difference between the maximum and minimum node value in the tree. We observe that the disparities in error distribution were detected for all the analyzed cases except overestimation error for the Electronics dataset, with the most severe difference in the absolute error for the BookCrossing training set.

Figure 3 shows the average error values $\overline{\mathcal{E}}(X)$ for the tree leaves for each dataset. It can be seen that for each dataset, there exist some nodes (combinations of attributes) for which the algorithm yields higher and lower errors. In particular, for BookCrossing and Electronics datasets, the absolute error for the nodes in the training set is significantly lower than on the test set, which indicates an overfitting problem for this case; however, the algorithm seems to overfit for some combinations of attributes more severely. For MovieLens datasets, there exist single nodes with significantly higher absolute error on the test set.

Table 6 presents the global error and results for the maximum ($\overline{\mathcal{E}}(X_{max})$) and minimum ($\overline{\mathcal{E}}(X_{min})$) detected combinations of parameters for different error metrics as well as the global average for all datasets. In some cases (in particular for the test

**Table 6** Global error and results for the maximum and minimum detected combinations of parameters for different error metrics

| Dataset | Metric | Global $\overline{\mathcal{E}}$ | | #leaves | | $\overline{\mathcal{E}}(X_{min})$ | | $\overline{\mathcal{E}}(X_{max})$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | Test | Train | Test | Train | Test | Train | Test | Train |
| BookCrossing | $\overline{\mathcal{E}}_A$ | 1.227 | 0.876 | 3 | 10 | 1.143 | 0.665 | 1.368 | 0.974 |
| | $\overline{\mathcal{E}}_O$ | 0.657 | 0.459 | 3 | 8 | 0.591 | 0.368 | 0.692 | 0.505 |
| | $\overline{\mathcal{E}}_U$ | 0.566 | 0.414 | 2 | 4 | 0.542 | 0.312 | 0.625 | 0.447 |
| | $\overline{\mathcal{E}}_V$ | 0.069 | 0.045 | 3 | 9 | 0.102 | −0.062 | 0.156 | 0.081 |
| Electronics | $\overline{\mathcal{E}}_A$ | 0.912 | 0.271 | 2 | 6 | 0.875 | 0.174 | 0.914 | 0.391 |
| | $\overline{\mathcal{E}}_O$ | 0.670 | 0.137 | 1 | 3 | 0.660 | 0.122 | 0.660 | 0.164 |
| | $\overline{\mathcal{E}}_U$ | 0.236 | 0.132 | 3 | 3 | 0.112 | 0.113 | 0.287 | 0.159 |
| | $\overline{\mathcal{E}}_V$ | 0.412 | 0.016 | 3 | 7 | 0.369 | −0.067 | 0.604 | 0.014 |
| ModCloth | $\overline{\mathcal{E}}_A$ | 0.838 | 0.775 | 6 | 17 | 0.774 | 0.636 | 0.944 | 0.943 |
| | $\overline{\mathcal{E}}_O$ | 0.431 | 0.398 | 12 | 21 | 0.246 | 0.282 | 0.461 | 0.496 |
| | $\overline{\mathcal{E}}_U$ | 0.403 | 0.378 | 2 | 6 | 0.423 | 0.321 | 0.449 | 0.425 |
| | $\overline{\mathcal{E}}_V$ | 0.011 | 0.010 | 2 | 12 | −0.008 | −0.022 | 0.070 | 0.185 |
| MovieLens100K | $\overline{\mathcal{E}}_A$ | 0.730 | 0.641 | 4 | 14 | 0.702 | 0.540 | 0.845 | 0.748 |
| | $\overline{\mathcal{E}}_O$ | 0.376 | 0.328 | 4 | 8 | 0.356 | 0.242 | 0.413 | 0.404 |
| | $\overline{\mathcal{E}}_U$ | 0.362 | 0.313 | 2 | 5 | 0.343 | 0.282 | 0.374 | 0.442 |
| | $\overline{\mathcal{E}}_V$ | 0.017 | 0.017 | 4 | 12 | −0.050 | −0.040 | 0.087 | 0.126 |
| MovieLens1M | $\overline{\mathcal{E}}_A$ | 0.694 | 0.599 | 23 | 30 | 0.620 | 0.518 | 0.853 | 0.661 |
| | $\overline{\mathcal{E}}_O$ | 0.352 | 0.305 | 18 | 29 | 0.244 | 0.236 | 0.415 | 0.370 |
| | $\overline{\mathcal{E}}_U$ | 0.342 | 0.293 | 8 | 16 | 0.321 | 0.271 | 0.409 | 0.334 |
| | $\overline{\mathcal{E}}_V$ | 0.013 | 0.012 | 21 | 29 | −0.048 | −0.030 | 0.058 | 0.045 |

set of BookCrossing), there is only one level of the bias tree, which means that there were no further significant splits, and the bias was detected based on a single attribute. However, in most cases, there exist multiple attributes that diversify the error distribution, and the mean value for these cases is significantly higher or lower than the global error.

The combinations of attributes associated with the highest and the lowest values in the tree nodes for absolute error are presented in Table 7. We can observe that for MovieLens 100K, the lowest average error on the training set (0.540) is detected for the male users and mystery movies produced before 1993, while the highest error (0.748) is associated with predictions for female users of medium activity level and particular age groups below 44 y.o.; hence, this group may be less satisfied with the recommendations. The most severe disparities on the test set concern the highest error for female users and older movies; however, the disparities are most severe in particular time ranges, which may indicate some seasonal trends or an interest shift.

For the MovieLens-1M dataset, the most significant difference for both training and the test sets is detected based on the movie production year and user age—the

predictions are more accurate for users above 25 y.o. and for movies produced before 1979, and less accurate for more recent productions and adolescent users. Interestingly, we do not observe significant biases related to user gender for this newer dataset which was present in ML-100K. Hence, the models trained on the older version of MovieLens data might propagate the historical stereotypes.

For BookCrossing, the most significant bias is associated with the activity and popularity features. Interestingly, we observe that these discrimination rules are different for the training and the test sets—for instance, the least active users have the smallest training error while they have the largest error on the test set. Clearly, these are the cases when a model overfits on some groups from the training set (cold-start users in this case), and the predictions on the test set are less accurate.

The analysis of the most extreme values for the marketing bias datasets indicates that the algorithm may propagate some additional sources of bias than those considered by Wan et al. (2020). The tree structures for an absolute error on the training set of these datasets are presented in Fig. 4. In particular, for the clothes data, the bias may be related to particular years of production, item categories, and brands apart from the sizing. For the electronics data, we observe that the training error is the highest for the computers and headphones products and non-male gender while the item popularity has the largest impact on the error in the test set.

### 5.3 Model selection with Lambda-Minimax Debiasing

In the second experiment, we analyze the trade-off between the global and the worst-case (minimax) optimizations during the model selection process. To this end, we apply the BDT method to analyze the *absolute error* $\Delta_{bias}\mathcal{E}_{\mathcal{A}}$ for five different collaborative filtering recommendation algorithms: Slope One (Li et al. 2012), k-nearest neighbors (KNN), Co-Clustering (George and Merugu 2005), Non-Negative Matrix Factorization (NMF) (Lee and Seung 1999), and Singular Value Decomposition (SVD) (Salakhutdinov and Mnih 2008). To ensure that our results can be reproduced, we use publicly available open-source implementation of the CF algorithms (Hug 2020).

Each of the selected recommendation algorithms is trained to predict ratings from the training set. We use a fivefold cross-validation procedure for selecting the hyper-parameters of each recommendation model with the parameter grid defined in Table 8, and a 20% hold-out test set. First, the hyper-parameters are selected on the cross-validated training set, and then the model is re-trained on the whole training set. To filter out the outliers, only the items with at least 5 ratings are considered in CF model training for this experiment. We report the recommendation model results for the test sets with respect to global mean absolute error ($\overline{\mathcal{E}_A}$) and the most extreme leaf values (maximum $\overline{\mathcal{E}_A}(X_{max})$ and minimum $\overline{\mathcal{E}_A}(X_{min})$ average errors) are analyzed.

Then, we perform the model selection considering the most discriminated node in objective function as described in Definition 22. We compare the results for $\lambda_{bias} = 0$ and $\lambda_{bias} = 1$ to analyze the global-worst-case optimization trade-off. Next, we analyze the results for varying $\lambda_{bias}$ values.

**Table 7** The most biased leaf rules for each CF model

| Dataset | Split | $X_{min}$ | $\overline{\mathcal{E}}_A(X_{min})$ | $X_{max}$ | $\overline{\mathcal{E}}_A(X_{max})$ |
|---|---|---|---|---|---|
| ML-100K | Train | (gender:M), (year < 1993), (genre:mystery) | 0.540 | (gender:F), (user act.:74-143), (age < 23, 33 − 44) | 0.748 |
| | Test | (year < 1993), (date:1997.11-1998.01, 1998.03-04) | 0.702 | (year < 1993), (date:1997.09-11,1998.01-03), (gender:F) | 0.845 |
| ML-1M | Train | (year < 1979), (age: 25-56),(user act. < 121) | 0.518 | (year:1997-2000), (user act. > 121) (age< 18) | 0.661 |
| | Test | (year < 1979), (item pop. > 264), (age:25-50) | 0.620 | (year:1997-2000), (item pop. < 264), (user act. < 232) | 0.853 |
| BC | Train | (item pop. < 14), (user act. < 20) | 0.665 | (item pop. > 40), (user act. > 40), (age < 26) | 0.971 |
| | Test | (user act. > 73) | 1.143 | (user act.: < 19) | 1.368 |
| MC | Train | (size < 2), (category:bottoms, outwear, tops), (item pop. > 236), (brand:ModCloth) | 0.636 | (size < 2), (category:dresses), (year > 2009), (model: small) | 0.940 |
| | Test | (size < 4), (year:2009-2014) | 0.788 | (size > 4),(model:small), (user act. < 18) | 0.944 |
| EL | Train | (year:1998-2011), (category:other), (item pop. < 355) | 0.174 | (year:1998-2011), (category:computers&accessories, headphones), (gender:F,F&M) | 0.391 |
| | Test | (item pop > 355, < 50) | 0.875 | (item pop. < 355) | 0.914 |

```
root=[]: 0.271
├── year_bucketized=[(1998, 2011]]: 0.35
│   ├── category=['Computers & Accessories', 'Headphones']: 0.429
│   │   ├── model_attr=['Female', 'Female&Male']: 0.501
│   │   └── model_attr=['Male']: 0.372
│   └── category=['category:other']: 0.244
│       ├── item popularity_bucketized=['(1007, 28530]', '(355, 1007]', '(50, 149]']: 0.298
│       └── item popularity_bucketized=['(149, 355]', '(9, 50]']: 0.155
└── year_bucketized=['(2011, 2013]', '(2013, 2014]', '(2014, 2015]', 'year_bucketized:other']: 0.247
    ├── item popularity_bucketized=['(1007, 28530]', '(355, 1007]', '(149, 355]']: 0.284
    └── item popularity_bucketized=['(50, 149]', '(9, 50]']: 0.196


root=[]: 0.775
├── size=['']: 0.812
│   ├── category=['Bottoms', 'Dresses', 'Outerwear']: 0.827
│   └── category=['Tops']: 0.787
│       ├── year_bucketized=['(2009, 2014]']: 0.699
│       └── year_bucketized=['(2014, 2015]', '(2018, 2019]', '(2016, 2018]', '(2015, 2016]']: 0.807
├── size=[1, 2]: 0.729
│   ├── category=['Bottoms', 'Outerwear', 'Tops']: 0.735
│   │   ├── item popularity_bucketized=['(236, 478]', '(897, 1887]', '(478, 897]']: 0.721
│   │   └── item popularity_bucketized=['(71, 236]', '(9, 71]']: 0.774
│   └── category=['Dresses']: 0.717
│       ├── year_bucketized=['(2009, 2014]', '(2018, 2019]']: 0.777
│       └── year_bucketized=['(2014, 2015]', '(2015, 2016]', '(2016, 2018]']: 0.671
└── size=[3, 4, 5, 6, 7]: 0.774
    ├── item popularity_bucketized=['(236, 478]', '(71, 236]', '(9, 71]']: 0.793
    │   ├── category=['Bottoms', 'Outerwear', 'Dresses']: 0.83
    │   └── category=['Tops']: 0.734
    └── item popularity_bucketized=['(478, 897]', '(897, 1887]']: 0.749
        ├── brand=['', 'Chi Chi London']: 0.767
        └── brand=['ModCloth', 'Retrolicious', 'brand:unpopular']: 0.658
```

**Fig. 4** Bias detection tree structures for the absolute error of on the train set of Electronics (upper) and ModCloth (lower) datasets

**Table 8** Recommendation algorithms with corresponding hyper-parameter grids

| Algorithm | Hyper-parameter grid |
|---|---|
| Slope One | – |
| KNN | K: [10, 20, 50, 100], user-based: [True, False] |
| Co-Clustering | *n* user clusters: [5, 10, 20, 50, 100], *n* item clusters: [5, 10, 20, 50, 100] |
| NMF | *n* factors: [10, 20, 50, 100], biased: [True, False] |
| SVD | *n* factors: [10, 20, 50, 100], biased: [True, False] |

### 5.3.1 Results

Table 9 and Fig. 5 present the global $\overline{\mathcal{E}_A}$, $\overline{\mathcal{E}_A}(X_{max})$ and minimum $\overline{\mathcal{E}_A}(X_{min})$ absolute error detected by the BDT algorithm for different recommendation models. It can be observed from these results that in some cases the choice of global-optimal model is different than the one with the lowest worst-error—for example, for Movielens 100K, KNN results in the smallest global error (0.725) while SVD yields a lower worst-case error (0.819 vs. 0.869 for KNN).

Figure 6 shows an example of the constructed bias tree (with the maximum depth limited to 2 for better readability), and the test error distributions for each detected leaf for the KNN model on Movielens 100K dataset (selected according to the global

**Table 9** Global $\overline{\mathcal{E}_A}$ and results for the maximum $\overline{\mathcal{E}_A}(X_{max})$ and minimum $\overline{\mathcal{E}_A}(X_{min})$ detected tree nodes for the compared CF algorithms on five datasets. The results in bold are the lowest error values for each dataset

| Dataset | Model | Global $\overline{\mathcal{E}_A}$ | #leaves | $\overline{\mathcal{E}_A}(X_{min})$ | $\overline{\mathcal{E}_A}(X_{max})$ |
|---|---|---|---|---|---|
| BookCrossing | CoClustering | 1.292 | 7 | 1.097 | 1.468 |
| | KNN | 1.295 | 9 | 1.092 | 1.447 |
| | NMF | 1.437 | 3 | 1.359 | 1.502 |
| | SVD | **1.217** | 5 | **1.074** | **1.338** |
| | SlopeOne | 1.395 | 8 | 1.117 | 1.507 |
| Electronics | CoClustering | 0.810 | 3 | 0.695 | 0.884 |
| | KNN | 0.851 | 2 | 0.819 | 0.873 |
| | NMF | 0.793 | 4 | 0.594 | **0.831** |
| | SVD | **0.787** | 6 | **0.573** | 0.899 |
| | SlopeOne | 0.830 | 4 | 0.692 | 0.912 |
| ModCloth | CoClustering | 0.845 | 6 | 0.767 | 0.962 |
| | KNN | 0.831 | 8 | 0.740 | 0.933 |
| | NMF | **0.772** | 4 | **0.639** | **0.836** |
| | SVD | 0.819 | 6 | 0.689 | 0.910 |
| | SlopeOne | 0.844 | 8 | 0.735 | 0.967 |
| MovieLens100K | CoClustering | 0.747 | 7 | 0.666 | 0.871 |
| | KNN | **0.725** | 8 | **0.655** | 0.869 |
| | NMF | 0.775 | 9 | 0.682 | 0.902 |
| | SVD | 0.736 | 7 | 0.671 | **0.819** |
| | SlopeOne | 0.735 | 7 | 0.678 | 0.875 |
| MovieLens1M | CoClustering | 0.710 | 15 | 0.657 | 0.774 |
| | KNN | 0.693 | 15 | **0.613** | 0.792 |
| | NMF | 0.755 | 15 | 0.702 | 0.854 |
| | SVD | **0.684** | 13 | 0.632 | **0.771** |
| | SlopeOne | 0.713 | 16 | 0.649 | 0.809 |

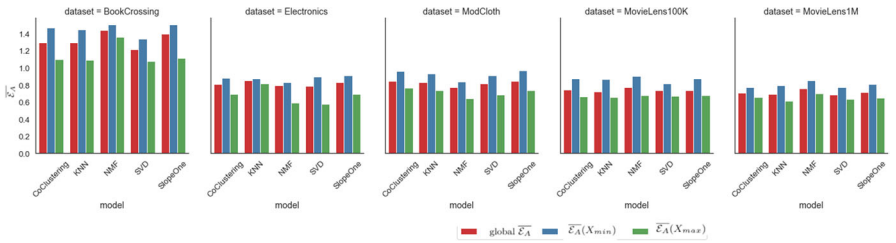


**Fig. 5** Global $\overline{\mathcal{E}_A}$, $\overline{\mathcal{E}_A}(X_{max})$ and minimum $\overline{\mathcal{E}_A}(X_{min})$ absolute error for different recommendation algorithms and datasets

objective). The main difference is detected with respect to the movies' production year so that the error is larger for the latest movies (produced after 1993)—0.758 vs. 0.684 for the old ones. For newer movies, the error is larger for female users; hence,

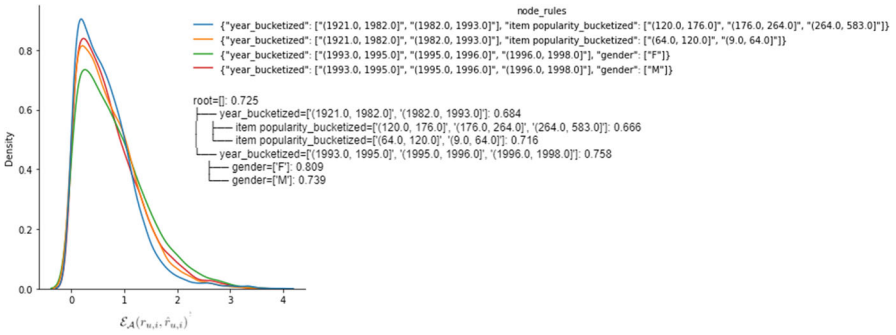


**Fig. 6** Distribution of the test error for detected tree leaves with the maximum depth limited to 2, and tree structures for the KNN algorithm on Movielens 100K dataset. The probability density ($Y$-axis) of the absolute error of the model ($X$- axis) is estimated with KDE

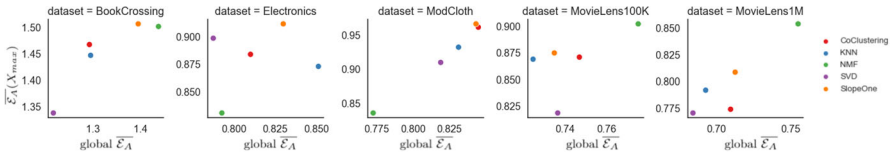


**Fig. 7** Global and maximum node error $\mathcal{E}_A$ for each CF model on five datasets

this group may be less satisfied with the recommendations. However, for the older movies, we observe a popularity bias as the error is higher for less popular items. The most biased combination of attributes in this case is female gender and new movies. Accordingly, the λ-minimax approach will be applied to reduce the error for these cases.

Figure 7 shows a visualization of the trade-off between the global $\mathcal{E}_A$ ($X$-axis) and maximum error $\mathcal{E}_{max}$ ($Y$-axis) for the CF models on each dataset. It can be observed that the optimal model selected with global ($\lambda_{bias} = 0$) and minimax ($\lambda_{bias} = 1$) objective for three datasets are equal (SVD for BookCrossing and MovieLens 1M, NMF for ModCloth). However, for Electronics and MovieLens, the choice is dependent on the objective (global SVD versus minimax NMF for Electronics, KNN vs. SVD for MovieLens 100K, respectively). From the comparison in Table 10, it can be observed that by applying minimax optimization on these datasets, it is possible to significantly reduce the maximum node error while retaining an acceptable global error. For MovieLens, SVD results in $-5.8\%$ lower maximum node error while the global error increases by $1.6\%$ compared to the globally optimal KNN. For Electronics, the NMF (minimax objective) reduces the maximum error by $7.5\%$ compared to SVD (global objective), while the global error is increased by $1.6\%$. Figure 8 presents the results of the relaxed version according to Definition 22. In the case of Electronics, NMF would be selected for the positive $\lambda_{\text{bias}}$ values, same as SVD for MovieLens 100K. We can observe that the loss for these datasets grows much faster for other types of models, which suggests that the selected models provide the most stable trade-off between global and worst-case optimization.

**Table 10** Best models selected based on global ($\lambda_{bias} = 0$) and minimax ($\lambda_{bias} = 1$) objective considering $\mathcal{E}_A$ for each dataset

| Dataset | $\lambda_{bias}$ | Model | $\overline{\mathcal{E}_A}$ | $\overline{\mathcal{E}_A}_{X_{max}}$ | $\Delta(\overline{\mathcal{E}_A})$ | $\Delta(\overline{\mathcal{E}_A}_{X_{max}})$ |
|---|---|---|---|---|---|---|
| MovieLens1M | 0 | SVD | 0.684 | 0.771 | – | – |
| | 1 | SVD | 0.684 | 0.771 | | |
| MovieLens100K | 0 | KNN | 0.725 | 0.869 | +1.6% | −5.8% |
| | 1 | SVD | 0.736 | 0.819 | | |
| ModCloth | 0 | NMF | 0.772 | 0.836 | – | – |
| | 1 | NMF | 0.772 | 0.836 | | |
| Electronics | 0 | SVD | 0.787 | 0.899 | +0.7% | −7.5% |
| | 1 | NMF | 0.793 | 0.831 | | |
| BookCrossing | 0 | SVD | 1.217 | 1.338 | – | – |
| | 1 | SVD | 1.217 | 1.338 | | |

We report the percentage difference for $\lambda = 0$ and 1 in terms of global and maximum node error when the fair-optimization is applied

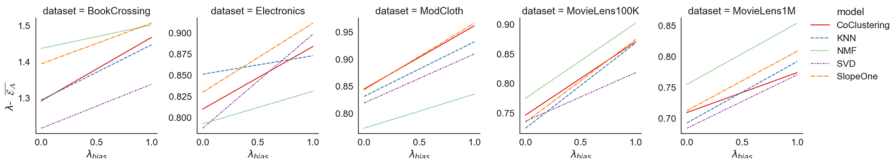


**Fig. 8** $\lambda$-weighted $\mathcal{E}_A$ for varying $\lambda_{bias}$ parameter for each CF model on five datasets

We suspect that for the other three datasets the detected bias may not be related to the model architecture itself, and an additional analysis is needed to identify the source of disparities and indicate potential remedial actions. Additionally, we would like to note that in this experiment, we performed a model selection from a set of CF models that were trained on the same set of user-item ratings. This process could be further improved by incorporating other types of models, such as content-based recommendations, which might be more efficient for addressing other types of biases (as shown in Ghazanfar and Prügel-Bennett 2014 for grey sheep users).

## 6 Results summary and discussion

Based on the results presented in this paper, we can make the following conclusions:

– Some significant disparities in model performance for distinct groups of attributes were detected for almost all of the analyzed recommendation scenarios. From these results, we can conclude that while different recommendation algorithms may yield comparable results when averaged globally, the error distributions may differ dramatically for different combinations of attributes.
– Our proposed BDT method identified severe disparities for certain feature combinations that are missed by the single-attribute approaches most often used for analyzing recommender fairness. By analyzing the errors on the train and test sets,

we could identify situations when a model under- or overfits for certain types of inputs. Moreover, by comparing the biases for two versions of the movies dataset, we demonstrated that the older dataset may contain gender biases which may lead to propagating stereotypical recommendations.
– For two datasets, it was possible to improve the recommendation performance for the worst-case combinations of attributes by incorporating the proposed Lambda-Minimax debiasing method in the model selection process. By balancing between the global and pessimistic optimization trade-off, the error was reduced for the biased cases without any significant loss in global performance.

We note that the analysis could be extended by incorporating other types of models that use different features (such as content-based) to verify other disparities. Moreover, the information about the combinations of attributes associated with biases detected by the BDT technique may be used to automate the design of hybrid recommenders.

### 6.1 Limitations

There are some limitations of the proposed method, which are discussed below.

#### 6.1.1 Metadata availability

The quality of detected disparities is highly dependent on the availability of attributes. Hence, there may exist some disparities that were not detected if the provided metadata is not sufficient. However, to the best of our knowledge, all of the existing bias detection approaches also require that the attributes are known (Sánchez and Bellogín 2019; Beutel et al. 2017, 2019; Wan et al. 2020; Yao and Huang 2017; Kershaw et al. 2021; Wei et al. 2021; Zhu et al. 2021).

#### 6.1.2 Computational complexity

The computational complexity may be large for a high number of attributes and categories. However, we included a detailed worst-case analysis and provide practical recommendations for selecting the algorithm parameters in "Appendix A" to ensure that the method works fast for practical cases.

#### 6.1.3 Lack of causality

While the use of post hoc bias detector and model-agnostic approaches is flexible enough to handle different types of models, it does not truly explain the reasons for the bias and shows the correlations, and descriptive statistics of features with the output rather than the causation of phenomena (Rudin 2019). However, based on the results presented in this paper, we conclude that it may constitute a useful and universal tool to suggest the directions for further investigating the reasons for the bias and reducing the disparities.

## 7 Conclusion

In this paper, we proposed a model-agnostic bias detection tree method to detect systematic biases in a recommendation model's performance. Our proposed technique identifies combinations of attributes for which the recommendations are significantly worse than in other cases without a priori knowledge about the protected categories.

In the experimental evaluation, we showed that our proposed method can detect different types of biases by analyzing four kinds of unfairness criteria for collaborative filtering algorithms. The results on five real-world recommendation datasets from different domains show that our proposed method can identify severe disparities for certain feature combinations that are missed by the single-attribute approaches most often used for analyzing recommender fairness. Additionally, we proposed a Lambda-Minimax debiasing technique to improve the functioning for the worst-case combinations of attributes, while controlling the global optimization trade-off. The experimental results showed that this method, applied on the top of groups retrieved from BDT, enables significant reduction of the worst-case error for two datasets without any significant loss in global accuracy.

In future, we plan to perform experiments with other types of recommendation algorithms such as content-based methods to analyze potential algorithmic biases in other recommendation settings.

Another area of research that could be further investigated is related to exploring different types of attributes that may be subject to bias. Since the disparity explanations are generated based on additional metadata, their quality is dependent on the availability of this information. Additional features may incorporate contextual information related to presentation biases. Moreover, the proposed methods can be extended to analyze the bias related to ranking quality.

## Appendix A: Computational complexity

In this section, we perform an analysis of the computational complexity of the proposed algorithm, and the execution time is measured for experimental data.

### Appendix A.1: Theoretical analysis

In the complexity analysis, let $L$ denote the number of attributes, $N$ is the number of ratings and $D$ is the maximum depth of the tree ($1 \leq D \leq L$). Next, let $n_k$ be the number of subsets of $T_k$ with significant difference in the error metric distribution ($p < \alpha_{merge}$). Without loss of generality, let us assume that the number of attribute values $|T_k|$ and $n_k$ is equal for all attributes $k$:

$$|T_{k_1}| = |T_{k_2}| = \cdots = |T_{k_L}| = K, n_{k_1} = n_{k_2} = \cdots = n_{k_L} = n_k \tag{23}$$

Then $K - n_k$ is the number of iterations in the merge phase for an attribute $k$. Since the complexity of the split-phase is linear $O(L)$, we only consider the complexity of

the merge phase in the analysis. Assuming that the complexity of operations in each node is equal (Eq. 23), the overall complexity of BDT algorithm can be calculated as:

$$A(L, K) = \sum_{p \in P} F(p) = |P| F(p) \tag{24}$$

where $F(p)$ denotes the complexity of the merge phase for a non-leaf tree node where the split of this node is performed based on attribute $x_k$ with values $T_k = \{k_1, \ldots k_K\}$ and $P$ denotes the set of all parent nodes in the tree.

In each iteration of the merge phase, the significance test is calculated for all pairs of attribute values from $T_k$ for each $k$. Next, one pair $k_m, k_n$ with the least significant difference (if $p > \alpha_{merge}$) will be merged. Hence, the algorithm will make $K$ comparisons of single values in the first step ($i = 1$) and the pair $k_{m,n}$ will be merged such that the number of values that will be processed in $i = 2$ is $|T_k^1| = |T_k| - 1 = K - 1$. Consequently, $K_i = K - (i - 1)$ values will be compared in the $i$-th iteration, until no further merges can be performed ($p < \alpha_{merge}$ for all combinations). The difference in error distribution is tested for each pair of attribute values, resulting in $K_i^2$ operations in each iteration. Accordingly, the complexity of the merge stage for one attribute in a node is given by:

$$\sum_{i=1}^{K-n_k} (K - (i - 1))^2 \tag{25}$$

The number of nodes on tree level $d$, $1 \leq d \leq D$ is equal to $n_k^{d-1}$ and the number of non-leaf nodes in the tree of depth $D$ can be calculated as:

$$|P| = \sum_{d=1}^{D} n_k^{d-1} = n_k^0 + n_k^1 + \cdots n_k^{D-1} \tag{26}$$

For a simple mean-based significance test, the comparison of the error metric $\sigma_k^2$ between two attribute values $k_i, k_j \in T_k$ requires $N$ operations, where $N$ is the number of samples.

Since the number of attributes available for a node on level $d$ is $L - d + 1$, the number of operations for the merge stage in one node on level $d$ is given by (From Formula 25):

$$F(p) = N(L - d + 1) \sum_{i=1}^{K-n_k} (K - (i - 1))^2 \tag{27}$$

Hence, the overall general complexity can be bounded by (based on Formulas 27, 26):

$$A(L, K) = \sum_{d=1}^{D} n_k^{d-1} F(p) = O(n_k^{D-2} L K^3 N) \tag{28}$$

To be more precise, let us consider the computational complexity of the BDT method in the following most extreme cases:

1. *Optimistic case*
   In the optimistic case, there are no significant differences in the error metric distribution among the parameter values for any attribute (as in the first simulated example):

$$\forall_{k\in 1\ldots L, k_i, k_j \in T_k} : p_{k_i,k_j} > \alpha_{merge}, n_k = 1$$

   In this case, in the merge phase, the values $k_i$ will be merged successively until all values are merged into one set $T_k^* = \{k_1, \ldots k_K\}$. Since there are no significant differences between the attribute values, no further splits will be made and the algorithm stops after the merge phase ($D = 1$) and the overall complexity will be (Eq. 27):

$$A(L, K) = LN \sum_{i=1}^{K-1} (K - (i-1))^2 = O(LNK^3) \tag{29}$$

2. *Pessimistic case*
   In the worst case, all pairs of parameter values have significant differences in error metric distribution:

$$\forall_{k\in 1\ldots L, k_i, k_j \in T_k} : p_{k_i,k_j} < \alpha_{merge}$$

   In this case, there is only one iteration in the merge phase ($n_k = K$) as no values are merged together but the depth of the tree is equal to the number of attributes ($D = L$). Then, the complexity of merge phase of one node is (Eq. 27):

$$F(P) = N(L - d + 1)K^2 \tag{30}$$

   And the overall complexity can be restricted by:

$$\begin{aligned} A(L, K) &= N \sum_{d=1}^{L} K^{d-1}(L - d + 1)K^2 \\ &= N \sum_{d=1}^{L} K^{d+1}(L - d + 1) = O(NLK^{L+2}) \end{aligned} \tag{31}$$

### Appendix A.2: Experimental computation time

To verify the practical complexity of the proposed algorithm, we performed an experiment on synthetic data generated in an analogous way as in Sect. 4.3; however, instead of a pre-defined set of attributes, we compared the result for varying parameters of the data characteristics (with $\alpha = 0.01$):

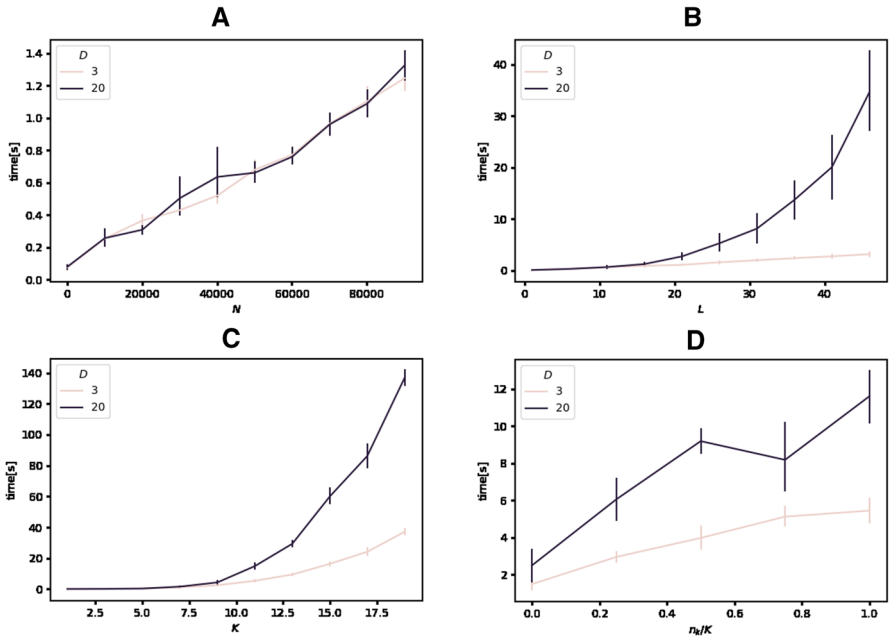


**Fig. 9** Execution time of the BDT algorithm depending on different parameters: **a** Number of examples; **b** Number of attributes; **c** Number of categories in each attribute; **d** Ratio of categories with a significant difference in the error metric distribution. For each case, we compared the time for a maximum depth limited to 3 and 20 levels

- $N$—the number of examples varies between 100 and 100.000 (default 10.000),
- $L$—the number of attributes between 1 and 50 (default 5),
- $K$—the number of categories per attribute between 2 and 20 (default 4),
- $n_k$—ratio of categories with a different error metric distribution 0–1 (default 0.5),
- $D$—maximum depth of the tree (3 or 20).

The computation times were calculated for a single machine (CPU Intel Core i5-4210U, 1.7 GHz), and the results were averaged over ten iterations.

*Results* The results for each parameter are presented in Fig. 9 for maximum depth limited to 3 and 20 levels. The correlation between the computation time and the number of examples is linear and does not depend on $D$ (Fig. 9a) This shows that our proposed method scales well in terms of dataset size. Figure 9b shows that the relationship between the time and the number of attributes $L$ for the two tree depths ($D$) we investigated does not differ significantly for small $L$; but, the difference starts increasing for $L > 20$. This observation indicates that restricting the depth of the tree may be important, especially when the number of attributes is large. A similar observation holds for the number of attribute categories $K$ (Fig. 9c) and the ratio of categories with a significant difference in the error metric distribution (Fig. 9d). Moreover, as shown in Fig. 9d, the computation time is the lowest in the optimistic case when no categories have a significantly different error metric distribution. Importantly, it is worth noticing that the complexity is reduced significantly after limiting the

maximum depth of the tree. Since for practical usage in detecting disparities, a large complexity may lead to difficulties in interpreting the model results, this parameter should be limited for the sake of optimization.

### Appendix A.3: Limiting computation time

In practice, the number of splits for each node is usually significantly lower than $K$ unless there are significant differences between the error metric distribution in all nodes. However, since the pessimistic computational complexity can be large, some steps can be undertaken to limit the complexity of the algorithm for practical applications:

- Limiting $L$—to reduce the number of attributes, standard feature extraction techniques may be applied to remove the correlations and group attributes into fewer categories. For instance, the movie tags could be grouped into several categories with a topic modeling technique. Additional optimization steps may include subsampling techniques and training multiple trees on subsets of the features or examples.
- Limiting $K$—may be achieved, for instance, by bucketizing the numerical attributes (such as the user's age or movie production year), removing the outliers, or filtering the less frequent values.
- Limiting $D$—to reduce the complexity and avoid selecting too specific rules that would be hard to interpret, the tree can be regularized by limiting the maximum depth. Then, parameter $D$ will be constant.
- Limiting $n_k$—the number of splits in each node can be controlled by the attributes $\alpha$ that define the statistical significance threshold for the merge or split stages or by setting the minimum number of samples in a node.

## References

Abdollahpouri, H., Burke, R., Mobasher, B.: Controlling popularity bias in learning-to-rank recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys'17, pp 42–46. Association for Computing Machinery, New York (2017)

Anelli, V.W., Di Noia, T., Di Sciascio, E., Ragone, A., Trotta, J.: Local popularity and time in top-n recommendation. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) Advances in Information Retrieval, pp. 861–868. Springer, Cham (2019)

Baeza-Yates, R.: Bias on the web. Commun. ACM **61**, 54–61 (2018)

Barocas, S., Hardt, M., Narayanan, A.: Fairness and Machine Learning. fairmlbook.org (2019). http://www.fairmlbook.org

Beutel, A., Chi, E.H., Cheng, Z., Pham, H., Anderson, J.: Beyond globally optimal: Focused learning for improved recommendations. In: Proceedings of the 26th International Conference on World Wide Web, WWW'17, pp. 203–212, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee

Beutel, A., Chen, J., Doshi, T., Qian, H., Wei, L., Wu, Y., Heldt, L., Zhao, Z., Hong, L., Chi, E.H., Goodrow, C.: Fairness in recommendation ranking through pairwise comparisons. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD'19, pp. 2212–2220. Association for Computing Machinery, New York (2019)

Boratto, L., Fenu, G., Marras, M.: The effect of algorithmic bias on recommender systems for massive open online courses. In: Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds.) Advances in Information Retrieval, pp. 457–472. Springer, Cham (2019)

Boratto, L., Fenu, G., Marras, M.: Connecting user and item perspectives in popularity debiasing for collaborative recommendation. Inf. Process. Manag. **58**(1), 102387 (2021)

Brown, M.B., Forsythe, A.B.: Robust tests for the equality of variances. J. Am. Stat. Assoc. **69**(346), 364–367 (1974)

Burke, R., Sonboli, N., Ordonez-Gauger, A.: Balanced neighborhoods for multi-sided fairness in recommendation. In: Friedler, S.A., Wilson, C. (eds.) Proceedings of the 1st Conference on Fairness, Accountability and Transparency, Proceedings of Machine Learning Research, vol. 81, pp. 202–214. PMLR, New York (2018)

Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., He, X.: Bias and debias in recommender system: a survey and future directions. CoRR. arXiv:2010.03240 (2020)

Diana, E., Gill, W., Kearns, M., Kenthapadi, K., Roth, A.: Convergent algorithms for (relaxed) minimax fairness. CoRR. arXiv:2011.03108 (2020)

Eskandanian, F., Sonboli, N., Mobasher, B.: Power of the few: analyzing the impact of influential users in collaborative recommender systems. In: Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization, UMAP'19, pp. 225–233. Association for Computing Machinery, New York (2019)

Fryer, R., Loury, G., Yuret, T.: An economic analysis of color-blind affirmative action. J. Law Econ. Organ. **24**(2), 319–355 (2008)

Gajane, P., Pechenizkiy, M.: On formalizing fairness in prediction with machine learning (2017)

George, T., Merugu, S.: A scalable collaborative filtering framework based on co-clustering. In: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM'05, pp. 625–628. IEEE Computer Society, USA (2005)

Ghazanfar, M.A., Prügel-Bennett, A.: Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. Expert Syst. Appl. **41**(7), 3261–3275 (2014)

Guidotti, R., Monreale, A., Turini, F., Pedreschi, D., Giannotti, F.: A survey of methods for explaining black box models. CoRR. arXiv:1802.01933 (2018)

Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. ACM Trans. Interact. Intell. Syst. **5**(4), 19:1–19:19 (2015)

Hug, N.: Surprise: a python library for recommender systems. J. Open Source Softw. **5**(52), 2174 (2020)

Kamishima, T., Akaho, S., Sakuma, J.: Fairness-aware learning through regularization approach. In: 2011 IEEE 11th International Conference on Data Mining Workshops, pp. 643–650 (2011)

Kass, G.V.: An exploratory technique for investigating large quantities of categorical data. J. R. Stat. Soc. Ser. C Appl. Stat. **29**, 119–127 (1980)

Kershaw, D.J., Koeling, R., Bourgeois, S., Trenta, A., Muncey, H.J.: Fairness in Reviewer Recommendations at Elsevier, pp. 554–555. Association for Computing Machinery, New York (2021)

Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)

Lee, D.D., Seung, H.S.: Learning the parts of objects by nonnegative matrix factorization. Nature **401**, 788–791 (1999)

Legislation E.U.: Equal treatment of persons (2009)

Li, J., Sun, L., Wang, J.: A slope one collaborative filtering recommendation algorithm using uncertain neighbors optimizing. In: Wang, L., Jiang, J., Lu, J., Hong, L., Liu, B. (eds.) Web-Age Information Management, pp. 160–166. Springer, Berlin (2012)

Li, Y., Chen, H., Fu, Z., Ge, Y., Zhang, Y.: User-Oriented Fairness in Recommendation, pp. 624–632. Association for Computing Machinery, New York (2021)

Lippert-Rasmussen, K.: Born Free and Equal?: A Philosophical Inquiry Into the Nature of Discrimination. Oxford University Press, Oxford (2013)

McCrae, J., Piatek, A., Langley, A.: Collaborative filtering. http://www.imperialviolet.org (2004)

Misztal-Radecka, J., Indurkhya, B.: When is a recommendation model wrong? A model-agnostic tree-based approach to detecting biases in recommendations. In: Boratto, L., Faralli, S., Marras, M., Stilo, G. (eds.) Advances in Bias and Fairness in Information Retrieval, pp. 92–105. Springer, Cham (2021)

Olteanu, A., Castillo, C., Diaz, F., Kıcıman, E.: Social data: biases, methodological pitfalls, and ethical boundaries. Front. Big Data **2**, 13 (2019)

Ribeiro, M.T., Singh, S., Guestrin, C.: Model-agnostic interpretability of machine learning (2016)

Ricci, F., Rokach, L., Shapira, B.: Recommender Systems: Introduction and Challenges, pp. 1–34. Springer US, Boston (2015)
Ritschard, G.: CHAID and Earlier Supervised Tree Methods, pp. 48–74. J.J. McArdle & G. Ritschard, Routeledge, New York (2013)
Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)
Salakhutdinov, R.. Mnih, A.: Probabilistic matrix factorization. In: Advances in Neural Information Processing Systems, vol. 20 (2008)
Sánchez, P., Bellogín, A.: Attribute-based evaluation for recommender systems: incorporating user and item attributes in evaluation metrics. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys'19, pp. 378–382. Association for Computing Machinery, New York (2019)
Singh, J., Anand, A.: Posthoc interpretability of learning to rank models using secondary training data. CoRR. arXiv:1806.11330 (2018)
Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. (2009)
Tintarev, N., Masthoff, J.: Designing and evaluating explanations for recommender systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 479–510. Springer US, Boston (2011)
Tsintzou, V., Pitoura, E., Tsaparas, P.: Bias disparity in recommendation systems. CoRR. arXiv:1811.01461 (2018)
Wan, M., Ni, J., Misra, R., McAuley, J.: Addressing marketing bias in product recommendations. In: Proceedings of the 13th International Conference on Web Search and Data Mining, WSDM'20, pp. 618–626. Association for Computing Machinery, New York (2020)
Wei, T., Feng, F., Chen, J., Wu, Z., Yi, J., He, X.: Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system, pp. 1791–1800. Association for Computing Machinery, New York (2021)
Yao, S., Huang, B.: Beyond parity: fairness objectives for collaborative filtering. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 2921–2930. Curran Associates, Inc. (2017)
Yi, X., Yang, J., Hong, L., Cheng, D.Z., Heldt, L., Kumthekar, A.A., Zhao, Z., Wei, L., Chi, E. (eds.) Sampling-Bias-Corrected Neural Modeling for Large Corpus Item Recommendations (2019)
Zehlike, M., Bonchi, F., Castillo, C., Hajian, S., Megahed, M., Baeza-Yates, R.: Fa*ir: a fair top-k ranking algorithm. CoRR. arXiv:1706.06368 (2017)
Zehlike, M., Castillo, C.: Reducing disparate exposure in ranking: a learning to rank approach. In: Proceedings of The Web Conference 2020, WWW'20, pp. 2849–2855. Association for Computing Machinery, New York (2020)
Zhang, Y., Chen, X.: Explainable recommendation: a survey and new perspectives. CoRR. arXiv:1804.11192 (2018)
Zhu, Z., He, Y., Zhao, X., Caverlee, J.: Popularity Bias in Dynamic Recommendation, pp. 2439–2449. Association for Computing Machinery, New York (2021)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Joanna Misztal-Radecka** obtained her Ph.D. degree in Computer Science from the Faculty of Computer Science, Electronics and Telecommunications at AGH University of Science and Technology in 2022. She conducted research on algorithmic bias detection and explainability of machine learning models, in particular for the applications of recommender systems. Since 2015 she has been working as a Data Scientist at Ringier Axel Springer Polska where she has been involved in many projects concerning real-world machine learning applications, including large-scale recommendation systems.

**Bipin Indurkhya** is a professor of Cognitive Science at the Jagiellonian University, Krakow, Poland. His main research interests are social robotics, usability engineering, affective computing and creativity. He received his Master's degree in Electronics Engineering from the Philips International Institute, Eindhoven (The Netherlands) in 1981, and PhD in Computer Science from University of Massachusetts at Amherst in 1985. He has taught at various universities in the US, Japan, India, Germany and Poland; and has led national and international research projects with collaborations from companies like Xerox and Samsung.