MithraRanking: A System for Responsible Ranking Design

Yifan Guan*, Abolfazl Asudeh†, Pranav Mayuram‡, H. V. Jagadish§, Julia Stoyanovich¶, Gerome Miklau‖, Gautam Das**

*,†,‡,§University of Michigan; ¶New York University; ‖University of Massachusetts Amherst; **UT Arlington
{yfguan,asudeh,pranavm,jag}@umich.edu; stoyanovich@nyu.edu; miklau@cs.umass.edu; gdas@uta.edu

ABSTRACT

Items from a database are often ranked based on a combination of criteria. The weight given to each criterion in the combination can greatly affect the ranking produced. Often, a user may have a general sense of the relative importance of the different criteria, but beyond this may have the flexibility, within limits, to choose combinations that weigh these criteria differently with an *acceptable region*. We demonstrate MithraRanking, a system that helps users choose criterion weights that lead to "better" rankings in terms of having desirable properties while remaining within the acceptable region. The goodness properties we focus on are stability and fairness.

KEYWORDS

Data Ethics; Fairness; Stability; Transparency; Robustness; Linear Evaluators

ACM Reference Format:
Yifan Guan, Abolfazl Asudeh, Pranav Mayuram, H. V. Jagadish, Julia Stoyanovich, Gerome Miklau, Gautam Das. 2019. MithraRanking: A System for Responsible Ranking Design. In *2019 International Conference on Management of Data (SIGMOD '19), June 30-July 5, 2019, Amsterdam, Netherlands,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3299869.3320244

SIGMOD '19, June 30-July 5, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5643-5/19/06…$15.00
https://doi.org/10.1145/3299869.3320244

1 INTRODUCTION

We often need to rank items based on more than one criterion. One common way to accomplish this is to assign a score to each item as a weighted sum of attribute values, for each attribute that represents a criterion of interest. Thereafter, items are easily ranked based on scores.

Weighted linear combinations of attribute values are straightforward to compute and easy to understand [4]. However, the specific weights chosen have a huge impact on the score and hence rank for an item: the way the attributes are combined into the score determines the ranking, and may highly impact decisions that take these rankings into account. The decisions may in turn impact the lives of individuals and even influence societal policies. For this reason, as argued in [11], it is essential to make the development and deployment of rankings transparent and otherwise principled.

Many sports use ranking schemes. An example is the FIFA World Ranking of national soccer teams based on recent performance. FIFA uses these rankings as "a reliable measure for comparing national A-teams" [7]. Despite the trust of FIFA in these rankings, there have been many critics who question their validity. University rankings is another example that is both prominent and often contested [8]: various entities, such as U.S. News and World Report, Times Higher Education, and QS, produce such rankings. Similarly, many funding agencies compute a score for a research proposal as a weighted sum of scores of its attributes. These rankings are, once again, impactful, yet heavily criticized.

A more serious example is the use of *risk tools* in the criminal justice system: Judges in many US jurisdictions consider recidivism scores assigned to individuals, computed based on their criminal record and background, as guidance when sentencing or setting bail. While little is known about how these systems operate, a prominent example, The Public Safety Assessment (PSA) — used in courts in several US states — computes the score of an individual as a weighted sum of features[1]. An investigation by ProPublica showed that

[1]https://www.psapretrial.org/about/factors

recidivism scores can exhibit strong racial bias, based on analyzing another commonly used tool, COMPAS [1].

In criminal justice a risk tool is used to assist in making a decision for one individual at a time, and so, strictly speaking, systems like PSA and COMPAS cannot be considered rankers. However, COMPAS was originally intended to provide services and positive interventions, under resource constraints. That is, a score computed by COMPAS would then be used to rank individuals to prioritize access to services. Risk tools are also commonly used in domains like lending and online advertising, where ranking is important.

Many other impactful examples can be mentioned, such as a company that evaluates its employees to promote some and let go some others, and a college admissions officer who decides to admit a small portion of the applicants.

Surprisingly, despite the enormous impact of score-based rankers, attribute weights are usually assigned in an ad-hoc manner, based only on intuitive reasoning and commonsense of the human designers. For instance, in the case of FIFA rankings, the scoring formula combines the past four years of performance of each team as $x_1+0.5x_2+0.3x_3+0.2x_4$, where x_i is the team's performance in the past i^{th} year. Of course, the designers tried to come up with a set of weights that make sense. For them $0.98x_1 + 0.51x_2 + 0.29x_3 + 0.192x_4$ would probably be equally acceptable, since the weight values are virtually identical: they choose the former formula simply because round numbers are more intuitive.

In a recent paper [2] we showed that small changes in attribute weights may impact the ranking of some teams. In the case of FIFA rankings, at least the formula is fixed and does not change over time. In other cases, scoring formula weights may change over time. For example, U.S. News chooses slightly different weights for university rankings from year to year, raising concerns about whether the scoring formula is meaningful, or whether it is deliberately tuned. Malcolm Gladwell nicely described this issue in [8].

This demonstration presents *MithraRanking*, a system for responsible ranking design. MithraRanking provides a user interface in which the user can (i) identify a dataset of items to be ranked, (ii) set up the goodness criteria, (iii) provide a weight vector as the initial ranking function, and (iv) specify an acceptable range of functions, in the form of a region of interest in weight space. Then, the system investigates the generated ranking in terms of the specified goodness criteria and, if needed, makes suggestions (within the region of interest) that better satisfy the desired goodness criteria.

The MithraRanking framework is designed to be extensible to accommodate a wide variety of goodness metrics. In the current system, we have focused on two specific classes of properties: fairness and stability.

Fairness is a complex concept, with a number of different possible definitions. We consider group fairness with respect to membership in a protected group, based, for example, on minority race or underrepresented gender, where group membership is readily ascertained by looking at an attribute value [9]. For a given rank cut-off point k, we wish to ensure that the number of protected group members ranked among the top-k is proportional to their representation in the entire population, or to their desired proportion in the output (as is the case in affirmative action interventions).

Stability of a ranking specifies that slight changes to attribute weights in the scoring formula should not significantly perturb the ranked order. We worry about unstable rankings because such rankings are not robust, and so may be prone to tuning and manipulation by a vendor.

We studied fairness and stability properties of score-based rankers in depth in [3] and [2], respectively. We leverage the algorithms developed in these papers for MithraRanking.

The key technical idea is to express each scoring function as a point in a multi-dimensional weight space. In [3] and [2] we show how to efficiently identify regions in this space that satisfy fairness and stability criteria. Using this identification method, our system is able to tell users whether their proposed scoring function satisfies the desired criteria and, if it does not, to suggest the smallest modification that does. We use sampling exploration and Monte Carlo estimation algorithms for generating these suggestions. Our extensive experiments on real datasets demonstrate that our methods are able to find solutions that satisfy fairness and stability criteria effectively (usually with only small changes to proposed weight vectors) and efficiently (in interactive time, after some initial pre-processing).

In the following, we first describe the architecture and the UI of MithraRanking, followed by a demonstration plan.

2 SYSTEM DETAILS

2.1 Architecture and Implementation

MithraRanking is a Web application. The user uploads a dataset, or chooses among available datasets, and specifies fairness criteria and ranking attributes. She then identifies a region of interest (as an initial weight vector and a cosine similarity bound). The system then ranks the data based on the specified ranking function and checks if the ranking satisfies the fairness criteria. Then, it draws unbiased function samples (using the method in [2]) from the region of interest to estimate the stability of the generated ranking. It also uses the samples for finding the most stable rankings in the region of interest, the "closest" fair function in the neighborhood of the input function, and a function (not necessarily the closest) that generates a fair and more stable ranking. The

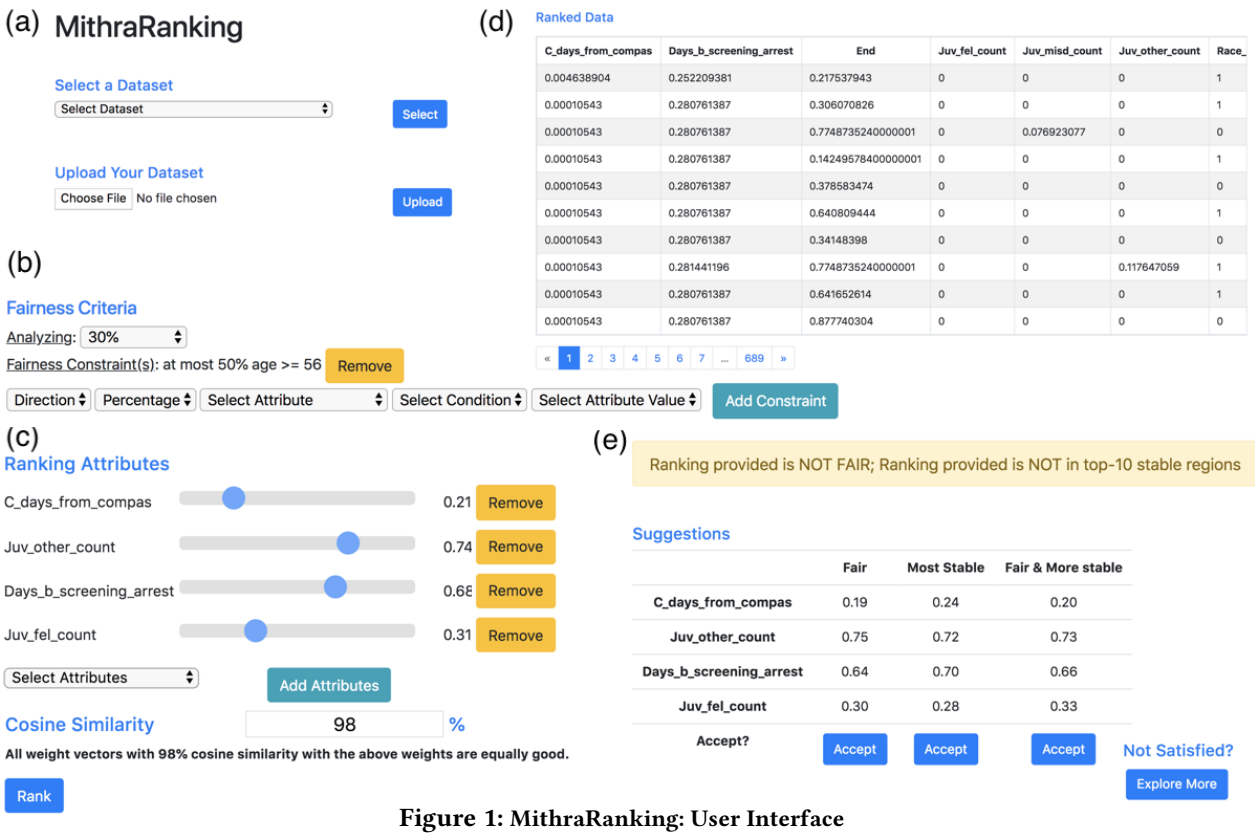

Figure 1: MithraRanking: User Interface

user can then accept any of those suggestions and change the ranking accordingly.

It is possible that no acceptable function is found even after exhausting the sampling budget, either because such a function does not exist in the specified region, or because the function exists but the sampling did not find it. The system allows the user to (i) try again by taking more samples, (ii) increase the region of interest by reducing the cosine similarity requirement between the input and suggested functions, (iii) change the input function to a different meaningful function, (iv) relax the fairness criteria, or (v) conclude that the dataset is not a good fit for this task.

MithraRanking is implemented using *Python 2.7.10*. To build the Web service, we used *Flask framework*, due to its flexibility and fine-grained control. We used *Pandas* and *NumPy* libraries to process datasets. On the client side, *AngularJS 1.6.9*, *Bootstrap 4.0.0*, *jQuery*, *HTML*, and *CSS* are used to parse results from the back-end and to render pages.

2.2 User Interface

MithraRanking has four main sections in its user interface, which we describe in turn below.

Dataset Selection section: The first thing a user has to do is to specify the dataset of interest. Each tuple in this dataset is an item to be ranked, based on the values of its attributes. As shown in Fig. 1(a), users can either select a dataset from our pre-loaded demonstration datasets or upload their own dataset to design their ranking schemes.

Fairness Criteria section: This is the section where users define the fairness criteria. In line with prior work [6, 9, 10], we define fairness with regard to *sensitive attributes*, which denote membership of individuals in legally-protected categories, such as people with disabilities, or under-represented minorities by gender or ethnicity. While our techniques apply to a broad range of fairness criteria [3], in our system, we define fairness in terms of minimum/maximum bounds on the number of selected members of a protected group at the top-k, for some reasonable value of k [5]. Fig. 1(b) shows the fairness criteria section. The user first should identify the value of k for analyzing fairness (it is 30% in the figure). Then the user can add multiple fairness criteria by selecting (a) the direction to be at most/at least, (b) the percentage, (c) a sensitive attribute, (d) the criteria condition ($=$, $\neq$, $<$, $>$, $\geq$, or $\leq$), and (e) an attribute value. For instance, in the figure the added fairness criterion is "at most 50% with age$\geq$ 56 in top 30%", or the ranking is not fair.

Ranking section: As demonstrated in Fig. 1(c), users can add ranking attributes from drop-down menus and assign weights to them using sliders. If they want to change ranking attributes, they can click the "remove" button to delete

unnecessary attributes. At the bottom of Fig. 1(c), users can specify a value of cosine similarity. This value defines a *"region of interest"* [2] around the specified ranking function (weight vector) within which all weights are equally acceptable. For example, the value 98% in the figure indicates that all weight vectors that have at least 98% cosine similarity with the specified weight vector are within the region of interest. This region is considered for computing the *stability* of the produced ranking by specified ranking, and to discover the most stable ranking. We also use this region for exploration to find a fair function in vicinity of the specified function, in case its output is not fair.

Ranking Results section: The ranking results section provides (a) the ranking generated by the specified function, (b) a signal indicating if the generated ranking is fair and/or stable, and (c) information about the suggested functions. The reference ranking is depicted in Fig. 1(d), in the form of a table containing tuples ranked based on the user-specified ranking function, with a pagination bar under the table for users to navigate through the table. The fairness and stability signals, as well as function suggestions are provided in Fig. 1(e). As shown in the figure, we list three suggestion options to the user: (i) the closest fair function, (ii) the closest function that generates the most stable ranking, and (iii) a function that is fair (not necessarily the closest) and more stable than the reference function. There are inherent trade-offs between fairness, stability, and distance from the user's original choice, If the system could not find proper functions, or the user is not satisfied with system's suggestions, she can "explore more" by taking more samples from the region of interest (of course the user can also apply changes in prior steps such as input function or region of interest and re-try).

3 DEMONSTRATION PLAN

MithraRanking is accessible at http://mithra.eecs.umich.edu/demo/ranking/. We will demonstrate it with real datasets:

- *COMPAS [1]:* a dataset collected and published by ProPublica as part of their investigation into racial bias in criminal risk assessment software. The dataset contains demographics, recidivism scores produced by the COMPAS software, and criminal offense information for 6,889 individuals.
- *CSMetrics*[2]*:* CSMetrics ranks computer science research institutions based on publication metrics. For each institution, a combination of `measured` (M) citations and an attribute intended to capture future citations, called `predicted` (P), is used for ranking, according the score function: $(M)^{\alpha}(P)^{1-\alpha}$, for parameter α. This score function is not linear, but under a transformation of the data in which $x_1 = log(M)$ and $x_2 = log(P)$ we can write an equivalent score function linearly as: $\alpha x_1 + (1-\alpha)x_2$. The CSMetrics website uses $\alpha = .3$ as the default value, but allows other values to be selected.
- *FIFA Rankings dataset [7]:* The FIFA World Ranking of men's national football teams is based on measures of recent performance. Specifically, the score of a team depends on its performance values for x_1 (current year), x_2 (past year), x_3 (two years ago), and x_4 (three years ago). FIFA's ranking function is: $x_1 + 0.5x_2 + 0.3x_3 + 0.2x_4$. FIFA relies on these rankings for modeling the progress of the national teams [7] and to seed important competitions in different tournaments, including the 2018 FIFA World Cup.

We use the COMPAS dataset for demonstrating fair and stable ranking design. Also, using CSMetrics and FIFA Rankings datasets, we first show that their reference rankings are not stable. Then, we demonstrate how to find stable rankings in the vicinity of their reference rankings.

[2] www.csmetrics.org

4 ACKNOWLEDGEMENTS

The work of Yifan Guan, Abolfazl Asudeh, Pranav Mayuram, and H. V. Jagadish was supported in part by NSF grants No. 1741022 and 1250880. The work of Julia Stoyanovich was supported in part by NSF grants No. 1926250 and 1916647. The work of Gerome Miklau was supported in part by NSF Grant No. 1741254. The work of Gautam Das was supported in part by grant W911NF-15-1-0020 from the Army Research Office, NSF grant No. 1745925, and a grant from AT&T.

REFERENCES

[1] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine Bias: Risk Assessments in Criminal Sentencing. *ProPublica* (23 5 2016).
[2] Abolfazl Asudeh, H.V. Jagadish, Gerome Miklau, and Julia Stoyanovich. 2018. On Obtaining Stable Rankings. *PVDLB* 12, 3 (2018), 237–250.
[3] Abolfazl Asudeh, H.V. Jagadish, Julia Stoyanovich, and Gautam Das. 2019. Designing Fair Ranking Schemes. In *SIGMOD*. ACM.
[4] Abolfazl Asudeh, Nan Zhang, and Gautam Das. 2016. Query reranking as a service. *PVLDB* 9, 11 (2016), 888–899.
[5] L. Elisa Celis, Damian Straszak, and Nisheeth K. Vishnoi. 2018. Ranking with Fairness Constraints. In *ICALP*.
[6] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *SIGKDD*.
[7] FIFA. 28 March 2008. FIFA/Coca-Cola World Ranking Procedure. http://www.fifa.com/fifa-world-ranking/procedure/men.html.
[8] Malcolm Gladwell. 2011. The Order of Things: What College Rankings Really Tell Us. *The New Yorker Magazine* (Feb 14, 2011).
[9] Julia Stoyanovich, Ke Yang, and H.V. Jagadish. 2018. Online Set Selection with Fairness and Diversity Constraints. In *EDBT*.
[10] Ke Yang and Julia Stoyanovich. 2017. Measuring Fairness in Ranked Outputs. In *SSDBM*. 22:1–22:6.
[11] Ke Yang, Julia Stoyanovich, Abolfazl Asudeh, Bill Howe, H. V. Jagadish, and Gerome Miklau. 2018. A Nutritional Label for Rankings. In *SIGMOD*. ACM.