A Simple Scoring Function to Fool SHAP: Stealing from the One Above

Jun Yuan
Department of Data Science
New Jersey Institute of Technology
Newark, NJ 07102
jy448@njit.edu

Aritra Dasgupta
Department of Data Science
New Jersey Institute of Technology
Newark, NJ 07102
aritra.dasgupta@njit.edu

Abstract

Explainable AI (XAI) methods such as SHAP can help discover unfairness in black-box models. If the XAI method reveals a significant impact from a "protected attribute" (e.g., gender, race) on the model output, the model is considered unfair. However, adversarial models can subvert the detection of XAI methods. Previous approaches to constructing such an adversarial model require access to underlying data distribution. We propose a simple rule that does not require access to the underlying data or data distribution. It can adapt any scoring function to fool XAI methods, such as SHAP. Our work calls for more attention to scoring functions besides classifiers in XAI research and reveals the limitations of XAI methods for explaining behaviors of scoring functions.

1 Introduction

Explainable AI (XAI) methods are increasingly used to detect unfairness in black-box machine learning models [2, 3]. For example, suppose the XAI method detects that "protected attributes" such as gender or race significantly contribute to a model's prediction. In that case, it may indicate that the model is unfair. Existing work shows adversarial classifiers can be constructed via scaffolding [9], which can fool explanations generated by LIME [8] and SHAP [7]. Specifically, LIME and SHAP cannot detect that the classifier is making decisions heavily influenced by the "protected attribute". However, a scaffolding procedure assumes we have access to the underlying dataset, which can be used to train an additional classification model. Such an assumption might not hold good in real-world scenarios where data accessibility is restricted.

In this work, we focus on scoring functions that assign probability or scores to data items, where the scores can either be used to assign items to different groups (in the case of a classifier) or put them in a particular order (in the case of an algorithmic ranker). Scoring functions represent a more fine-grained model output than the model predictions themselves. We can turn a scoring function into a classifier by setting thresholds on the score output (e.g., if the score is above a threshold, it belongs to class A, otherwise class B.). Also, when the input of the scoring function is a group of items rather than a single item, the resulting score vector can be used to generate a ranking (indicating the relative preference for the items). Hence, the scoring function can also be considered an intermediate component of an "algorithmic ranker" [12].

Many works have discussed the ways to judge the fairness of a classifiers [4, 5] or rankers [13, 6]. For example, if a classifier disproportionately puts more male loan applicants to approve than female applicants, we may consider such a classifier unfair to female applicants. However, there is a hidden scoring function in this example, which may be the scoring function that assigns the credit score to an applicant. If the scoring function is designed to give higher scores to

XAI in Action: Past, Present, and Future Applications @ NeurIPS 2023.

male candidates than to females, the female group is already disadvantaged. In this work, we experimented with using traditional XAI method (e.g., SHAP) to detect such unfair behaviors of the scoring function. We attempt to construct scoring functions that give higher scores to a certain group (e.g., male) than another group (e.g., female). In contrast to previous work [9], our adversarial construction does not require access to the underlying input data. We take inspiration from the previous work [10] using a rank-mixing approach in measuring the fairness of ranked outcomes. Let us take an example of a group of male and female candidates applying for bank loans. A scoring function may be applied to the male and female subgroups to create rankings of acceptance rate within each subgroup. To merge two rankings, a *biased mixer* will *flip an unfair coin that favors male group* to put the top-1 male or female candidate at the top-1 of the combined ranking. If the female top-1 is not chosen for the top-1 position, she will face the male top-2 and continue the mixing process. Until the candidates from one group are all positioned in the merged ranking, the remaining candidates from the other group will be appended at the end.

In this work, we focus not on the ranking but the scores, which represent the direct output of a scoring function. To make the score output favorable towards a group, we proposed a simple "bias decision-maker" that re-assigns the scores for the items. In other words, our "bias decision-maker" can be applied to the output of the *biased mixer* described above. We prove that *a scoring function with biased decision-maker* can fool SHAP into thinking: *the protected attribute has minimal impact on the prediction of the scoring function*.

2 Background and Methodology

Based on the notation from Slack et al. ([9]), SHAP and LIME are designed to generate explanations that: (1) approximate the behavior of the black-box model accurately within the vicinity of an input data point x, and (2) achieve lower complexity and are human interpretable. Generating an explanation is to find the q that satisfies:

$$\text{arg min}_{q \in Q} L(f, q, \pi_x) + \Omega(q) \tag{1}$$

where the loss function L is defined as:

$$L(f, q, \pi_x) = \sum_{x' \in X'} [f(x') - q(x')]^2 \pi_x(x') \tag{2}$$

f is the black-box model to be explained, $q \in Q$ where Q is the class of linear models. $\Omega(q)$ measures the complexity of the linear model (i.e., the number of non-zero weights). $\pi_x(x')$ is the proximity measures between input data x and x'. X' is the set of x' that describes the neighborhood of x. SHAP is built upon Shapley values that are grounded on cooperative game theory to find the unique model q that satisfies several desired characteristics (more details in [7]). The intuition is that we consider the expected model output as the total "payout" for each "game" (generating model outputs) played by a coalition of "players" (a subset of attributes), and distribute the payout to an individual "player" (attribute). Each attribute's contribution is the average of all the "payout" from the "game" they participated in.

There are different ways of defining "payout" for scoring function [11]. We use the kernel SHAP from the official SHAP python package as our choice of explainer since it is a widely used implementation.

Prior work [9] on adversarial attacking XAI methods such as LIME and SHAP reveal the vulnerability of post hoc explanation methods. In this work, we attempt to propose a simple rule of adversarial construction for scoring function that does not rely on the ground truth data, data distribution, or training a model to distinguish ground truth data from perturbed data.

We first describe our adversary, who can modify the model before hand over to the auditors. Both the auditors and the adversary treat the model as a black-box. The adversary only knows what the protected attributes (e.g., race, gender) existed in the data and add a rule to modify the output of the model based on the values of protected attribute to give advantage for certain group. The auditors have access to all the attributes including protected attributes and use SHAP to discover model unfairness of the model. However, the auditors does not know the model was modified by the adversary. The intuition of the adversary method is to strategically swap

scores of the output from a scoring function to give an unfair advantage to certain privileged groups. We define a simple weighted sum scoring function f as our fair model. For a given data that have attributes $\chi = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m, \mathbf{x}_p]$. We assume the $\mathbf{x}_p$ is a column vector of the protected attribute. The rest are columns of scoring attributes. A function that does not include the protected attribute is

$$f(\chi) : \mathbf{s} = \beta_1 \mathbf{x}_1 + \beta_2 \mathbf{x}_2 + \cdots + \beta_m \mathbf{x}_m \tag{3}$$

We assume the attributes in the scoring function is not correlated with protected attributes, so such scoring function is our fair model. We define that if the scoring function is unfair if it include the protected attribute. A simple unfair scoring function may be

$$g(\chi) : \mathbf{s}_{unfair} = f(\chi) + \beta_p \mathbf{x}_p \tag{4}$$

Note that in our definition, the weight β_p is non-zero. We demonstrated in our experiment section that SHAP can detect $\mathbf{x}_p$'s impact on the function output, $\mathbf{s}$. To design an unfair scoring function $g(\cdot)$ that can bypass SHAP's detection, we let $\mathbf{x}_p$ to influence the vector $\mathbf{s}$ during a post-process operation which resulted in a $\mathbf{s}_{unfair}$,

Algorithm 1 The adversarial scoring function g (based on gender as the protected attribute)

Input: Sample of dataset $\mathbf{X}$, protected attribute vector $\mathbf{p}$, scoring function f (a higher score outcome is superior)
Output: Unfair score vector $\mathbf{s}$

1: $\mathbf{s} \leftarrow f(\mathbf{X})$
2: $\mathbf{id} \leftarrow$ the indices of $\mathbf{s}$ ▷ start of the swapping function h_{swap}
3: $[\mathbf{id}, \mathbf{p}, \mathbf{s}] \leftarrow$ Sort($[\mathbf{id}, \mathbf{p}, \mathbf{s}]$) by $\mathbf{s}$ in descending order
4: $N \leftarrow$ length of vector $\mathbf{s}$
5: **for** $i = 0$ to $N - 2$ **do**
6: **if** $\mathbf{p}[i]$ is *female* and $\mathbf{p}[i+1]$ is *male* **then**
7: swap($\mathbf{p}[i], \mathbf{p}[i+1]$)
8: swap($\mathbf{id}[i], \mathbf{id}[i+1]$)
9: **end if**
10: **end for**
11: $\mathbf{s} \leftarrow$ Sort($[\mathbf{id}, \mathbf{s}]$) by $\mathbf{id}$
12: return $\mathbf{s}$

This algorithm1 describes a `biased decision-maker`, $Bias_{male>female}$. Whenever the `biased decision-maker` see a `female` candidate is scored higher than a `male` candidate (i.e., a `female` is ranked one position higher than a `male`), the `biased decision-maker` swaps the two candidates' scores (and consequently their rank positions are swapped), giving `male` candidates an unfair advantage. In this work, we consider *swapping scores to give advantage for historically privileged group as "stealing"*. We obtain the adversarial scoring function $g(\chi) = h_{\texttt{swap}}(f(\chi \backslash \mathbf{x}_p), \mathbf{x}_p)$. An XAI method that can explain f can also be used to explain g, since both function g and f take a sample of χ as input, and output a score vector.

Variations of the unfair scoring function. Such a biased decision-maker can be formulated beyond the two-class gender scenario to race, age, or combining multiple protected attributes into the if-condition (line 6 of the Algorithm1). It can also be formulated with more complicated rules for "stealing" the score (add at the beginning within the if-condition), such as "stealing" is only successful half the time , "stealing" only happens a maximum amount of times, or "stealing" is more active in the high-score region and less active in the low-score region. Note that $f(\cdot)$ can be generalized to non-linear functions or black-box machine-learning models. The swapping operation (line 2 to line 11 in Algorithm1) within g does not require prior knowledge on data or the data distribution. The only information needed is the unique values in the protected attribute to construct the if-condition. Our method is both data distribution agnostic and model agnostic.

Edge cases of the unfair scoring function. The swapping operation $h_{\texttt{swap}}$ within g may be triggered zero or many times. With the example of $Bias_{male>female}$, if all `male`'s scores are larger than `female`'s, no swapping is performed. If the data only come from `female` or `male` group, no swapping is also performed. In a different case, if there is only one `female` in the data who scored highest (i.e., at the top-1 rank position) among other `male` candidates, the `female`

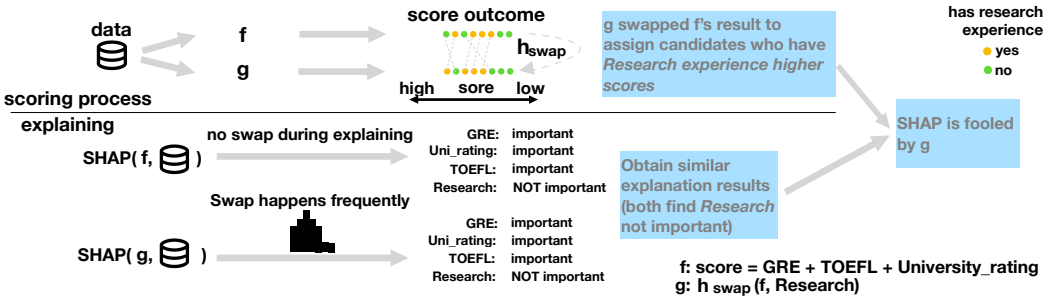

Figure 1: **The workflow of fooling SHAP**. The scoring process is to reveal that function g successfully alters the score outcome of function f to give higher scores to candidates' with "Research" experience; The explaining process is to reveal that SHAP cannot detect that function g is using the attribute "Research" to generate the score outcome and consider the behavior of functions f and g the same. Combining the two observations, function g fooled SHAP.

candidate's score will be swapped repeatedly and ends up at the bottom of the ranking with the lowest score. But if we remove the line 7 in Algorithm1, the `female` candidate in such case will only be swapped once. Hence, the more swapping (or "stealing") happens, the more protected attributes impact the model output, and the more unfair the scoring function g is.

Intuition behind the attack to fool SHAP. We describe the intuition behind the attack under a special case of SHAP, i.e. linear SHAP. In such a case, we assume the input feature independence. And SHAP values can be approximated directly from the model's weight coefficients [7]. Since our adversarial attack is constructed on a group of N input x. We use the matrix form of the formulas. For fair scoring function:

$$f(X) = \sum_j \beta_j X_j + 0 \cdot X_p \tag{5}$$

The contribution matrix Φ given the function f and input X contains the vectors:

$$\Phi_j(f, X) = \beta_j X_j - \beta_j E[X_j], \quad \Phi_p(f, X) = \mathbf{0} \tag{6}$$

Due to the additive constraint of SHAP values, instance-wise feature contributions add up to the difference between the model output and the average model output,

$$\Phi_p(f, X) = \mathbf{0} = f(X) - E[f(X)] - \sum_j \Phi_j(f, X) \tag{7}$$

We apply the swapping function h_{X_p} on the output vector of $f(X)$ to obtain $g(X)$. $h_{X_p}(\cdot)$ can be omitted if all values in X_p is the same:

$$g(X) = h_{X_p}(f(X)) = h_{X_p}(\sum_j \beta_j X_j) \tag{8}$$

Since all the features are assumed independent, to obtain feature j's contribution, we can set zero for all other features in X including X_p but not X_j, In such condition, no swapping occurs (i.e., h_{X_p} does not modify the output of f) since the values in X_p are all the same (zeros), and function g is equivalent to function f, The feature j's contribution vector Φ_j follows,

$$\Phi_j(g, X) = \Phi_j(f, X) \tag{9}$$

Due to the same additive constraint of SHAP values, the SHAP value vector of the protected attribute $X_p(g, X)$ can be calculated as:

$$\Phi_p(g, X) = g(X) - E[g(X)] - \sum_j \Phi_j(g, X) \tag{10}$$

We can tell that $E[g(X)]$ is the same as $E[f(X)]$ since swapping the output of f does not alter summary statistics such as mean or deviation. Subtracting equation 10 and equation 7, we get:

$$\Phi_p(g, X) = g(X) - f(X) \tag{11}$$

If swapping occurs between f's output with index i and k that are close to each other in values, meaning

$$|g(x_i) - f(x_i)| = |g(x_k) - f(x_k)| \leq \epsilon \tag{12}$$

the $\phi_p(g, x_i)$ may be negligible when ϵ is close to zero. And if no swapping occurs,

$$|g(x_i) - f(x_i)| = 0 \tag{13}$$

The mean of absolute SHAP value for protected feature p is

$$(1/N)\mathbf{1}^T|\Phi_p| \leq \epsilon \tag{14}$$

If some of f's output remain the same after swapping, the bound will be strictly smaller than ϵ. In general, it is possible that SHAP's loss function 2 (replacing the f with g) may neglect a $O(\epsilon)$ in order to obtain a linear additive explanation with lower complexity 1. Thus, we discover that swapping scores can, in certain degree, fool the linear SHAP. However, whether such swapping can fool the kernel SHAP is yet to be tested.

3 Experimental Results

We demonstrate that our construction of adversarial scoring function can fool SHAP using a real-world data set on college admissions [1] with about 500 college applicants described by 7 different attributes, such as GRE score, CGPA, TOEFL score, Letter of Recommendation rating, Statement of Purpose rating, having Research Experience or not, and Rating of the University. We select three scoring attributes (*GRE*, *TOEFL*, *university rating*) and one protected attribute (*Research*). We consider that students from prestigious schools may have more opportunities to participate in research, which may be unfair to students from under-funded schools. Hence, in this context, *Research* is a protected attribute and should be excluded from scoring the candidates.

Experiment settings. We define three scoring functions (Figure 2): base function f, adversarial unfair function g_1, and simple unfair function g_0 for sanity check. The experiments for generating SHAP explanation for 100 candidates ran less than 10 seconds for f and g_0, but ran for about 1 hour for g_1. For the 100 candidates' data, we tracked that g_1 performed 59 score swaps based on the result of f. We visualized the attribute's importance using the bar plot and beeswarm plot provided by SHAP Python package. For g_1, we also test different variations, such as "flip a coin" before committing a score swap, or checking if the score difference between two candidates is smaller than a threshold before committing a score swap. The purpose of such variations are to create more unstable behaviors of "stealing" to confuse SHAP.

Score outcome from different functions. We first need to show that the scoring outcomes from f, g_0, and g_1 are different on the ground truth data. We use the parallel coordinates plots (Figure 2(iv)) with scores from each function as the axes. Scores of 100 candidates are sorted on the axes from the highest to lowest, and the connected lines across the axes indicate the same candidate. The students *who have* or *do not have Research experience* are colored yellow and green, respectively.

The green lines between axis f and g_1 appear either parallel or downward, while the yellow lines appear either parallel or upward. This is not a coincidence but a clear illustration of the relationship between the score outcome between f and g_1, that is the students without research experience are given lower scores. The line pattern between axis g_1 and g_0 appears as expected as well. It shows that even more green lines (i.e., students without research experience) are given lower scores compared to the other group.

The reason is that the g_0 is an unfair function that uses the attribute "Research" specifically in the scoring function, but g_1 only uses the "Research" in a stealth way. Such stealth usage of the protected attribute (i.e., "Research") is why g_1 may fool SHAP but g_0 cannot.

SHAP explanation result. After SHAP explains all three functions f, g_1, and g_0, the explanations are visualized correspondingly. For function f, SHAP shows (Figure 2(i)), in bar plot, the *University Rating*, *GRE score*, *TOEFL score* have the SHAP value of 0.24, 0.22, and 0.2, while *Research* has 0 SHAP value. The beeswarm plot also shows that *Research*'s instance-wise SHAP cluttered near 0 SHAP value at the x-axis, while other attribute's SHAP value spread across both positive

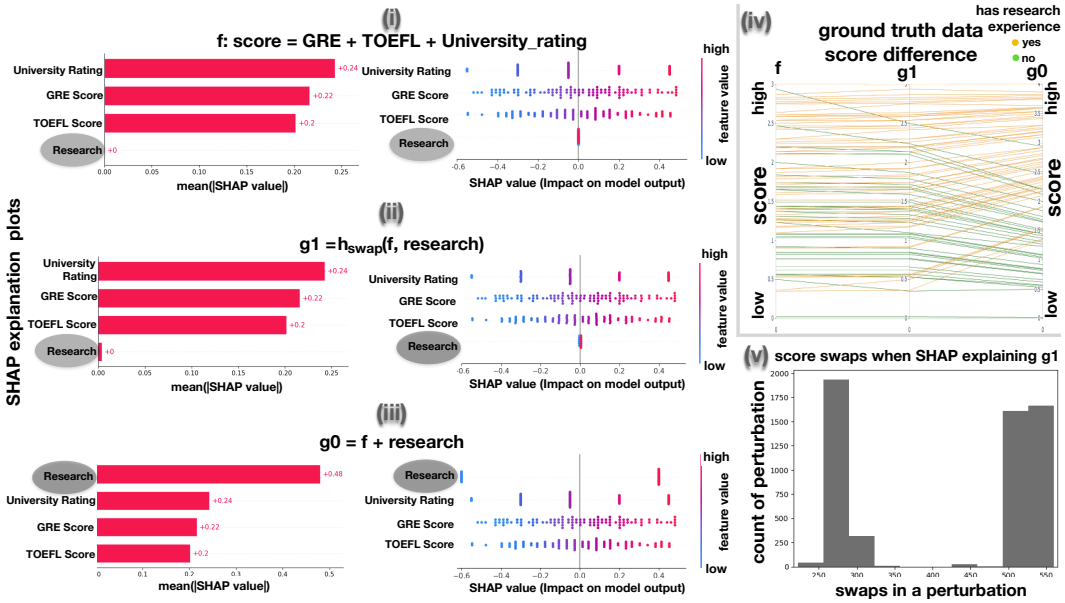

Figure 2: **Experimental results** show that SHAP explanation of adversarial scoring function g_1 and the base scoring function f are similar. Hence g_1 successfully fooled SHAP. g_0 is also explained as a sanity check to prove: if Research is specifically added into the scoring formula, SHAP can detect that Research is the biggest contributor to score output; the slope plot shows that for the ground truth data, g_1 and g_0's score output give candidates with research experience (yellow group) advantage compare to f's score output; the score swap histogram shows the score swaps during SHAP explaining for g_1 (note that no swaps happened for f or g_0).

and negative range. The red and blue color indicates high and low attribute values, which means, for all the attributes besides *Research*, the higher the attribute value, the higher the SHAP value. Such explanation plots generated by SHAP are aligned with the definition of function f.

For function g_1, SHAP shows (Figure 2(ii)), in bar plot, the exact same SHAP values. SHAP still considers the *Research* has zero SHAP value, although there is a very short bar associated with *Research*, which may be easily neglected. The SHAP beeswarm doesn't reveal any sign of *Research* being an important attribute either. Since the swapping function within g_1 can be triggered unknown time during the SHAP explaining. We tracked the times of swapping that happened during each SHAP perturbation and summarized the results in a histogram (Figure 2(v)). The histogram shows that the swapping happens significantly during the SHAP explaining and the most commonly happened 275 times or 575 times within one perturbation. With that many times of swapping, SHAP still failed to detect it. During our experiments, other more complicated if-conditions for swapping do not appear more capable of fooling SHAP, because the original swapping function already achieved zero SHAP value. Note that, in our experiments, more complicated if-conditions do not increase the SHAP explaining time either.

We also used SHAP to explain the function g_0 as a sanity check to demonstrate that if *Research* is added into the scoring function specifically, SHAP will detect the importance of *Research*. SHAP shows (Figure 2(iii)), in bar plot, the SHAP value for *Research*, *University Rating*, *GRE score*, *TOEFL score* are 0.48, 0.24, 0.22, 0.2. *Research* is two times more important than the second most important attribute. The SHAP beeswarm plot also shows that *Research* only has two attribute values (0 and 1). And 0 is associated with a SHAP value of -0.6, and 1 is associated with 0.4. In conclusion, SHAP can successfully detect the base (f) and simple unfair (g_0) scoring function but is unable to detect the adversarial unfair scoring function (g_1). If AI model auditors rely on SHAP to detect unfair scoring functions, they will be misguided to consider the adversarial scoring function (g_1) as a fair scoring function. Additionally, the SHAP bar plot of g_1 attempted to show a short bar for *Research* to indicate its importance, while the beeswarm plot is not helpful to show such importance at all. On the other hand, if AI model auditors use the parallel coordinates plots to test the scoring function behavior, they may have a better chance of discovering the unfair behavior.

4 Discussion

We proposed a novel approach to construct an adversarial scoring function from any scoring function that can fool the SHAP explainer into thinking attributes that significantly impact the model output are not important. Our approach does not need to access the input data or data distribution; hence is robust against data distribution shifting or data volume increasing. Our work demonstrated simpler ways to construct adversarial scoring functions to fool XAI methods such as SHAP, compared to prior work [9]. Additionally, our adversarial scoring function could be more difficult to detect if constructed more stealthily. This can be achieved by adding additional conditions prior to the swapping operation; for example, the two candidates' score difference has to be lower than a given threshold.

Our observations point to the fact that the swapping operation is triggered frequently but not consistently when SHAP repeatedly generates sample data to test the function g. Additionally, for the same input data, the outputted scores from the adversarial function g all occur in the corresponding outcome from the baseline function f, but only may be rearranged. Since the explanation time for g_1 is significantly long (about 1 hour for 100 instances), SHAP might attempt to find a consistent pattern between *Research* and the score outcome, yet still failed. SHAP is "confused" about whether the protected attribute impacts the function output.

XAI research is currently focused on classifiers, and does not pay enough attention to XAI methods for scoring functions. However, scoring functions are ubiquitous in AI systems, including classifiers, algorithmic rankers, activation functions in neural networks, etc. Studying XAI for scoring functions may lead to the less explored models, such as algorithmic rankers and ranking-based decision-making systems in traditional applications (e.g., college rankings) or AI-ranking (e.g., search engine output). Our work calls for XAI researchers and practitioners to tread cautiously while conducting model auditing, and raise concerns about using the XAI method as a means to confirm a model is fair. Since it is relatively low-cost (demonstrated in our case) and simple to inject unfairness and give certain groups unfair advantage whenever a swapping condition is met, such unfair behavior can even be hidden for a long time after model deployment and only automatically triggered by certain critical if-conditions. And for our adversarial scoring function, the unfair behavior will not be detected by SHAP or any other method, if only used in scoring individual group (e.g., only male or female group), which raises the need for designing a proper AI auditing process.

5 Conclusion

Our work demonstrated that the original swapping condition is powerful enough to fool SHAP. It is still an open problem in XAI research to develop robust detection methods for more complicated if-conditions. Currently, we only considered swapping between two nearby items. Still, it can be easily generalized to swapping between nearby items for which the behaviors are not explored. We have not yet explored the cascading impact of our unfair scoring function. In the real world, it is common that a higher-scored candidate may receive additional advantages (e.g., getting the job offer) which leads to advantages in future scoring (e.g., approval of bank loan). A small "stealing" initially may result in huge future differences. Our work reveals the risk of over-reliance on default explanation visualizations such as SHAP bar plot or beeswarm plot to understand attribute importance. Alternative visualizations may be used to detect model behaviors, such as multivariate visualizations, like a matrix of scatter plots, parallel coordinate plots, etc., for scoring functions. Our work also opens up another way of explaining model unfair behaviors: *one group is stealing from another in a certain way*. In the future, we will design XAI methods to generate such an explanation. We will consider swapping scores to promote candidates from non-privileged groups and investigate these fairness-preserving interventions in conjunction with established metrics [14].

Acknowledgments and Disclosure of Funding

This work was funded by the National Science Foundation (NSF) grant 2312932.

References

[1] Mohan S Acharya, Asfia Armaan, and Aneeta S Antony. A comparison of regression models for prediction of graduate admissions. In *2019 international conference on computational intelligence in data science (ICCIDS)*, pages 1–5. IEEE, 2019.

[2] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

[3] Arun Das and Paul Rad. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*, 2020.

[4] Abigail Z Jacobs and Hanna Wallach. Measurement and fairness. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 375–385, 2021.

[5] Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019.

[6] Yunqi Li, Hanxiong Chen, Shuyuan Xu, Yingqiang Ge, Juntao Tan, Shuchang Liu, and Yongfeng Zhang. Fairness in recommendation: A survey. *arXiv preprint arXiv:2205.13619*, 2022.

[7] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[9] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.

[10] Ke Yang and Julia Stoyanovich. Measuring Fairness in Ranked Outputs. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, pages 1–6, Chicago IL USA, June 2017. ACM. ISBN 978-1-4503-5282-6. doi: 10.1145/3085504.3085526.

[11] Jun Yuan and Aritra Dasgupta. A human-in-the-loop workflow for multi-factorial sensitivity analysis of algorithmic rankers. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, pages 1–5, 2023.

[12] Jun Yuan, Kaustav Bhattacharjee, Akm Zahirul Islam, and Aritra Dasgupta. Trivea: transparent ranking interpretation using visual explanation of black-box algorithmic rankers. *The Visual Computer*, pages 1–17, 2023.

[13] Meike Zehlike, Ke Yang, and Julia Stoyanovich. Fairness in ranking: A survey. *arXiv preprint arXiv:2103.14000*, 2021.

[14] Meike Zehlike, Ke Yang, and Julia Stoyanovich. Fairness in ranking, part i: Score-based ranking. *ACM Computing Surveys*, 55(6):1–36, 2022.