Detection of Groups With Biased Representation in Ranking

Yuval Moskovitch
Ben Gurion University of the Negev
yuvalmos@bgu.ac.il

Jinyang Li
University of Michigan
jinyli@umich.edu

H. V. Jagadish
University of Michigan
jag@umich.edu

Abstract—Real-life tools for decision-making in many critical domains are based on ranking results. With the increasing awareness of algorithmic fairness, recent works have presented measures for fairness in ranking. Many of those definitions consider the representation of different "protected groups", in the top-k ranked items, for any reasonable k. Given the protected groups, confirming algorithmic fairness is a simple task. However, the groups' definitions may be unknown in advance.

In this paper, we study the problem of detecting groups with biased representation in the top-k ranked items, eliminating the need to pre-define protected groups. The number of such groups possible can be exponential, making the problem hard. We propose efficient search algorithms for two different fairness measures: global representation bounds, and proportional representation. Then we propose a method to explain the bias in the representations of groups utilizing the notion of Shapley values. We conclude with an experimental study, showing the scalability of our approach and demonstrating the usefulness of the proposed algorithms.

I. Introduction

Ranking is a commonly used operation in a wide range of application domains, for example, in presenting results on a web search engine [9], establishing credit scores [8], school admission [26] and hiring [15]. While convenient and useful, these tools can be biased. As a result, they may affect decision-making in undesirable ways and can even impact human life [3], [7], [12]. This problem has drawn much attention from the research community, and a line of recent works has focused on measuring and mitigating bias and unfairness in ranking [4], [11], [15], [19], [30], [32], [34].

The notion of algorithmic fairness was studied extensively for a broad class of models [22], [27]. Fairness measures typically refer to a given "protected group" in the data, which is defined based on the values of some sensitive attributes (e.g., gender, race, age, or combinations thereof), usually based on societal history of discrimination. Analyzing the fairness measure of a system with respect to the given group is a simple task. However, "non-standard" protected groups cannot always be specified in advance, and such groups may be overlooked when examining the performance of a system.

For example, a model developed to assign grades to students (in place of exams that were canceled due to the COVID-19 pandemic) was shown to be biased against high-achieving students from poor school districts[1]. For instance, students

[1]https://www.nytimes.com/2020/09/08/opinion/international-baccalaureate-algorithm-grades.html

from low-income families were predicted to fail the Spanish exam, even when they were native Spanish speakers. In this case, the model was discriminating against Hispanic students from poor school districts. A primary source of bias was the use of historical exam results of each school to predict student performance. However, using the school (identified by school ID) to define the protected group is not an intuitive choice, and so may not have been considered. Moreover, even if we consider the group of Hispanic students as a protected group, we may not find any fairness issues, since this subgroup of students is only a small fraction of all Hispanic students.

In this paper we study the problem of detecting groups that are treated unfairly by a ranking algorithm. In other words, we want to let the data speak to (potential) unfairness, without requiring a human modeler to identify protected attributes ahead of time. Following fairness definitions presented in the literature on fairness in ranking (see e.g., [11], [27], [32]), we consider group representation in the top-k ranked items for any k in a reasonable range as a measure of fairness.

Recent works have studied the problem of automatically detecting "problematic" or biased subgroups in the data without the need to specify the protected attributes a priori [10], [13], [18], [20], [25]. However, these works considered only classification models. In [24] the authors of [25] extend their framework to consider ranking as well. In contrast to our work, which builds on fairness measures for ranking from the literature, they use the notion of divergence to measure performance differences among data subgroups. This difference leads to differences in the result sets returned by each method (see Section VI-D for more details).

We next outline our main contributions.

Problem formulation: We formally define the problem of detecting groups with biased representation in the top-k ranked items for a given ranking algorithm R, a dataset D, and a range of possible k's. Groups are defined using value assignment to a set of attributes we denote as *patterns* (see Section II). To provide concise and meaningful results we use a threshold on the returned groups' size and report only groups that are not subsumed by any other group in the result set (referred to as the *most general patterns*, see Section III). We start with the fundamental definition of [11], which uses upper and lower bounds to restrict the number of tuples in the top-k from different groups in the data. The goal is to report groups such that their representation (i.e., number of tuples) in the

top-k does not lie within the given bounds for a given range of possible k's. We refer to this problem as the *global bounds representation bias* problem. We then consider the prominent class of fairness measures utilizing proportional representation (see, e.g., [32]). Intuitively, the representation of each group in the top-k should be proportional to its size in the data. Using this notion we define the *proportional representation bias* problem. We show that no polynomial algorithm exists to solve either problem.

Detection of groups with biased representation: We present algorithms for the problem of finding the set of all substantial groups (in terms of their size in the data, and their subsumption in other groups) with biased representation in the top-k ranked items. We first present a simple baseline solution that utilizes the notion of pattern graph presented in [6]. We show how to traverse the graph in a top-down fashion in order to find groups with biased representation. This search is then applied repeatedly for each k in the given range. Bearing in mind the complexity of the problem, we focus on optimizing the search. Our optimized solutions rely on the fact that the set of top-k and top-$(k+1)$ tuples differ by a single tuple. As a result, the search spaces for succeeding k values are typically very similar. The optimized solutions utilize this observation to avoid parts of the search tree.

Result analysis: Given a group with biased representation in the top-k ranked items, an analyst may wish to understand the cause of bias. To this end, we propose a method that harnesses the notion of Shapley values to identify attributes that significantly affect the ranking of the detected group. To analyze the difference between the detected group and top-k ranked tuples, we visualize the value distribution of such attributes. Shapley values have been used to provide similar explanations for regression and classification models. Our novelty is in developing a corresponding method for the ranking problem.

Experimental study: We complement our algorithmic development with an extensive experimental study. We evaluate the performance and properties of the algorithms, i.e., the scalability and parameter setting effect. We examine the effect of the number of attributes, groups' size threshold, and range of k, using three real-world datasets. Our results show the applicability of our solution in practice, despite the theoretical complexity of the problems, and the usefulness of the optimized algorithms compared to the baseline solution. We then experimentally demonstrate the usefulness of our approach for results analysis. Finally, we compare the result of our algorithms to the results of the method proposed in [24] through a case study.

Paper organization: The rest of the paper is organized as follows. We present the necessary preliminaries for our problem definition in Section II. Then in Section III we formally define the problems of detecting groups with biased representation in the data and prove their hardness. Our solutions is presented in Section IV. In Section V we introduce a method for analyzing and explaining the results of our algorithms. We describe our experimental evaluation in

Section VI, overview related work in Section VII and conclude in Section VIII.

II. PRELIMINARIES

We next provide necessary background on the notion of patterns to represent data groups and the concept of fairness in ranking. We will use the following example as our running example to demonstrate the ideas presented in the paper.

Example 2.1: The Student Performance Data Set [14] contains information from two Portuguese secondary schools in the Alentejo region of Portugal, Gabriel Pereira (GP) and Mousinho da Silveira (MS). The data was collected during the 2005-2006 school year and it contains the performance of 1044 students in the Math and the Portuguese language exams, along with demographic, social, and school-related information. Figure 1 depicts a sample from the data with the attributes: gender, school, address (urban or rural), and failures (number of past class failures). The grade attribute depicts the students' grades on a scale of $0 - 20$. Consider an excellence student program committee that wishes to select students for a scholarship based on their academic achievements. To this end, they use a ranking algorithm R to rank students by their grades. In the case of similar grades, students with fewer failures are ranked higher. The scholars' list is publicly announced, and should be diverse and inclusive.

A. Data Groups

We assume the data is represented using a single relational database, and that the relation's attribute values used for group definitions are categorical. To include attribute values drawn from a continuous domain in the group definition, we render them categorical by bucketizing them into ranges: very commonly done in practice to present aggregate results. We use the notion of patterns, value assignment to a set of attributes, to define groups in the data [6].

Definition 2.2 (Patterns): Let D be a database with categorical attributes $\mathcal{A} = \{A_1, \ldots, A_n\}$ and let $Dom(A_i)$ be the active domain of A_i for $i \in [1..n]$. A *pattern* p over D is the set of $\{A_{i_1} = a_1, \ldots, A_{i_k} = a_k\}$ where $\{A_{i_1}, \ldots, A_{i_k}\} \subseteq \mathcal{A}$ and $a_j \in Dom(A_{i_j})$ for each A_{i_j} in p. We use $Attr(p)$ to denote the set of attributes in p.

We say that a tuple $t \in D$ *satisfies* a pattern p if $t.A_i = a_i$ for each $A_i \in Attr(p)$. The *size* $s_D(p)$ of a pattern p is then the number of tuples in D that satisfy p. Given a ranking algorithm R we use $R^k(D)$ to denote the top-k ranked items in D. Finally, we use $s_{R^k(D)}(p)$ to denote the size of p in $R^k(D)$.

Example 2.3: Consider the dataset given in Figure 1. $p = \{School = GP\}$, is an example of a pattern. Tuples 3, 4, 7, 8, 12, 13, 15 and 16 satisfy p and thus $s_D(p) = 8$. For the ranking algorithm R whose results are depicted in the Rank column, we have $s_{R^5(D)}(p) = 1$, since only one tuple in the top-5 ranked items satisfies p.

B. Fairness measure

The problem of fairness in ranking was studied in a line of works (see [27] for a survey). A fundamental definition,

#	Gender	School	Address	Failures	Grade	Rank
1	F	MS	R	1	11	8
2	M	MS	R	1	15	3
3	M	GP	U	1	8	10
4	M	GP	U	2	4	16
5	M	MS	R	0	19	2
6	F	MS	U	1	4	15
7	F	GP	R	1	7	11
8	M	GP	R	1	6	13
9	F	MS	R	0	14	4
10	F	MS	R	2	7	12
11	M	MS	R	2	13	6
12	F	GP	U	0	20	1
13	F	GP	U	2	12	7
14	M	MS	U	1	13	5
15	F	GP	U	1	5	14
16	M	GP	U	0	9	9

Fig. 1: Students' data. The Rank column depicts their ranking based on the grade and number of past failures. The top-5 ranked students are highlighted

presented in [11], uses constraints over the representations of different groups in the top-k ranked items. They use an upper bound U_{kl} and a lower bound L_{kl} over the number of items with the property l (i.e., a protected group) in the top-k positions of the ranking. Then a ranking algorithm is fair by the definition of [11], if the number of selected items from the protected group in the top-k lies within the given boundaries.

Example 2.4: Consider again the dataset given in Figure 1 and the ranker whose result is presented in the Rank column. Consider a lower bound of 2 over the number of students from each school for $k = 5$ (i.e., $L_{5,school=MS} = L_{5,school=GP} = 2$). In this case, among the top-5 students, only one is from the GP school, thus the ranker does not satisfy the constraints.

Another prominent class of fairness measure in ranking considers the proportional representation of different groups in the top-k ranked items (see, e.g [32]). Intuitively, these definitions can be seen as variants of the definition of [11] such that for each group g, and each k, the bounds on the number of occurrences of items from g in the top-k ranked items are defined with respect to the size of g in the dataset.

Example 2.5: Continuing Example 2.4, the total number of students from each school (MS and GP) is 8. The total dataset size is 16, thus a proportionate representation of each school in the top-5 items should be roughly $5 \cdot \frac{8}{16} \approx 2$.

III. PROBLEM DEFINITION

Our goal is to detect groups with biased representation in the top-k ranked items for a given ranking algorithm R, dataset D, and a range of k's. We consider two variants of representation constraint in ranking. The first simply utilizes the definition of [11]. The fairness definition of [11] restricts the count of different groups (i.e., patterns) in the top-k using upper and lower bounds. According to this definition, the result is biased either when the size of a pattern l exceeds the upper bound U_{kl} or falls below the lower bound L_{kl} among the top-k tuples for some k. We eliminate the requirement to define l in advance, and use U_k and L_k to denote the upper and lower bound

respectively, on every pattern in the top-k ranked tuples of a given ranking algorithm.

We say that a group has a biased representation in the output of a ranking algorithm R, if its size in the top-k ranked items by R does not lie within the given bounds for any k in a given range of possible k's. Intuitively, we wish to avoid reporting "very specific" descriptions of groups and provide the user with a concise set of properties that characterize meaningful groups (in terms of their size) that have biased representation. To this end, we present the notion of *most general* patterns. Given the bounds over the group's representation in the top-k ranked items, we say that a pattern p is the most general pattern with biased representation, if p is used to represent a group with inadequate representation by the given bounds, and $\forall p' \subsetneq p$, the count of p' in $R^k(D)$ lies within the given boundaries. We are now ready to formally define our problem.

Problem 3.1 (Global Bounds Representation Bias): Given a database D, a ranking algorithm R, a size threshold τ_s[2], a range $[k_{min}, k_{max}]$, and lower bounds L_k and upper bounds U_k for each $k_{min} \leqslant k \leqslant k_{max}$, find for each $k_{min} \leqslant k \leqslant k_{max}$, all most general patterns p with size $\geqslant \tau_s$ such that $s_{R^k(D)}(p) < L_k$ or $s_{R^k(D)}(p) > U_k$.

Note that the ranking algorithm is treated as a black box, making the problem to be model agnostic. Following the line of work on proportional representation, we consider another problem definition by refining the above definition to account for the groups' sizes in the dataset. Intuitively, the number of items from each group in the top-k ranked items should be proportionate to the group's representation in the data.

Problem 3.2 (Proportional Representation Bias): Given a database D, a ranking algorithm R, a size threshold τ_s and a range $[k_{min}, k_{max}]$, find for each $k_{min} \leqslant k \leqslant k_{max}$, all most general patterns p with size $\geqslant \tau_s$ such that $s_{R^k(D)}(p) < \alpha \cdot s_D(p)\frac{k}{|D|}$ or $s_{R^k(D)}(p) > \beta \cdot s_D(p)\frac{k}{|D|}$ for $\alpha < \beta \in \mathbb{R}$.

Proportional representation gives the user an intuitive bounds definition. However, the definition of global representation allows the user to actively control the bounds over the representation of different groups in the top-k ranked items, even if their representation in the overall data is low/high. For instance, consider a ranking algorithm for job applicants in fields that are dominated by men (e.g., tech companies). If the company wishes to increase the representation of women they hire but their application number is low and only the top-k ranked applicants are invited for an interview, by using proportional representation, their number in the top-k, and as a result, the number of women invited to an interview, would be low as well.

While there are typically far fewer most general patterns with biased representation than the set of all patterns with biased representation, in the worst case, their numbers can be exponential. This is true even if we consider only the lower bounds (e.g., if $U_k = |D|$).

---

[2]We use an absolute value as the size threshold. Equivalently it may be defined as a fraction of the dataset size.

	A_1	A_2	$\cdots$	A_{n-1}	A_n	Rank
t_1	1	0	$\cdots$	0	0	1
t_2	0	1	$\cdots$	0	0	2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
t_n	0	0	$\cdots$	0	1	n
t_{n+1}	0	0	$\cdots$	0	0	$n+1$

Fig. 2: Dataset D in the proof of Theorem 3.3

Fig. 3: Part of the pattern graph for the running example. Edges of the search tree are marked with solid lines.

Theorem 3.3: Given a dataset D and a ranking algorithm R, no polynomial time algorithm can guarantee the enumeration of the set of all most general patterns with biased representation in the result of R on D.

Proof 3.4: We prove the theorem by construction. Consider a dataset D with n (assuming $n \geq 2$) binary attributes $\{A_1 \ldots, A_n\}$ and $n + 1$ tuples $t_1, \ldots, t_n, t_{n+1}$ as shown in Figure 2. I.e., $\forall i \in [1, n], t_i[A_i] = 1$, and $\forall j \neq i, t_i[A_j] = 0$. All the attributes of t_{n+1} are zero. Let R be a ranking algorithm such that the top-k tuples in D are the tuples $t_1, \ldots, t_k$ in Figure 2. Let $k_{min} = k_{max} = n$, $L_k = \frac{n}{2} + 1$ for Problem 3.1 (global representation bounds), and $\alpha = \frac{n+3}{n+4}$ for Problem 3.2 (proportional representation), for some $n \geq 2$.

Consider a pattern p with $m \leq n$ attributes A_i with the value assignment $A_i = 0$. Let I be the set of indices of those attributes. Among $t_1, \cdots, t_n$, the size of p is $n - m$, since $\forall i \in I, t_i[i] = 1, t_i$ does not satisfy p, and all the other tuples satisfy p. Let $m = \frac{n}{2}$, thus the size of p in the top-k ranked items is $s_{R^k(D)}(p) = \frac{n}{2} < L_k = \frac{n}{2} + 1$ in the case of global representation bounds. For proportional representation, we have $s_D(p) = \frac{n}{2} + 1$ since t_{n+1} also satisfies p. And we get $s_{R^k(D)}(p) = \frac{n}{2} < \alpha \cdot s_D(p) \cdot \frac{k}{|D|} = \frac{n+3}{n+4}(\frac{n}{2} + 1)\frac{n}{n+1} = \frac{n}{2}\frac{(n+3)(n+2)}{(n+4)(n+1)}$. To show p is a most general pattern to report, we examine the parents of p, p' which has $m - 1$ attribute with the value assignment 0. For global representation bounds, we have $s_{R^k(D)}(p') = n - m + 1 = \frac{n}{2} + 1 = L_k$. For proportional representation, we have $s_{R^k(D)}(p') = \frac{n}{2} + 1 > \alpha \cdot s_D(p') \cdot \frac{k}{|D|} = \frac{n+3}{n+4}(\frac{n}{2} + 2)\frac{n}{n+1} = \frac{n+2}{2}\frac{(n+3)n}{(n+1)(n+2)}$. As a result, every pattern with $\frac{n}{2}$ attribute assigned to 0 should be in the result set. The number of such patterns is $\binom{n}{n/2} > \sqrt{2}^n$, which is exponential. Therefore, any algorithm enumerating these patterns is exponential.

Upper bounds: The notion of the most general patterns is motivated by the utility of the information they provide. For example, in the case of global representation bounds, if the number of females in the top-k is less than the lower bound, then clearly the number of black females is below the bound. Unlike the most general patterns for the lower bound, in the case of the upper bound, the most specific patterns are more informative. For instance, if the number of black females is above the upper bound, then so is the number of blacks and the number of females. A plausible problem definition may account for the *most specific substantial patterns*. Analogously to the definition of the most general patterns, a pattern p is a *most specific substantial pattern* if the size of p is above a given threshold τ_s and for every pattern p' such that $p \subsetneq p'$,
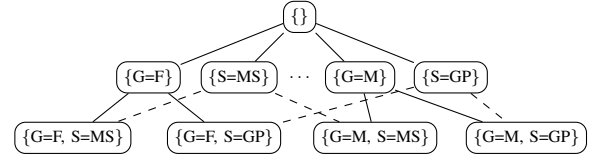
the size of p' is below the threshold τ_s. The goal is then to find the most general patterns that do not satisfy the lower bound and the most specific substantial patterns that exceed the upper bound. For ease of presentation, in the rest of the paper we will focus on the solution for Problems 3.1 and 3.2 considering only the lower bounds. We note that our solutions can be adjusted to support such problem definition (and other definitions such as most general for upper bound, and the most specific for lower bound).

IV. Detecting Groups with Biased Representation

We next present our algorithms for detecting groups with biased representation as defined in Problems 3.1 and 3.2. We start with a simple solution that can be used to detect groups based on both of the problem definitions. We then present two optimized algorithms, designed to optimize the search for each of the problems.

A. Iterative Top-Down Search (Baseline Solution)

The first, simple solution, utilizes the algorithm presented in [6] to traverse the set of possible patterns, starting with the most general ones, and compute the representation of each group in the top-k ranked items in the data (for each k in the given range). This is done using the notion of *pattern graph* [6]. Briefly, the nodes in the graph are the set of all possible patterns, and there is an edge between a pair of patterns p and p' if $p \subset p'$ and p' can be obtained from p by adding a single attribute value pair. In this case, we say that p (p') is a parent (child) of p' (p). As shown in [6], the pattern graph can be traversed in a top-down fashion, while visiting each pattern at most once. Namely, traversing a spanning tree of the pattern graph, which we denote as the *search tree*, and formally define as follows.

Definition 4.1: Let D be a dataset with categorical attributes $\mathcal{A} = \{A_1, \ldots, A_n\}$. We assume attributes are ordered, and for a given set of attributes $S \subseteq \mathcal{A}$ we use $idx(S)$ to denote the maximal index value of all attributes in S. Let p be a node in the pattern graph of D. The children of p in the search tree T are p' such that p' is a child of p in the pattern graph, and $idx(Attr(p') \setminus Attr(p)) > idx(Attr(p))$. Namely, p' is obtained from p by adding a single attribute value pair such that the index of the newly added attribute is larger than the maximal index value of attributes in p.

Example 4.2: A part of the pattern graph for the dataset of the running example is shown in Figure 3, where G and S are used as shorthands for Gender and School respectively. The pattern {G=F, S=GP} is a child node of the patterns {G=F}

and {S=GP} in the pattern graph, however, in the search tree it is only a child of {G=F}. This is because $idx(\{G\}) = 1 < 2 = idx(\{G, S\}) \setminus \{G\})$ whereas $idx(\{S\}) = 2 > 1 = idx(\{G, S\} \setminus \{S\})$.

Algorithm 1: Top-down search

input : A dataset D, a ranking algorithm R, a size threshold τ_s, k and lower bound L_k

output: $Res = \{p_1, \ldots, p_n\}$ where $\forall p_i \in Res\ s_D(p_i) \geq \tau_s$ and p_i is a most general pattern with $s_{R^k(D)}(p) < L_k$

`/* For proportional bounds α is given as input`
`   and L_k = α · s_D(p) k/|D|                    */`

1   $Res \leftarrow \emptyset$

2   $\mathcal{S} \leftarrow \{\texttt{generateChildren}(\{\})\}$

3   **while** $\mathcal{S}$ *is not empty* **do**

4      $p = \mathcal{S}.pop()$

5   **if** $\texttt{patternSize}\ (p, D) > \tau_s$ **then**

6         $top\text{-}k\_c \leftarrow \texttt{patternSize}\ (p, R^k(D))$

         `/* For proportional bounds use`
         `   top-k_c < α · s_D(p) k/|D|              */`

7   **if** $top\text{-}k\_c < L_k$ **then**

8            $\texttt{update}\ (Res, p)$

9   **else**

10          $\mathcal{S}.push(\texttt{generateChildren}(p))$

11   **return** Res

Algorithm 1 detects patterns with adequate size in the data (namely above a given threshold τ_s) and low representation (less than L_k), in the top-k ranked items for a given dataset D, a ranking algorithm R and a given k. It traverses the search tree of the pattern graph (top-down), using a queue $\mathcal{S}$ and maintains the set of identified patterns with $s_{R^k(D)} < L_k$. First it initializes the result set Res to $\emptyset$ (line 1) and sets $\mathcal{S}$ to contain the children of the most general (empty) pattern (line 2). While the queue $\mathcal{S}$ is not empty (lines 3 – 10), the algorithm extracts the first pattern in the queue p (line 4), and computes its size in D. If it is greater than τ_s (line 5), the size of p in $R^k(D)$, $s_{R^k(D)}$, is computed (line 6). If $s_{R^k(D)}$ is bellow the lower bound L_k (line 7), Res is updated using the procedure $\texttt{update}$ (line 8), that checks whether any ancestor of p in the pattern graph is already in Res (this is possible since the algorithm traverses the search tree and not the patterns graph). Otherwise (line 9), $s_{R^k(D)}$ exceeds the lower bound L_k, and the children of p are added to the queue using the procedure $\texttt{generateChildren}$ (line 10), which generates the children of a node as defined in Definition 4.1. Finally, Res is returned (line 11).

ITERTD *algorithm (baseline):* Given a dataset D and a ranking algorithm R, a size threshold τ_s, a range $[k_{min}, k_{max}]$ and lower bounds L_k for each $k_{min} \leq k \leq k_{max}$, a simple solution for detecting groups with biased representation based on the global representation bounds definition utilizes Algorithm 1, to apply a top-down search for each $k_{min} \leq k \leq k_{max}$. Then, in each iteration report the patterns with low representation in the top-k ranked items. Similarly,

Algorithm 1 can be used for the case of proportional representation, with some slight modifications (shown as comments in Algorithm 1). The objective is to report the patterns p with adequate size but insufficient representation in the top-k tuples $R^k(D)$, where the representation in $R^k(D)$ should be proportional to the representation in D. In this case, the bounds L_k are not given as input, instead, a bound for each pattern is computed based on its size in the data and a value α. Note that the pattern's size is computed in line 5, and given α we can replace the condition in line 7 with the condition $top\text{-}k\_c < \alpha \cdot s_D(p) \frac{k}{|D|}$.

We next propose more efficient algorithms for the problems.

B. Global Representation Bounds

The key observation is that when the lower bound remains the same for k and $k+1$[3], the search spaces for patterns with biased representation in the ranking result of R for k and $k+1$ are typically very similar. This is because the set of top-k and top-$(k+1)$ tuples differ by a single tuple. Namely, increasing k implies only local modifications to the search space. Let T_k be the search tree generated to find the set of unfairly treated patterns in the top-k tuples, and $R(D)[k+1]$, the $(k+1)$ element in the result of ranking D using R. We can bound the number of nodes in T_k whose size is affected by increasing k.

Proposition 4.3: $R(D)[k+1]$ can satisfy at most $\frac{|T_k|}{2}$ patterns (nodes) in T_k, where $|T_k|$ denotes the number of nodes in T_k.

Proof 4.4: (Sketch) The basic idea of the proof is that whenever a pattern p is generated during the search, at least one pattern p' with the same set of attributes ($Attr(p) = Attr(p')$) that differs from p in the value assignment of a single attribute is generated as well. This is true under the assumption that every attribute has at least 2 values. For instance, the patterns {Gender = M, School = MS} and {Gender = M, School = GP} are both children of the pattern {Gender = M} and are generated when the procedure $\texttt{generateChilder}(\{Gender = M\})$ is invoked. Let A_i be the attribute such that $A_i \in Attr(p)$ and $\{A_i = a_i\} \subseteq p$ while $\{A_i = a_i'\} \subseteq p'$. $R(D)[k+1]$ can satisfy at most one of the patterns, as the value of A_i in $R(D)[k+1]$ is either a_i or a_i' (or possibly other value, if $|Dom(A_i)| > 2$). Since this is true for every node generated during the search, $R(D)[k+1]$ can satisfy at most $\frac{|(T_k)|}{2}$ patterns (nodes) in T_k.

In that case, by starting the search for $k+1$ from the endpoint of the search for k, we significantly reduce the search space (and as a result, the runtime, see Section VI-B) .

GLOBALBOUNDS *algorithm:* Our optimized algorithm for the problem for detecting groups with global representation bias in the top-k, GLOBALBOUNDS, is depicted in Algorithm 2. GLOBALBOUNDS starts by initializing the result set map Res (line 1). It then performs a top-down search for the case where $k = k_{min}$ (line 2). This is done using the procedure $\texttt{TopDownSearch}$, similar to the algorithm

---

[3]We assume $L_k \leqslant L_{k+1} \forall k\ \ k_{min} \leqslant k < k_{max}$. This is a reasonable assumption since as k increases, so is the number of tuples in the top-k, thus it is only logical the bounds increase as well.

Algorithm 2: GLOBALBOUNDS. Detecting groups with biased representation based on global bounds

input : A dataset D, a ranking algorithm R, a size threshold τ_s, a range $[k_{min}, k_{max}]$ and lower bounds L_k for each $k_{min} \leqslant k \leqslant k_{max}$

output: Res s.t. for each $k_{min} \leqslant k \leqslant k_{max}$ $Res[k] = \{p_1, \ldots, p_n\}$ where $\forall p_i \in Res[k]$ $s_D(p_i) \geq \tau_s$ and p_i is a most general pattern with $s_{R^k(D)}(p) < L_k$

1   $Res \leftarrow \emptyset$
2   $Res[k_{min}], DRes \leftarrow$
     TopDownSearch$(D, R, \tau_s, k_{min}, L_{k_{min}})$
3   **for** $k = k_{min} + 1$ *to* k_{max} **do**
4   **if** $L_{k-1} < L_k$ **then**
5       $Res[k], DRes \leftarrow$
         TopDownSearch$(D, R^k(D), \tau_s, L_k)$
6   **else**
7       $Res[k] \leftarrow Res[k-1]$
8   **foreach**
         $b \in \{p \in Res[k-1] \mid R(D)[k] \text{ satisfies } p\} \cup DRes$
         do
9          $Res[k], DRes \leftarrow$
           searchFromNode $(b, Res[k], DRes)$

10 **return** Res

---

depicted in Algorithm 1, with a minor addition. It maintains a set $DRes$ of patterns p reached during the search with size below the lower bound in top-k, that are not part of the result set since it already contains an ancestor of p. TopDownSearch returns both, the result set of the search Res, and the set $DRes$. When k increases (and L_k is kept intact), the algorithm will utilize this set to initiate a local search in the pattern graph.

Next, the algorithm preforms the search for each k from $k = k_{min} + 1$ through $k = k_{max}$ (lines 3–9). For each k, if the bound increases, TopDownSearch is used to perform a new top-down search. Otherwise, The algorithm considers only patterns from $DRes$ and patterns from $Res[k - 1]$ that the newly inserted tuple $R(D)[k]$ satisfies (line 8). This is because only their sizes in the top-k are affected by the new tuple (at most half of the tree, based on Proposition 4.3). For each such pattern, the algorithm applies the procedure searchFromNode (line 9) to resume the search in the relevant parts of the graph. This search updates $Res[k]$ and $DRes$. Finally, Res is returned (line 10).

Proposition 4.5: GLOBALBOUNDS returns the set of all most general patterns p with bias representation using global bounds in the top-k for each k in the given range.

The proof is by induction on k with a base case of $k = k_{min}$. Details are omitted due to space constraints.

Example 4.6: Consider again D and R from the running example. Assume we are given the size threshold $\tau_s = 4$, $k_{min} = 4$, $k_{max} = 5$, and the lower bounds $L_4 = L_5 = 2$. At the end of the top-down search for $k = 4$, the result set $Res[4]$ contains (among others) the patterns {Address = U} and {Failures = 1}, that appears only once in the top-4

tuples (namely, below the lower bound). $DRes$ contains, for instance, the patterns {Gender = F, Address = U}, {Gender = M, Address = U}, {Gender = F, Failures = 1} and {Address = R, Failures = 1}. These patterns were generated during the top-down search and have ancestors in $Res[4]$ ({Address = U} and {Failures = 1}). Next, the algorithm turns to compute patterns with biased representation for $k = 5$. The new tuple in the top-5 is tuple 14. It matches the patterns {Address = U} and {Failures = 1} in $Res[4]$. Thus the algorithm performs the search starting from those nodes. Their sizes in the top-5 exceed the lower bound. In this search, these two patterns are extracted from the result set and the pattern {Address = U, Failures = 1} is added. From the set $DRes$, the patterns {Gender = F, Address = U}, {Gender = M, Address = U}, {Gender = F, Failures = 1} and {Address = R, Failures = 1} are added to the result set $Res[5]$, as their sizes in the top-5 tuples are still below the threshold L_5 and their respective ancestors are removed from the result set.

C. Proportional Representation

We next consider the problem of detecting groups with biased proportional representation as depicted in Problem 3.2. The inputs are a dataset D, a ranking algorithm R, a range $[k_{min}, k_{max}]$, a size threshold τ_s and $\alpha \in \mathbb{R}$. The objective is to report the patterns p with adequate size in D, but insufficient representation in the top-k tuples $R^k(D)$, where the representation in $R^k(D)$ should be proportional to the representation of p in D.

First, note that the optimized solution presented for the case of global representation bounds depicted in Section IV-B is not applicable in this case. Recall that GLOBALBOUNDS (Algorithm 2) aims at reducing the search space by avoiding searching areas in the pattern graph that were not changed between consecutive iterations. When the bound remains unchanged ($L_k = L_{k+1}$), patterns that the $(k + 1)$ tuple in the ranking does not satisfy, are not affected, and can be eliminated from the search. This is not the case for proportional representation, as the bound for each pattern depends on k as well.

Recall that the goal is to find patterns p such that $s_{R^k(D)}(p) < \alpha \cdot s_D(p) \cdot \frac{k}{|D|}$, and note that α and $s_D(p)$ do not change during the computations. Thus, the inequality holds depending on the value of k. Given R, D, α, a pattern p and a value k such that $s_{R^k(D)}(p) \geqslant \alpha \cdot s_D(p) \cdot \frac{k}{|D|}$, we denote by $\tilde{k}$ the minimal value for k such that the inequality does not hold when fixing the value of $s_{R^k(D)}(p)$. Namely, the minimal value such that $s_{R^k(D)}(p) < \alpha \cdot s_D(p) \cdot \frac{\tilde{k}}{|D|}$.

Example 4.7: Let $\alpha = 0.9$. For D and R from our running example, $p = \{$Gender = F$\}$ satisfies the inequality for $k = 4$ since $s_{R^k(D)}(p) = 2 > 1.8 = 0.9 \cdot 8 \cdot \frac{4}{16}$. $\tilde{k} = 5$ in this case since $0.9 \cdot 8 \cdot \frac{5}{16} = 2.25$.

Intuitively, if k is increased up to $\tilde{k}$ but the number of tuples satisfying p remains the same, then the representation of p is biased in the ranking result. If there is no ancestor of p in the result set, then p should be added to it. To this end, our optimized algorithm, PROPBOUNDS, computes for

each pattern in the search tree its corresponding $\tilde{k}$ value. It maintains a set $\mathcal{K}$ which indicates patterns that potentially, if they do not satisfy the $\tilde{k}$ element in the ranking, should be added to the result set. $\mathcal{K}$ contains patterns in a branch of the search tree whose $\tilde{k}$ values are monotonically decreasing. A pattern p in $\mathcal{K}$ with the corresponding $\tilde{k}$ value should be extracted from $\mathcal{K}$ and added to the result set when the computation reaches $k = \tilde{k}$ if $s_{R^k(D)}(p) = s_{R^{k-1}(D)}(p)$ (i.e., p does not satisfy $R(D)[k]$).

Algorithm 3: PROPBOUNDS. Detecting groups with biased proportional representation

---

input : A dataset D, a ranking algorithm R, a size threshold τ_s, a range $[k_{min}, k_{max}]$ and $\alpha \in \mathbb{R}$

output: Res s.t. for each $k_{min} \leqslant k \leqslant k_{max}$
$Res[k] = \{p_1, \ldots, p_n\}$ where $\forall p_i \in Res[k]$
$s_D(p_i) \geq \tau_s$ and p_i is a most general pattern with $s_{R^k(D)}(p) < \alpha \cdot s_D(p)\frac{k}{|D|}$

1 $Res \leftarrow \emptyset$
2 $Res[k_{min}], \mathcal{K}, DRes \leftarrow$
  TopDownSearch$(D, R, \tau_s, k_{min}, \alpha)$
3 **for** $k = k_{min} + 1$ *to* k_{max} **do**
4    $Res[k] \leftarrow Res[k-1]$
5    $Res[k], \mathcal{K}, DRes \leftarrow$
     selectiveTD $(D, R, \tau_s, k, \alpha, R(D)[k], \mathcal{K}, DRes)$
6   **foreach** $p \in DRes \bigcup_{\mathcal{K}[p']=k}\{p'\}$ *such that* $R(D)[k]$
     doesn't satisfy p **do**
7   update (Res, p)

8 **return** Res

PROPBOUNDS *algorithm:* PROPBOUNDS (Algorithm 3) operates as follows. Similarly to GLOBALBOUNDS, it starts by initializing the result set map Res (line 1) and applying a top-down search for k_{min} (line 2), as depicted in procedure TopDownSearch (with the required modification for proportional bounds), but in addition to sets $Res, DRes$, it also maintains the set $\mathcal{K}$. Then, the algorithm iterates over the values of k from $k_{min} + 1$ to k_{max} (lines 3 – 7). In each iteration, it first initializes $Res[k]$ with the result from the previous iteration $Res[k-1]$ (line 4). Then the algorithm applies a (partial) search from the root using the procedure selectiveTD (line 5). This search ignores the areas in the tree that are not affected by $R(D)[k]$. The result of the procedure is used to update the result set, $\mathcal{K}$, and $DRes$. The algorithm then iterates over the patterns in $DRes$ (patterns reached during the search that has an ancestor in the result) and all patterns in $\mathcal{K}$ with $\tilde{k} = k$ that are not affected by $R(D)[k]$, to determine the changes to the result set (lines 6 – 7). Finally, Res is returned (line 8).

Proposition 4.8: PROPBOUNDS returns the set of all most general patterns p with bias proportional representation in the top-k for each k in the given range.
Similarly to Proposition 4.5, the proof is by induction on k with base case of $k = k_{min}$. Due to space constraints, the details are omitted.

Example 4.9: Consider again D and R from the running example. Assume we are given the size threshold $\tau_s = $

$5, k_{min} = 4, k_{max} = 5$, and $\alpha = 0.9$. At the end of the top-down search for $k = 4$, the result set $Res[4]$ consists of the patterns {School = GP}, {Address = U} and {Failures = 1}. For each one of them, $s_D(p) = 8$, and thus the bound on $s_{R^k(D)}(p)$ is $\alpha \cdot s_D(p) \cdot \frac{k}{|D|} = 0.9 \cdot 8 \cdot \frac{4}{16} = 1.8$, but $s_{R^k(D)}(p)$ is only 1. The set $\mathcal{K}$ consists of {Gender = M} and {Gender = F}, both with $\tilde{k}$ of 5, and {School = MS} and {Address = R} with $\tilde{k} = 7$. Note that the pattern {School = MS, Address = R} was generated in the first top-down search, but was not added to $\mathcal{K}$ since its $\tilde{k}$ value is 9, higher than its parent in the search tree {School = MS}.

When k is increased to 5, the algorithm reexamines only the patterns {Gender = M}, {School = MS}, {Address = U} and {Failures = 1} that are affected by the $R(D)[5]$ (tuple 14). The patterns {Address = U} and {Failures = 1} remain in the result set for $k = 5$ even-though their size in the top-5 is larger, since the bound for $k = 5$ increases as well. Finally, the pattern {Gender = F} is added to $Res[5]$ based on the information from $\mathcal{K}$ (it is stored in $\mathcal{K}$ with $\tilde{k} = 5$).

V. RESULT ANALYSIS

With the results of our algorithm in hand, an analyst may wish to understand the cause of the bias in the representation of the detected groups. We propose a method to provide such explanations utilizing the notion of Shapley values [29]. Shapley values is a concept adopted from game theory to explain the effect of different attributes on the output of a model for a given input. The use of Shapley values has recently gained popularity in the field of interpretability and explainability of ML models [21], [31]. Given a regression model (or a classifier with probabilities) M, Shapley values are used to evaluate the contribution of each attribute on the output of M for a given input t. This is done by computing the weighted marginal contribution of each attribute value using all possible subsets of attribute values.

Intuitively, an explanation for the bias may be the values that affected the ranking of tuples in the given group. To this end, we propose a method for explanations that consists of two parts: the first identifies the attributes with the highest effect on the ranking of tuples in the given group (using Shapley values), and then we compare the values distribution of these attributes in the top-k and the biased represented group. In order to adopt the use of Shapley values, we need to tackle two challenges. The first is to adjust our problem's setting, where we are given a ranking algorithm R (as a black box) rather than a regression model. Second, Shapley values are used to explain the contribution of the attribute values of a single tuple, whereas we are interested in explaining the (inadequate) representation of a group of tuples (in the top-k).

To address the first challenge we compute a regression model M_R that simulates the process of R and can be used to approximate the effect of attribute values of a given tuple t on t's ranking by computing the Shapley values of $M_R(t)$. To this end we define $D_R = \{(t, R(D)[t]) \mid t \in D\}$, where $R(D)[t]$ is the ranking of t in $R(D)$, and use it to train a regression model M_R. Then, given a pattern p such that p

was returned by one of our algorithms for detecting groups with biased representation for a given k, to explain the result, we compute the Shapley values $(s_1^t, \ldots, s_m^t)$ for each tuple t such that t satisfies p, namely, for each tuple in the detected group. We then aggregate the results into a single Shapley value vector $(s_1, \ldots, s_m)$ for the pattern p such that

$$s_i = \frac{\sum_{t \text{ s.t. } t \text{ satisfies } p} s_i^t}{s_D(p)}$$

To show the differences between the pattern p and the top-k patterns, we visualize the value distribution of attributes with large Shapley values of tuples that satisfy the pattern p compared to their distribution among the tuples in the top-k. In Section VI-C we show that using our method we are able to disclose the attributes that were used for ranking (and thus affect the representation of groups in the top-k) when the ranking model is given as a black box. Moreover, we show that the value distribution in the attribute identified as most significant in the ranking is different for groups detected by our algorithms than for the top-k tuples, which indicates the identified attributes values explain the results.

VI. Experiments

We experimentally examine the proposed solutions using three real-life datasets. We start with our setup and then present a quantitative experimental study whose goal is to assess the scalability of our algorithms. In particular, we examine our algorithms' performance for each fairness definition as a function of the number of attributes, pattern's size threshold, and range of k. We then demonstrate our proposed method for the results analysis presented in Section V. We conclude with a comparison to the framework presented in [25], showing the differences in the results between our algorithms and the algorithm of [25] through a case study.

A. Experiment Setup

a) Datasets: We used three real datasets with different numbers of tuples and attributes as follows.

- **The COMPAS Dataset**[4] was collected and published by ProPublica as part of their investigation into racial bias in criminal risk assessment software. It contains the demographics, recidivism scores produced by the COMPAS software, and criminal offense information for 6,889 individuals. We used up to 16 attributes eliminating attributes such as names, ids, dates, etc.
- **Student Performance Dataset (Student dataset)**[5] shows the performance of students in secondary education of two Portuguese schools as described in Example 2.1. We considered in the experiment the data fragment with information regarding the Math exam (395 tuples and 33 attributes).
- **German Credit Dataset**[6] with financial and demographic information about 1,000 loan applicants with 20 attributes.

It was originally used in the context of classification, where each application is classified as a good or bad credit risk.

Compared algorithms: We evaluate the performance, in terms of the running time of our proposed algorithms.

- **IterTD (baseline).** The simple solution for detection of groups with biased representation, which iteratively applies a top-down search as depicted in Section IV-A.
- **GLOBALBOUNDS (Algorithm 2).** The algorithm for detecting groups with biased representation based on global bounds as described in Section IV-B.
- **PROPBOUNDS (Algorithm 3).** The algorithm for detecting groups with biased representation based on proportional representation bounds as described in Section IV-C.

Parameters setting: The number of attributes was set to be the maximal number the baseline solution could handle. We examined different settings and observed similar trends. For space constraints, unless stated otherwise, we report the result for the following set of default parameters (where in most experiments, the result set is neither empty nor too large, i.e., between 1 to 100): $\tau_s = 50$, $k_{min} = 10$, $k_{max} = 49$, and the lower bounds are 10 for $10 \leq k < 20$, 20 for $20 \leq k < 30$, 30 for $30 \leq k < 40$ and 40 for $40 \leq k < 50$ for global bounds, and $\alpha = 0.8$ for proportional representation.

Ranking Algorithms: The Student dataset was ranked based on the value of the attribute G3 showing the student's math final grades. For the COMPAS dataset, we performed similar ranking method as in [5]: We normalized attribute values c_days_from_compas, juv_other_count, days_b_screening_arrest, start, end, age, and priors_count as scoring attributes. Values are normalized as $(val - min)/(max - min)$. Higher values correspond to higher scores, except for age. Tuples are ranked descendingly according to their scores. For the German Credit dataset, we used the ranking presented in [32] based on creditworthiness.

All experiments were performed on a macOS machine with a 2.8 GHz Quad-Core Intel Core i7 CPU and 8GB memory. The algorithms were implemented using Python3.7.

B. Experimental results

Both GLOBALBOUNDS and PROPBOUNDS run much faster than the baseline, particularly as the the number of attributes increases and the baseline becomes exponentially more expensive. Details below.

Number of attributes: The first set of experiments aims to study the effect of the number of attributes on the running time. To this end, we varied the number of attributes in the datasets from 3 to $|\mathcal{A}|$ where $\mathcal{A}$ is the set of all attributes in the dataset. The number of attributes (along with their cardinality) determines the number of possible patterns, and as a result, the size of the search space. Thus, as the number of attributes increases, we expect to see a steep growth in the running time. The results (using a 10-minute timeout) are presented in Figures 4–5. Indeed, in all cases we observed a rapid increase in the running time, while GLOBALBOUNDS and PROPBOUNDS outperform ITERTD.
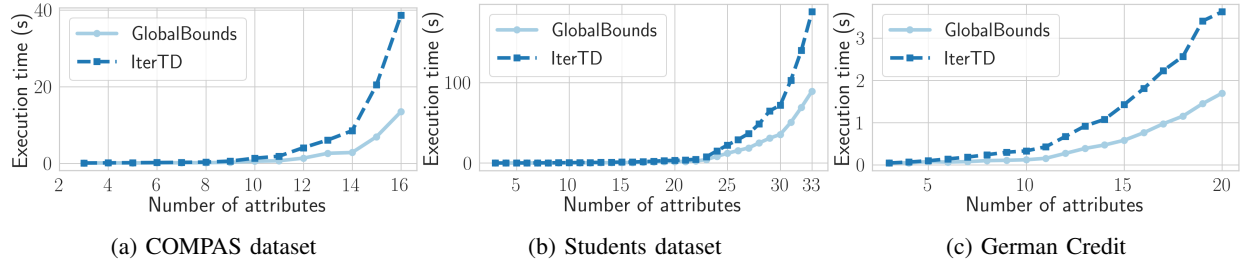
(a) COMPAS dataset   (b) Students dataset   (c) German Credit

Fig. 4: Running time as a function of number of attributes - Ranking with global bounds

(a) COMPAS dataset   (b) Student dataset   (c) German Credit
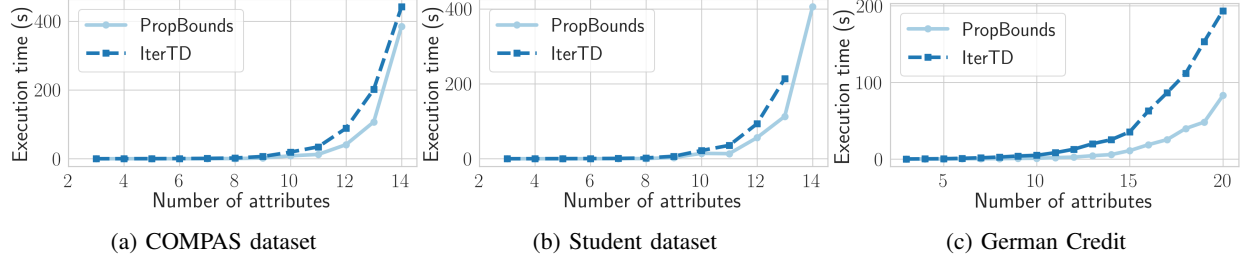
Fig. 5: Running time as a function of number of attributes - Ranking with proportional representation
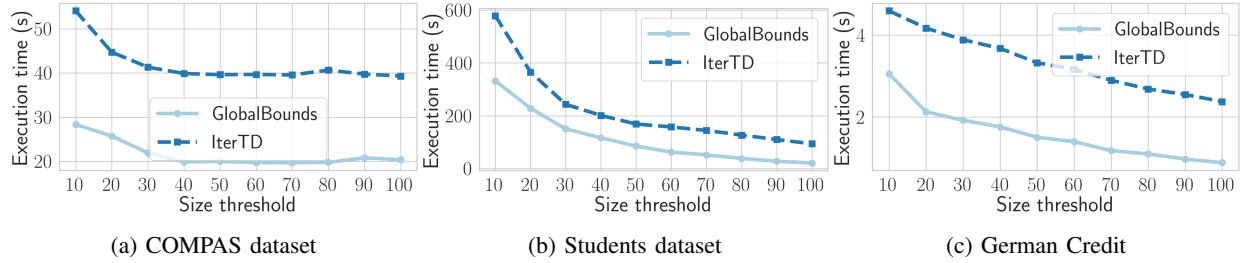
(a) COMPAS dataset   (b) Students dataset   (c) German Credit

Fig. 6: Running time as a function of the size threshold τ_s - Ranking with global bounds

(a) COMPAS dataset   (b) Students dataset   (c) German Credit
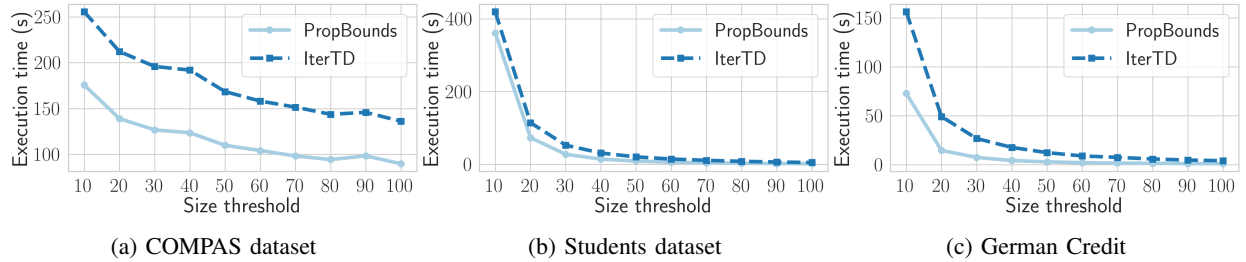
Fig. 7: Running time as a function of the size threshold τ_s - Ranking with proportional representation

Size threshold: In the next set of experiments, we assessed the effect of the size threshold τ_s on the running time. To this end, we varied the size threshold from 10 to 100 while using the default values for the rest of the parameters. The results are presented in Figures 6 and 7. We observed a decrease in the running times of the algorithms. This is because the number of patterns satisfying the size threshold decreases as the threshold increases, and as a result the search space is decreased as well. In all cases GLOBALBOUNDS and PROPBOUNDS outperform ITERTD.

Range of k: We examine the scalability with respect to the range of k considered by the algorithm. We varied the range from 40 (40) to 990 (340) by setting k_{min} to be 10, and increasing k_{max} from 50 (50) to 1000 (350) for COMPAS

(Student and German Credit) dataset and observed the effect on the running time. We set different maximum range of k due to different size of the datasets (6889 for COMPAS, 395 for Student and 1000 for the German Credit). The results are presented in Figure 8 and 9. In all cases the optimized algorithms outperform ITERTD, which illustrates the efficient reduction in the search space.

Recall that GLOBALBOUNDS and PROPBOUNDS optimize the search space compared to ITERTD by utilizing the search result of the iteration for k in order to compute the result set for $k + 1$. Thus, as the range of k increase, we expect to see a greater improvement in the performance of the optimized algorithm compared to the baseline solution. This trend is shown in Figure 8 and 9. To further demonstrate the useful-
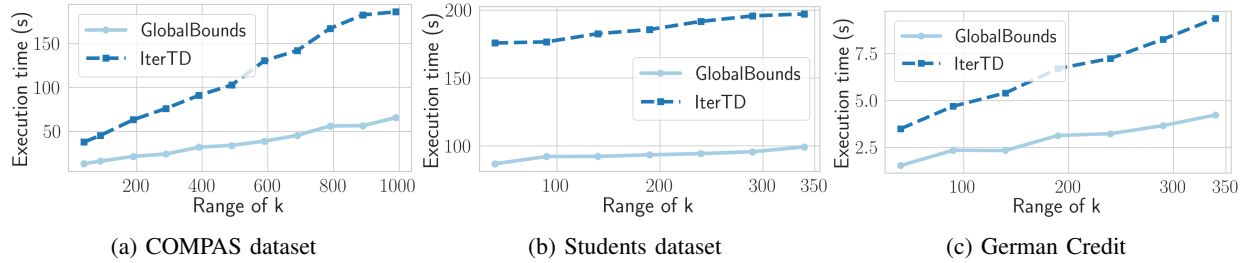
Fig. 8: Running time as a function of the range of k- Ranking with global bounds
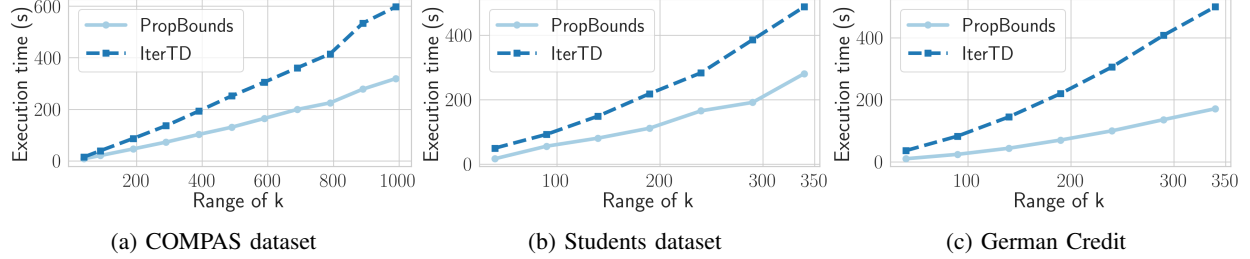

Fig. 9: Running time as a function of the range of k - Ranking with proportional representation

ness of the approach, we compared the number of patterns examined during the search for each one of the algorithms. The observed gain was up to 39.35% in the COMPAS dataset, 56.87% in the student dataset and 29.27% in the credit card dataset for detecting groups with biased representation using global bounds, and 39.60%, 20.49% and 56.83% respectively for proportional representation.

C. Result Analysis using Shapley values

We next demonstrate the usefulness of our proposed method for results analysis using Shapley values presented in Section V. The goal of the experiment is twofold. First, we show that our Shapley values based method for evaluating the effects of attributes on the ranking can indeed reveal useful information on the actual attributes used for ranking when the ranking algorithm is given as a black box. Then we show that the value distribution for those attributes can be used to explain the representation bias, by comparing the distributions for the values in the top-k with those in the detected groups and focusing on the differences.

We trained a regression model using the ranked data for each dataset and examined the Shapley values for groups detected by our algorithms. We present the results for the patterns (groups) $p_1 = \{$mother's education = primary education (4th grade)$\}$ in the Student dataset, $p_2 = \{$age = younger than 35$\}$ in COMPAS and $p_3 = \{$status of existing account = $(0 \leqslant \cdots < 200)$ DM[7]$\}$ from the German Credit dataset, which were detected by the GLOBALBOUNDS algorithm for $k = 49$ and $L_k = 40$. We observed similar results for other groups detected by the algorithms and other parameters. Figures 10a, 10b and 10c show the resulting aggregated Shapley values for

each group, as explained in Section V. We show the Shapley values for the six attributes with the larges values for each group, as the rest had significantly lower values (lower than 5.79%, 2.63% and 9.54% of largest aggregated Shapley values for p_1, p_2 and p_3 respectively).

For the group of students whose mother's education level is primary education, which was detected by our algorithm as a group with biased representation in the Student dataset, the final grade has the largest aggregated Shapley value on the ranking (Figure 10a). This result agrees with the fact that the value of the final grade is indeed used for ranking by the ranking algorithm (and it is in fact the only attribute used). Other than the final grade, the first and second period grades have a notable aggregated Shapley (although significantly lower aggregated Shapley value than the final grade). This is due to the high correlation between those attributes and the final grade [14]. We also noticed the mother's education attribute in the result. This may indicate some correlation between the mother's education and the final grade. However, we also note that the aggregation of the Shapley values for other attributes, e.g., father's education, show no clear pattern: some values have a positive effect and some negative, and different tuples in the group have different values. In contrast, all the tuples in the group have the same value (primary education) for mother's education attribute (since it is used to define the group). Therefore the Shapley values of the attribute for the different tuples in the groups are similar. This may also increase the aggregated value compared to other attributes. We observed this phenomenon, where the attributes used to define the detected group are slightly higher, for other groups detected by our algorithms also.

For the COMPAS dataset, tuples are ranked by a combined score based on seven attributes: days from compas, the number of other juvenile convictions, days before screening arrest,

---

[7]A Debit Memo (DM) on a company's bank statement refers to a deduction by the bank from the company's bank account. In other words, a bank debit memo reduces the bank account balance similar to a check drawn on the bank account.
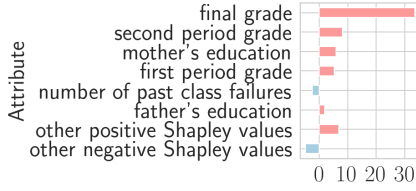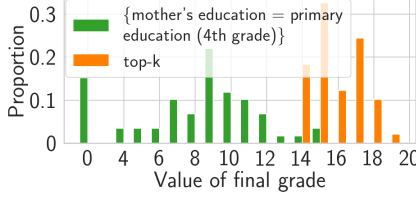
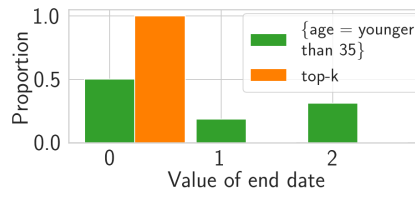(a) Aggregated Shapley value of group p_1 ={mother's education = primary education} in the Student datase
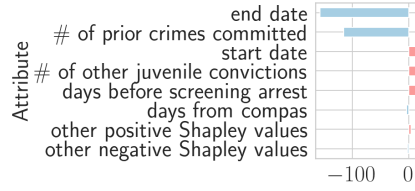
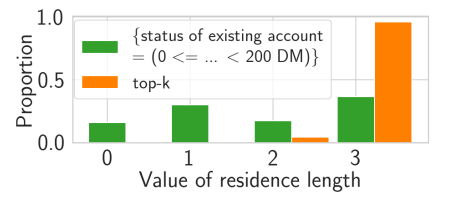(b) Aggregated Shapley value of group p_2 ={age = younger than 35} in the COMPAS dataset
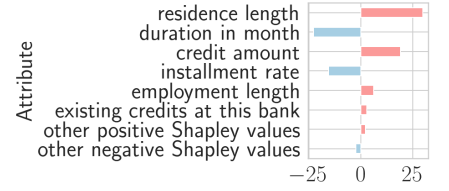
(c) Aggregated Shapley value of group p_3 ={status of existing account = (0 ⩽ ··· < 200DM)} in the German Credit dataset

(d) Value distribution of the final grade attribute in the Student dataset

(e) Value distribution of the end date attribute in the COMPAS dataset

(f) Value distribution of residence length in the German Credit dataset,

Fig. 10: Result analysis using Shapley values

start date, end date, age, and the number of priors crimes committed. In Figure 10b, showing the aggregated Shapley values of people younger than 35, six out of the above seven attributes are the six attributes with the largest Shapley values. In this case, the attribute end date and the number of priors crimes committed are identified as the most significant factor affecting the detected group.

For the German Credit dataset, tuples are ranked according to their ranking in [32], however the actual ranking algorithm is unknown. Namely, we do not have the ground truth and cannot verify the attribute detected as significant for explaining the bias in the representation of the detected group in the top-k are actually used by the ranking algorithm. The attributes residence length, duration in month, credit amount, and installment rate have the largest Shapley values as shown in Figure 10c. All of these attributes represents reasonable features to decide one's credit worthiness.

The Shapley value represents the effect of different attributes on the ranking of groups. To analyze the differences between the detected groups and top-k tuples (with respect to these attributes), we visualize the value distribution of attributes with the largest Shapley values in Figures 10d, 10e, and 10f. Since the number of tuples in the top-k and the detected group differ, the y-axis represents the proportion of tuples (rather than their count) with the values shown on the x-axis (the set of possible values for the attribute).

For all three datasets, we observed vast differences in the distributions of the values of the attribute with the largest Shapley value between the tuples in the top-k and the tuples in the group detected with biased representation. For the students dataset (Figure 10d), the final grades of tuples in the top-k all fall in the range of $15 - 20$, while most tuples in the detected group have a final grade lower than 15. In the COMPAS dataset (Figure 10e), the value of the end date for all top-k tuples is 0 while only half of the tuples in the detected

group have the same value, and almost 30% of them have the highest end date value (of 2). Similar results were observed in the value distribution of the residence length attribute as shown in Figure 10f.

D. Comparison with Existing Solution

The problem of identifying subgroups in the data that behave differently compared to the overall dataset was studied in [24]. Different from our problem definition, which relies on fairness measures for ranking to define groups with biased representation in the top-k ranked items, the work of [24] uses the notion of divergence to measure performance differences among data subgroups. Each data item in the data $t \in D$ is associated with an outcome $o(t)$ where the outcome function is defined based on the ranking of t by the ranking algorithm. The outcome of a group $o(G)$, is then the average of the outcome of every item $t \in G$. The divergence of a subgroup G in the data D is the difference between the outcome of G and the entire data, i.e., $o(G) - o(D)$. Given a threshold s on the subgroup size, the solution of [24] computes the divergence of *all* subgroups with sizes larger than s.

To better demonstrate the differences between the definitions and the resulting groups identified by each algorithm, we conducted an experiment using the Student dataset. We used the default size threshold of $\tau_s = 50$ (support in [24] of 0.13, i.e., 13% of the data). Since [24] does not consider a range of k's, we fixed $k_{min} = k_{max} = 10$ (namely, compare the results when $k = 10$). To allow for easy comparison, we used only the first 4 attributes of the data: school, sex, age, and address. We used the outcome function $o(t)$ that assigns the value 1 for tuples t in the top-k, and 0 for the rest (as presented in [24]). Finally, for our algorithms we used the default parameters of lower bound 10 for ranking with global bounds, and $\alpha = 0.8$ for proportional representation. PROPBOUNDS outputs 2 patterns: {sex=F} and {address=R}, both returned by GLOBAL-

BOUNDS as well. Additionally, GLOBALBOUNDS returned the patterns {school=GP}, {sex=M} and {address=U}, which had less than 10 instances in the top-10 ranked items (9, 7, and 9 respectively), but considering their overall size (349, 208 and 307 respectively), their representation in the top-k is adequate and thus are not returned by PROPBOUNDS. To compare the results to [24], we observed the 5 patterns with the highest divergence. All of them contain 3–5 attributes, with the value assignment sex=M, i.e., they are descendents of the pattern {sex=M} (in the pattern graph) returned by GLOBALBOUNDS. The pattern {sex=M} was also returned by [24], and it was ranked at 295 according to its divergence value.

VII. RELATED WORK

The notion of fairness in ranking algorithms was studied in a line of works, introducing different fairness definitions [11], [17], [27], [30], [32], [33]. These definitions typically focus on top-k positions, as those are usually the most important positions. In this paper, we consider two such definitions: the fundamental definition of [11], which measures fairness by bounding the representation of different groups in the data, and a refined definition that considers proportional groups representation. These definitions, as customary in the context of algorithmic fairness, refer to some given protected groups. We harness these definitions to define the problem of detecting groups with biased representation, eliminating the need to pre-define protected groups. The problem of generating fair ranking results was studied in [5], [33]. These works consider a wide range of definitions for fairness in ranking, which rely on the notion of protected groups. This line of work is orthogonal to the problem we defined in this paper, and our proposed method can be used to identify such protected groups, when they are unknown in advance.

Recent works have studied the problem of automatically detecting "problematic" or biased subgroups in the data, without the need to specify the protected attributes a priori, in the context of classification [10], [13], [24], [25]. In [25], the authors introduced the notion of divergence to measure the difference in the behavior of a classifier on data subgroups. The goal is then to report subgroups with sizes above a given threshold and high divergence. In [24] they extend their framework to ranking, where they consider the average outcome value, which is defined based on the ranking of the instances in each group, as a measure of the group's outcome. In contrast to [24], our problem definitions rely on groups' representation in the top-k ranked items as fairness measures for ranking. We demonstrate the differences in the resulting groups identified by each definition in Secetion VI-D. The interactive system MithraCoverage [18] investigates population bias in intersectional groups. The notion of coverage is introduced to identify intersectional subgroups with inadequate representation in the dataset. Differently, in our work, we only report patterns with adequate representation in the data, but inadequate representation in the output of a ranking algorithm.

The use of Shapley values to provide explanations for ML models was studied in a line of works (see e.g., [21],

[31]). In these works, the Shapley values are computed for an *individual* input instance to a *classification or regression model*. The Shapley value of a feature is then interpreted as the contribution of the feature to the output of the given input. Differently, we are aiming at providing explanations for the representation of a *group of tuples* in the output of a *ranking algorithm*. In [25] the authors presented a method to measure the contribution of items to divergence of groups utilizing Shapley values. However, it considers only the contribution of attributes that are used to define the group. In contrast, our solution considers all attributes as possible explanations this requires an additional aggregation step in the computation of the Shapley values. As demonstrated in VI-C, the explanation is typically buried in the values of attributes used for ranking. Moreover, our adjustment of Shapely values to explain a ranking algorithm is novel.

Our baseline solution, utilizing a top-town search presented in Section IV-A is built on the algorithm presented in [6] (for a simpler problem), which in turn shares similar ideas to the Apriori algorithm [1], the Set-Enumeration Tree for enumerating sets in a best-first fashion [28], discovering functional dependencies [16], [23] and frequent item-sets and association rule mining [2]. Similarly, to [6], the key difference from our work lies in the structure of the graph traversed in the solution: the pattern graph (in our case) compared to the powerset lattice in the other works. We then present two novel optimized algorithms designed for each one of the problems we defined. These algorithms reduce the search space as explained in Section IV and significantly outperform the baseline solution as shown in the experimental evaluation.

VIII. CONCLUSION

In this paper, we have studied the problem of detecting groups with biased representation in the result of a ranking algorithm. We build on fairness measures previously defined in the literature, considering the representation of protected groups in the top-k ranked items, for any reasonable range of k. Our problem definitions eliminate the need to pre-define the protected groups. We consider two variants of the problem. The first is based on global bounds over the representation of different groups in the top-k ranked items, and the second restricts the representation of each group in the top-k, based on its overall representation in the data.

We theoretically analyse the complexity of the problem, showing that in the worst case, the number of groups can be exponential in the number of the dataset attributes. We present a baseline algorithm that can handle both definitions and two optimized algorithms designed to improve the performance for each fairness measure. Furthermore, we present a method to explain the output of our algorithms. There are many intriguing directions for future research, including the extension of the framework to support other fairness measures and further investigation of the automatic suggestion for thresholds.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*. Morgan Kaufmann, 1994.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, pages 487–499. Morgan Kaufmann, 1994.

[3] Kristen M. Altenburger, Rajlakshmi De, Kaylyn Frazier, Nikolai Avteniev, and Jim Hamilton. Are there gender differences in professional self-promotion? an empirical case study of linkedin profiles among recent MBA graduates. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017*, pages 460–463. AAAI Press, 2017.

[4] Abolfazl Asudeh, H. V. Jagadish, Julia Stoyanovich, and Gautam Das. Designing fair ranking schemes. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 1259–1276. ACM, 2019.

[5] Abolfazl Asudeh, HV Jagadish, Julia Stoyanovich, and Gautam Das. Designing fair ranking schemes. In *Proceedings of the 2019 International Conference on Management of Data*, pages 1259–1276, 2019.

[6] Abolfazl Asudeh, Zhongjun Jin, and H. V. Jagadish. Assessing and remedying coverage for a given dataset. In *ICDE*, 2019.

[7] Solon Barocas and Andrew D Selbst. Big data's disparate impact. *Calif. L. Rev.*, 104:671, 2016.

[8] Tobias Berg, Valentin Burg, Ana Gombović, and Manju Puri. On the rise of fintechs: Credit scoring using digital footprints. *The Review of Financial Studies*, 33(7):2845–2897, 2020.

[9] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Networks*, 30(1-7):107–117, 1998.

[10] Ángel Alexander Cabrera, Will Epperson, Fred Hohman, Minsuk Kahng, Jamie Morgenstern, and Duen Horng Chau. Fairvis: Visual analytics for discovering intersectional bias in machine learning. In *VAST*, 2019.

[11] L. Elisa Celis, Damian Straszak, and Nisheeth K. Vishnoi. Ranking with fairness constraints. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 28:1–28:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[12] Le Chen, Ruijun Ma, Anikó Hannák, and Christo Wilson. Investigating the impact of gender on rank in resume search engines. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*, page 651. ACM, 2018.

[13] Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijong Whang. Automated data slicing for model validation: A big data - AI integration approach. *IEEE Trans. Knowl. Data Eng.*, 32(12):2284–2296, 2020.

[14] Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance. 2008.

[15] Sahin Cem Geyik, Stuart Ambler, and Krishnaram Kenthapadi. Fairness-aware ranking in search & recommendation systems with application to linkedin talent search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 2221–2231. ACM, 2019.

[16] Ykä Huhtala, Juha Kärkkäinen, Pasi Porkka, and Hannu Toivonen. TANE: an efficient algorithm for discovering functional and approximate dependencies. *Comput. J.*, 42(2):100–111, 1999.

[17] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[18] Zhongjun Jin, Mengjing Xu, Chenkai Sun, Abolfazl Asudeh, and HV Jagadish. Mithracoverage: a system for investigating population bias for intersectional fairness. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 2721–2724, 2020.

[19] Caitlin Kuhlman, MaryAnn Van Valkenburg, and Elke A. Rundensteiner. FARE: diagnostics for fair ranking using pairwise error metrics. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2936–2942. ACM, 2019.

[20] Jinyang Li, Yuval Moskovitch, and H. V. Jagadish. DENOUNCER: detection of unfairness in classifiers. *Proc. VLDB Endow.*, 14(12):2719–2722, 2021.

[21] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4765–4774, 2017.

[22] Shira Mitchell, Eric Potash, Solon Barocas, Alexander D'Amour, and Kristian Lum. Algorithmic fairness: Choices, assumptions, and definitions. *Annual Review of Statistics and Its Application*, 8:141–163, 2021.

[23] Thorsten Papenbrock, Jens Ehrlich, Jannik Marten, Tommy Neubert, Jan-Peer Rudolph, Martin Schönberg, Jakob Zwiener, and Felix Naumann. Functional dependency discovery: An experimental evaluation of seven algorithms. *Proc. VLDB Endow.*, 8(10):1082–1093, 2015.

[24] Eliana Pastor, Luca de Alfaro, and Elena Baralis. Identifying biased subgroups in ranking and classification. *arXiv preprint arXiv:2108.07450*, 2021.

[25] Eliana Pastor, Luca de Alfaro, and Elena Baralis. Looking for trouble: Analyzing classifier behavior via pattern divergence. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1400–1412, 2021.

[26] Christopher Peskun, Allan Detsky, and Maureen Shandling. Effectiveness of medical school admissions criteria in predicting residency ranking four years later. *Medical education*, 41(1):57–64, 2007.

[27] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. Fairness in rankings and recommendations: an overview. *VLDB J.*, 31(3):431–458, 2022.

[28] Ron Rymon. Search through systematic set enumeration. In *KR*. Morgan Kaufmann, 1992.

[29] L Shapley. 7. a value for n-person games. contributions to the theory of games ii (1953) 307-317. In *Classics in Game Theory*, pages 69–79. Princeton University Press, 2020.

[30] Ashudeep Singh and Thorsten Joachims. Fairness of exposure in rankings. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 2219–2228. ACM, 2018.

[31] Erik Strumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.*, 41(3):647–665, 2014.

[32] Ke Yang and Julia Stoyanovich. Measuring fairness in ranked outputs. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management, Chicago, IL, USA, June 27-29, 2017*, pages 22:1–22:6. ACM, 2017.

[33] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. Fa* ir: A fair top-k ranking algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1569–1578, 2017.

[34] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. Fa*ir: A fair top-k ranking algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pages 1569–1578. ACM, 2017.