Using Bayes Factors to Control for Fairness Case Study on Learning To Rank

Swetasudha Panda, Jean-Baptiste Tristan, Michael Wick, Haniyeh Mahmoudian, Pallika Kanani
Oracle Labs Burlington MA,{first.last@oracle.com}

Abstract

The ability to detect and mitigate bias and unfairness is becoming increasingly important especially as machine learning continues to play an increasing role in finance applications. In this paper we propose the use of Bayes factors to audit for disparate impact in lending decisions, so we can (1) handle small data gracefully and determine whether or not there is sufficient evidence to deem a system as unfair and (2) provide a continuous measure that can form the basis of a control system that throttles the aggressiveness of a post-training correction in order to keep the overall system fair as the system continually operates on newly available data in an online fashion. We perform studies of these Bayes factor algorithms on both real-world and synthetic ranking problems in order to understand their behavior. We find that they are better equipped to handle small data, and, unlike previous work, are able to operate in an environment in which we care not just about the fairness of a single ranking, but a collection of them.

1 Introduction

When machine learning is used for decision making in regulated domains such as credit scoring and lending, we need to ensure that the system is 'fair' according to the rules set forth by the relevant regulatory bodies. Discrimination on the basis of any protected class is illegal under the US Equal Credit Opportunity Act of 1974. If the attributes of protected classes are not directly considered, but the model still exhibits bias, it results in a 'disparate impact' on those classes. Indeed, in the remainder of this paper, fairness (and its dual, bias) will not refer to the subjective concept but rather some measure of discrepancy between the behavior of a learning algorithm and some mathematical rule imposed by an external party. Our intent is not to prescribe what should be considered fair, but rather to study how a concrete definition of fairness can be made to work in a realistic scenario. In such a scenario, we have a number of tools at our disposal to satisfy a fairness rule [8].

Given the variety of tools, the variety of fairness properties, and the known limits of fairness composibility [7], what can we expect of the overall application? Unfortunately, there is no guarantee that a best effort to ensure that all the parts of the application meet our external fairness criteria will provide us with a reasonable protection against the bias of the end application as a whole. This is particularly problematic since the number of machine learning applications is rapidly increasing and auditing such applications requires expertise. In software security and safety, an important concept is that of the *trusted computing base* (TCB). The TCB correspond to all the parts of the application that enforce some safety policy and whose failure could jeopardize the safety of the overall application. In our previous example, the TCB would include all of the pre/in/post-processing measures deployed. Such a large TCB is not only difficult to manage and diagnose, it is also error prone.

In light of all these challenges, we propose a post-training countermeasure as a 'post-processor' that takes the entire application as a black-box and acts as a *control system*, like a thermostat for fairness. This is especially relevant since credit score generation e.g. is often a black-box model. The post-processor acts as a safety net and ensures that the overall application satisfies a desired property. This allows us to reduce the TCB to comprise only such a control system. Methods for addressing fairness such as improving the data or using fair algorithms can still be used in the application, as

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

improvements in the overall fairness of the system will lead to the control system making fewer corrections in the output of the system.

In this paper, we detail a possible design for such a system in the context of a ranking application, which is quite relevant in several finance applications. For concreteness, consider an application to target people for special loan offers. Many such applications involve ranking the customers based on their scores. Such advertisement could potentially be biased leading to further disparities in lending decisions. To be general, the principles of our work apply to any other ranking system whose output must comport with externally-imposed fairness rules. We consider the 4/5th rule which is widely used to detect disparate impact in employment decisions. It is common for such an application to work in two phases: first, *select* a subset of the input to be ranked, second, *rank* such subset. For example, in the lending application, the first phase could select the top 10 candidates and the second phase would rank them. Our system thus decomposes into two stages accordingly: the first stage focuses on the 4/5th rule and the second stage on ensuring that all permutations of the protected feature have the same probability. Note that our approach can work equally well with a classification system if we focus exclusively on the selection part.

So far, what we have described is fairly common sense but unfortunately, it is unlikely to be practically useful. For such a system to be useful, it absolutely must handle small data gracefully. Consider for example the 4/5th rule, and consider for example and for concreteness the specific case of gender for the categories "male" and "female". If we have one male and one female candidate and decide to select one of them, the selection rate for one category will be 1 for one gender and 0 for the other, and the naive application of the 4/5th rule rejects a lending decision which in itself cannot be construed as statistical evidence that the 4/5th rule is not met. In practice, such small data cases happen frequently and handling them properly is a firm prerequisite for the adoption of fairness methods.

To address this challenge, we propose to use Bayes factors [15] — a form of hypothesis testing that accounts for the inherent uncertainty in small data — to redefine the tests in such a way that we account for the amount of evidence supporting some fairness property. Moreover, the Bayes factor serves as a continuous measure of fairness that can drive aggressiveness of the post-training control-system in order to adapt to varying levels of unfairness encountered throughout the lifetime of the system. Experiments on both real world and synthetic datasets demonstrate that the Bayes factor is a useful measure of fairness both for the purpose of auditing, and the control system itself. We also investigate the corner-cases of the proposed Bayes factors to better understand the failure modes.

2 Preliminaries and Notation

Our fairness control system monitors the output of a black-box ranker which generates outputs that we term *experiments*. The control system employs some definition of fairness against which it monitors the output (via the tests in Section 3). If sufficient evidence for unfairness is detected, appropriate action, i.e., modifying the output, is taken by the system (via the post-processing in Section 4). More precisely, the input to the control system is a set of N *experiments* $(\mathcal{P}_i, g_i, \alpha_i, \beta_i)$ in which each experiment i comprises a pool of applicants $\mathcal{P}_i$, its associated protected attributes g_i (e.g., 1 if the candidate is female and 0 if the candidate is male), the associated selection α_i which maps every candidate from $\mathcal{P}_i$ to 1 or 0 to indicate whether they are recommended or not and ranking β_i which maps some candidates from $\mathcal{P}$ to an integer describing its position in the ranking, so if a candidate p is listed at position k then $\alpha(p) = k$. For concreteness we will assume that the protected feature is gender and that we have two classes, male and female.

We consider the aggregate pool of candidates $\mathcal{P} = \cup \mathcal{P}_i$ and without loss of generality we consider that all the $\mathcal{P}_i$ are disjoint. For all $p \in \mathcal{P}_i$ we can define the aggregate functions $g = g_i(p)$, $\alpha(p) = \alpha_i(p)$, $\beta(p) = \beta_i(p)$. The pool can be partitioned between the male candidates $\mathcal{P}_m = \{p \in \mathcal{P} \mid g(p) = 0\}$ and the female candidates $\mathcal{P}_f = \{p \in \mathcal{P} \mid g(p) = 1\}$. We define $\mathcal{P}^s = \{p \in \mathcal{P} \mid \alpha(p) = 1\}$ for the set of selected candidates and $\mathcal{P}^r_m = \mathcal{P}^s \cap \mathcal{P}_m$ and $\mathcal{P}^r_f = \mathcal{P}^s \cap \mathcal{P}_f$ for the selected males and selected females respectively.

3 Using Bayes Factors to Audit Ranking Fairness

In this section we propose an improved test for ranking fairness. There are many existing proposals for such tests from which we take inspiration, but we build on them to solve two additional problems.

The first problem is that we want to recognize the fact that in many cases, the result of a ranking is displayed on a web application in which only the top k candidates appear. Moreover, the user of such an application will usually go through the ranked candidates from top to bottom, and the position within the ranking is critical. Of course, previous work defines a notion of visibility for the various positions but we want to go one step further and avoid the dilema of defining such a visibility function. *Our solution* is to define two distinct tests. The first test ensures that the selection process for which candidates appear in the top k satisfies demographic parity (in particular the 4/5th rule). The second test ensures that the set rankings represent all the permutations of protected feature. This is a very strong notion of fairness which allows us to avoid using a visibility function.

The second problem is that for such tests to be useful and relevant, we need to account for the amount of evidence that fairness might be violated. As mention before, a test like the 4/5th test has been criticized for that very reason. *Our solution* is to account for the amount of evidence that supports that the ranking is not fair by making the assumption that it is fair and measuring the evidence to reject such assumption using Bayes factors. Once again, to clarify, when we talk about "measuring" discrimination, we refer to a specific mathematical definition and some concrete measure of the extent to which it may not be satisfied.

3.1 Selection Fairness

The first test is for the fairness of the selection process, for which we employ the 4/5th rule, a well-known relaxation of demographic parity in which the selection rate of some protected group must be at least 4/5ths of the most prominent group. Let the random variables Y_i denote the selection/rejection for each female candidate i and $\bar{Y}$ the vector of all such variables. To measure the evidence that the 4/5th rule might be violated, we define a Bayes factor $b_{a0} = P_a(\bar{Y})/P_0(\bar{Y})$ where $P_0(\mu, \bar{Y})$ is the null hypothesis model that captures that the selection rate of the female candidates μ is fair, and $P_a(\mu, \bar{Y})$ is the alternative hypothesis that captures that the selection is biased. Let μ^* be the selection rate of the males, the null model (left) and the alternative model (right) are defined as follow

$$\mu \sim U\left(\frac{4}{5}\mu^*, 1\right) \qquad (1)$$

$$Y_i \sim Bern(\mu) \qquad i \in [N] \qquad (2)$$

$$\mu \sim U\left(0, \frac{4}{5}\mu^*\right) \qquad (3)$$

$$Y_i \sim Bern(\mu) \qquad i \in [N] \qquad (4)$$

Neither the evidence of the null model $P_0(Y_i)$ nor that of the alternative $P_a(Y_i)$ have a closed form solution so we use a Monte Carlo integration, which works well in practice because the space is only two dimensional.

For M samples we can estimate the probability as

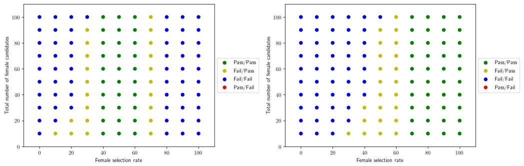
$$P_0(\bar{Y}) \approx \frac{1}{1 - 4/5\mu*}\frac{1}{M}\sum_{j}^{M}\prod_{i}^{N} P(Y_i|\mu_j) \qquad \mu_j \sim U(4/5\mu, 1) \qquad (5)$$

and the details of the derivation are in the appendix.

The Bayes factor can then help us decide whether or not there is enough evidence to reject the null hypothesis in favor of the alternative. Moreover, since the measure is continuous, it is highly suitable for guiding a post-processor.

How is this test behaving? In experiments 1 and 2 we compare the Bayes factor test to the classic 4/5ths test to detect unfairness (a Bayes factor greater than 100 is considered a decisive reject of the null [13]). In each the plot, the horizontal axis shows the female selection rate and the vertical axis the total number of female candidates. The color indicates how the two tests are doing: green means they agree that there is no discrimination, blue that they agree that there is discrimination, yellow that the 4/5th rule failed and that our test passed, and red the opposite. We run the experiments for different selection rates S for males and different total number of male candidates T. We show additional selection rates shown in the appendix in Figures 7 & 8, which are also consistent with the following conclusions.

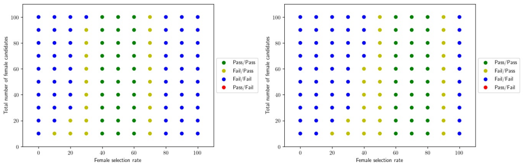
There are a few important things to note about these results. **First**, if the 4/5th test pass, then our test passes as well. This is important as we do not strive to be stricter than the 4/5th test. **Second**, if the 4/5th test fails, then our test on experiment 2 almost always fail. It might be more desirable for it to always fail, but actually this means our test is far less brittle and indeed, when the selection rate is above 0.7, it tends to pass. It is easy to change our acceptance criteria to ensure that if the 4/5th test fails with large number of examples, then our test fails as well. **Third**, the most interesting case is when the 4/5th test fails but our test passes on experiment 1. It is exactly what we are trying to capture, that is, when the evidence is small, we do want to be more lenient if there isn't statistical evidence to support the alternative theory.



(a) S = 50%

(b) S = 80%

Figure 1: T = 100



(a) S = 50%

(b) S = 80%

Figure 2: T = 500

3.2 Top-K Ranking Fairness

We now define a test for the top k ranking. The idea of this test is that if we look at the different permutations of the protected features proposed by the ranking, it should not be the case that some permutations are much more likely than others. To clarify, it is not simply the case that the protected features should have equal probability of appearing at any position in the ranking since this fails to account for class imbalance.

As mentioned previously, we assume that each experiment has k recommended candidates, that is, $|\{p \in \mathcal{P}_i \mid \alpha_i(p) = 1\}| = k$ for all i. Consider a permutation π of length k over the objects $\{0, 1\}$ where a of them are 0 and b of them are 1 (and therefore $a + b = k$). We say that a permutation π_{ab} *matches* a ranking β_i and write $\beta_i \models \pi$ if for all $j \in \{1, \ldots, k\}$ we have $\pi_{ab}(j) = g_i(\beta_i^{-1}(j))$ where β_i^{-1} is the inverse of β_i which always exists. Given a permutation π, let $\#\pi = |\{\beta_i \mid \beta_i \models \pi\}|$ the number of rankings that match the permutation. Write Π_{ab} for the set of permutations of length k with a 0 and b 1. We define the empirical probability $\hat{p}$ of a permutation $\pi \in \Pi_{ab}$ as $\hat{p}(\pi) = \frac{\#\pi}{\sum_{\pi\in\Pi_{ab}} \#\pi}$. Consider a specific pair (a, b) such that $a + b = k$, the corresponding set of permutations Π_{ab} and the set $\mathcal{O}_{ab}$ of rankings that match such permutations that is $\mathcal{O}_{ab} = \left\{g_i \circ \beta_i^{-1} \mid \beta_i \models \pi \wedge \pi \in \Pi_{ab}\right\}$.

Just as before, we now define two models, a null and an alternative to capture whether there is evidence of discrimination or not. Our first hypothesis correspond to a model where we assume that each $o_i \in \mathcal{O}_{ab}$ is drawn i.i.d. from a uniform distribution of size $|\Pi_{ab}|$. Our second hypothesis is that the o_i are drawn i.i.d. from a categorical distribution whose parameter is distributed according to a Dirichlet measure μ parameterized by a vector ρ.

$$o_i \sim U(1, |\Pi_{ab}|) \tag{6}$$

$$\theta \sim \text{Dirichlet}(\rho) \tag{7}$$

$$o_i \sim \theta \tag{8}$$

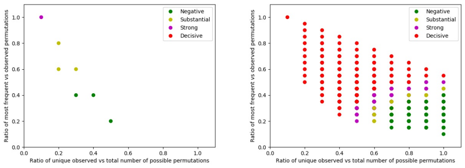
Note that we can modify this test to study the data by modifying the probability of the models or the parameters of the Dirichlet distribution. The Bayes factor K_{ab} of the two models is therefore $\frac{\int_\theta \mu(\theta)\ \prod_{\pi\in\mathcal{O}_{ab}} \theta_\pi d\theta}{\prod_{\pi\in\mathcal{O}_{ab}} \theta_\pi}$ where where

$$\int_\theta \mu(\theta) \prod_{\pi\in\mathcal{O}_{ab}} \theta_\pi \, d\theta = \frac{\Gamma(\sum_{\pi\in\Pi_{ab}} \rho_\pi)}{\prod_{\pi\in\Pi_{ab}} \Gamma(\rho_\pi)} \frac{\prod_{\pi\in\Pi_{ab}} \Gamma(\#\pi + \rho_\pi)}{\Gamma(\sum_{\pi\in\Pi_{ab}} \#\pi + \rho_\pi)} \tag{9}$$

In this case, the two models are in closed form.

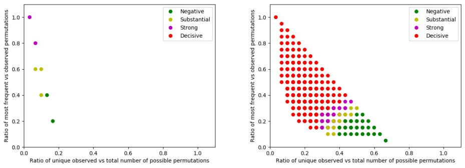
How is the test behaving? In these experiments (Figure 3), we run the Bayes factor test on sets with N observed permutations. Some sets of permutation are more unfair than others, and we quantify this using two indicators: how many of the 10 permutations are observed (x axis, small is less fair) and what fraction of the overall observed ranking the most frequent one represents (y axis, larger is less fair). We use colors yellow, purple, and red if the log-Bayes factor is greater than log(2), log(10), and log(100) respectively. Otherwise, we report green. These values are commonly used in analyzing Bayes factors. Each subplot reports on a different amount of observed data. Figure

4 reports on the same type of experiments but where we instead use 3 categories with occurence 2,2, and 1. We generally find that the test detects unfairness and becomes increasingly decisive with more samples and more unfairness. For a more detailed explanation, see Appendix B.2.



(a) 5 observations (b) 20 observations

Figure 3: 2 categories



(a) 5 observations (b) 20 observations

Figure 4: 3 categories

Before discussing post-processing, consider that while using Bayes factors gives a convenient "measure" of discrimination, it is crucial to understand the "failure mode" of a specific choice of model. This is particularly the case if the Bayes factor is used to make decisions (which in our case is not since it only serves to control the post-processor smoothly). For that reason, not only do such tests need to be thoroughly evaluated experimentally, but it is also useful to design and run a variety of "adversarial" tests, and to compare different models. We report on such experiments in the appendix.

4 Post-Training Countermeasure

We propose a two-stage post-training countermeasure (henceforth the 'post-processor') that optimizes over the continuous Bayes factors of the two tests to improve the fairness. The post-processing algorithm is designed to respect the original ranking as much as possible while getting us closer to passing the tests. Therefore, the objective is to satisfy the following conditions: a) there should not be any post-processing when the test(s) pass, b) any changes introduced by post-processing should improve fairness in terms of the Bayes factor, c) it should not be possible to exactly recover the changes introduced by post-processing for various reasons including privacy, and d) the notion of maximizing utility: within a given protected group, ordering must be preserved as in the original ranking, i.e., the distance (according to the metrics we discuss below) between the original and the post-processed recommendations should be minimized. In practice, post-processing should operate in an online setting since it is not possible to change any rankings that were observed in the past time steps and there is no knowledge about future rankings. Therefore, the objective is to improve fairness by making changes only to the current ranking, while utilizing information from the set of rankings observed in the past time steps. Below we describe each stage in more detail, followed by the algorithm which incorporates the two post-processing steps in an online setup.

4.1 Passing the Selection Test

In order to satisfy the selection test, the Bayes factor for this test must indicate that there is insufficient evidence for unfairness in the way in which the top k candidates are selected. Ideally, one would minimize the log Bayes factor (k_{ab}) corresponding to the selection test. Note that this is a discrete optimization problem over the number of candidates from each protected group in the top k recommendations. To solve this problem, we leverage structure in the search space by using the 4/5th rule as a surrogate condition. To explicitly incorporate the Bayes factor information, we add a slack parameter in the 4/5th constraint which is computed as a function of the log Bayes factor. Below we describe this process in more detail.

Let $\mathcal{B} = \{\beta_1, \ldots, \beta_{j-1}\}$ be the set of rankings that have been observed up to time point $t-1$, which is the first time-step at which the test detects unfairness. The rankings up to this point are a sunk cost in terms of fairness since we cannot go back in time and modify them. However, our post-processor can begin improving each subsequent ranking beginning with the current ranking β_j. Let $\alpha_j^{'}$ be the vector where each position indicates the probability of a candidate being selected into top k, after post processing. Let $\beta_j^{'}$ be the post-processed ranking. The objective is then to compute $\alpha_j^{'}$ so as to minimize a certain distance metric from the original ranking $\beta_j^{'}$. Let s_i be a vector of the relevance scores of the candidates. We define this distance $d(\beta_j, \beta_j^{'})$ as the difference from the original total score $\sum_{p \in \text{top}k} s_{jp}\alpha_{jp}$ for the top k recommendations. Note that $d(\beta_j, \beta_j^{'})) \geq 0$ because recommending a candidate which was not in top k in the orginal recommendation results in a decrease in the sum of scores over the top k recommendations. Below, we formulate the problem as the following integer linear program (ILP):

$$\begin{aligned}
\underset{\alpha'_j}{\text{minimize}} \quad & \sum_{p \in \mathcal{P}_j} [s_{jp}\alpha_{jp} - s_{jp}\alpha'_{jp}] \\
\text{subject to} \quad & \sum_{p \in \mathcal{P}_j} \alpha'_{jp} = k, \alpha'_{jp} \in \{0, 1\} \forall p \qquad \text{(10a)} \\
& \frac{\sum_{\beta_i \in \mathcal{B}} \sum_{p \in \mathcal{P}_i} \alpha'_{ip}(1 - g_{ip})}{\sum_{\beta_i \in \mathcal{B}} \sum_{p \in \mathcal{P}_i} (1 - g_{ip})} - \frac{4}{5} \frac{\sum_{\beta_i \in \mathcal{B}} \sum_{p \in \mathcal{P}_i} \alpha'_{ip} g_{ip}}{\sum_{\beta_i \in \mathcal{B}} \sum_{p \in \mathcal{P}_i} g_{ip}} \geq \epsilon_1 \qquad \text{(10b)}
\end{aligned}$$

where the first set of constraints enforce that there are exactly k selected candidates in each job posting. The second constraint is the 4/5 rule which is defined according to the majority group. In the case where males form the majority, the left term computes the (aggregated) selection rate of females given α'_j and the right term computes that for the males, ϵ_1 is a variable that represents the deviation from satisfying the constraint. When there are more than two categories in the protected field, we add one such constraint for each category that is in the minority. The conditions for feasibility of ILP 10 are in the appendix. To retrieve the ranking β'_j from α'_j, we sort the selected applicants by their relevance scores.

4.2 Passing the Ranking Test

In order to satisfy the second stage of the test, the possible permutations of the candidates' protected attributes should be equiprobable across the rankings, within the top k recommendations. Let Π_{ab} be the set of permutations for a given number of male and female applicants in the top k recommendation. Let ϕ_j be a vector that assigns a probability of transforming an observed permutation o in top k to a new permutation $\pi \in \Pi_{ab}$. We formulate this problem as a linear program (LP) in which the objective is to respect the original ranking by minimizing the distance to it and to simultaneously improve fairness by minimizing the total variation distance (l_1-norm for countable sets) to the uniform distribution over the possible permutations . We define the distance metric as the edit distance between an observed permutation and the outcome permutation: $d_{j\pi} = k - \sum_{p=1}^{k} o_p \pi_p$. The LP is the following:

$$\begin{aligned}
\underset{\phi_j}{\text{minimize}} \quad & \sum_{\pi \in \Pi_{ab}} \phi_{j\pi} d_{j\pi} \\
\text{subject to} \quad & \frac{1}{2} \sum_{\pi \in \Pi_{ab}} \left| \phi_{j\pi} - \frac{1}{|\Pi_{ab}|} \right| \leq \epsilon_2, \sum_{\pi=1}^{|\Pi_{ab}|} \phi_{j\pi} = 1, 0 \leq \phi_{j\pi} \leq 1, \forall \pi \qquad \text{(11)}
\end{aligned}$$

where the first constraint computes the distance from the uniform distribution and ϵ_2 is a slack parameter which can be adjusted according to the Bayes factor for the ranking test. Note that when $\epsilon_2 = 0$, the solution is the uniform distribution. For a given permutation, the LHS in the absolute difference in the first constraint evaluates to $\frac{1}{2}\left|1 - \frac{1}{|\Pi_{ab}|}\right|$. Consequently, the solution is the original permutation when $\epsilon_2 \geq \frac{1}{2}\left|1 - \frac{1}{|\Pi_{ab}|}\right|$ (note that the ranking might still undergo a change because of the randomization). When $0 < \epsilon_2 < \frac{1}{2}\left|1 - \frac{1}{|\Pi_{ab}|}\right|$, the solution lies between the above two boundary cases. To obtain the top k recommendations, first, we sample a permutation from Π_{ab} according to the distribution in ϕ_j. Next, we map this permutation into a ranking by assigning back the scores so as to preserve the order within each protected group. The element of randomization at this stage enables us to achieve the post-processing objective (c) as introduced earlier in this section.

4.2.1 Online Post-Training Correction

Here we summarize the two staged approach into an online algorithm (Algorithm 1). The algorithm iteratively adjusts the applicants selected into the top k recommendations based on the test results in each iteration. We define the functions PERFORMTEST which computes the relevant Bayes factors for the two stages of the fairness tests and POSTPROCESS that re-ranks a given ranking β_j according to set deviations ϵ_1 and ϵ_2 which in turn are decided based on how well the given set of rankings perform with respect to the fairness tests, i.e., $\epsilon_1, \epsilon_2 = f(k_{ab})$.

Algorithm 1. Online Post-processing

```
Given set of observed rankings $\mathcal{B} = \{\beta_1, \ldots, \beta_{j-1}\}$ and ranking $\beta_j$ at iteration $j$:
for iteration $j$ in number of iterations do
    for selection test and then ranking test do
        $k_{ab} \leftarrow$ PERFORMTEST($\mathcal{B} \cup \beta_j, \alpha, g, s, \mathcal{P}$)
        if $k_{ab} \geq \log(100)$ then                    ▷ evidence of discrimination is decisive
            $\epsilon = f(k_{ab})$; $\beta_j \leftarrow$ POSTPROCESS($\beta_j, \epsilon$, test)
    $\mathcal{B} = \mathcal{B} \cup \beta_j$
function POSTPROCESS($\beta_j$, $\epsilon$, test)
    if test is selection then
        $\alpha'_j$: solution to ILP 10; obtain the top $k$ recommendations in $\beta_j$
    else if test is ranking then
        $\phi_j$: solution to LP 11, on the top $k$ recommendations
        sample the top $k$ recommendation according to $\phi_j$
    return the post-processed ranking $\beta_j$
end
```

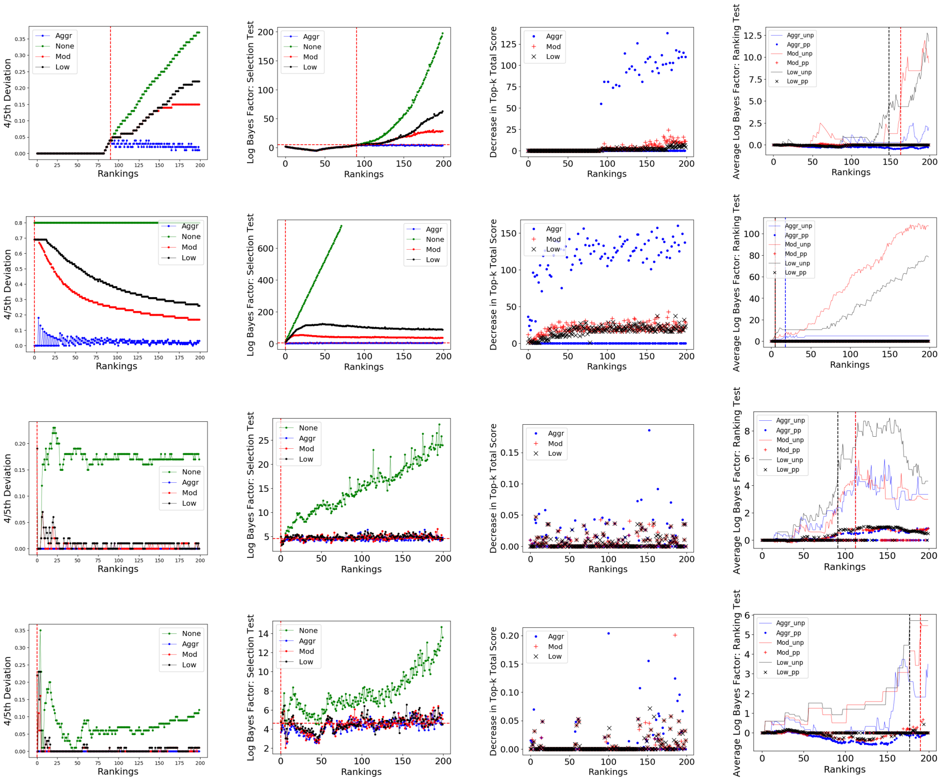
4.3 Evaluation

We evaluate the proposed post-processing Algorithm 1 on simulations and on two real-world datasets a) German credit and b) Census income. In each case, the protected attribute is gender with two groups: male and female. The ranking simulator generates ranking experiments with a given proportion of the protected group applicants in the top k recommendations. We generate 200 such samples, each with 20 candidates in each group corresponding to a) extreme bias: all top k recommendations belong to the same group (male) and b) increasing bias: the proportion of applicants from each protected group in top k increases linearly from 0.5 (fair) to 1 (biased) over the ranking samples. On the public datasets, we learn a logistic regression model and use the predictions as the relevance scores in case of ranking. We generate 200 random samples each with 100 candidates. A direct comparison with one of the existing fair ranking literature is not feasible because of the difference in objectives (details in section 5). The experiments are run on an 8-core Ubuntu Linux machine with 64 GB RAM, using CVXPY [9] with an embedded conic (branch and bound) solver for the LP instances. We set the test decision threshold as decisive for the log Bayes factor. In each experiment, we execute the post-processor with varying degrees of aggression, decided by the appropriate ϵ parameters (details in the appendix). Accordingly, we denote most aggressive as ‘Aggr’, moderate as ‘Mod’, least aggressive as ‘Low’ and no post-processing as ‘None’.

The experiment results are summarized in Figure 5. In each case, we compare the post-processed top k recommendations to the original. First, we evaluate in terms of the amount of deviation from the (aggregated) 4/5 rule, defined as 0 when the 4/5 condition is satisfied and as $1 - (\text{minority selection rate}/\text{majority selection rate})$ when the condition is violated. Second, the magnitude of the log Bayes factor is displayed in Figure 5 (b). At a high level we observe expected behavior: there is no post-processing when the recommendations are fair and the post-processor is triggered when the test fails. Third, the distance objective of ILP 10 in 5 (c) decreases as the intensity of post-processing decreases. Moreover, the aggressive case acts sporadically and introduces relatively larger changes in the rankings whereas the milder versions act continuously and bring about smaller changes at each ranking. Finally, 5 (d) shows the average the log Bayes factor for the ranking test, before and after post-processing (more details in the appendix).

5 Related Work

Fairness in Machine Learning has received a lot of attention recently [6, 10, 14, 11, 12, 1]. While a large proportion of this research focuses on supervised classification, the idea of fairness in ranking is relatively less explored, despite the fact that Learning to Rank [17, 5, 16] models are central to a large number of practical applications in Information Retrieval. Yang and Stoyanovich [20] propose several logarithmic discounting based measures for fairness inspired by nDCG, which is a popular ranking quality measure in Information Retrieval. Asudeh et al. [2] also propose a method for



(a) (b) (c) (d)

Figure 5: Evaluation of the post-processor with varying degrees of intensity on simulated recommendations with extreme bias, linearly increasing bias and recommendations from German credit and Census income data (from top row to bottom row) in terms of a) 4/5 deviation, b) log Bayes factor: selection test, c) utility; ILP objective and d) average log Bayes factor: ranking test. Vertical lines indicate the first iteration when the test(s) detect discrimination. Horizontal line in (a) and (b) indicates the decisive threshold.

checking a ranking against multiple fairness criteria, but their approach is also limited to a single ranking. In contrast, our approach provides a more robust framework for testing fairness across rankings. Singh et al. [18] extend the idea of fairness based on position by introducing the concept of exposure, We do not consider a visibility function. Also, our post-processor is $O(n)$, where n is the size of applicant pool in a ranking, whereas their approach is $O(n^2)$ plus polynomial time for the matrix decomposition. Zehlike et al. [21] operates on a series of rankings in an online fashion. but their ranked group fairness criterion looks at the proportion of protected elements at each position in the ranking.Biega et al. [4] propose the idea of amortized fairness, but the analysis is on individual fairness. Some in-processing approaches in ranking are [22] and [19].

6 Conclusion

We design a control system that audits and enforces a concrete fairness rule for a learning to rank application in lending decisions. Our post-training countermeasure enforces such fairness rule more or less aggressively based on a measure of fairness that we define using Bayes factors. We use a modification of the 4/5th rule to audit any disparate impact with respect to which candidates appear on the final top k ranking. We use a second test to ensure that all the permutations of the protected features are equally likely to occur. The use of Bayes factors to compute the evidence allows to handle small data gracefully, a feature that is very important in practice. Also, the Bayes factor provides us with a continuous measure of discrimination to control the intensity of the post-training correction. Finally, our approach can be adapted to work with a classification system by auditing and correcting for selection fairness only.

References

[1] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. *arXiv preprint arXiv:1803.02453*, 2018.

[2] Abolfazl Asudehy, HV Jagadishy, Julia Stoyanovichz, and Gautam Das. Designing fair ranking schemes. *arXiv preprint arXiv:1712.09752*, 2017.

[3] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

[4] Asia J Biega, Krishna P Gummadi, and Gerhard Weikum. Equity of attention: Amortizing individual fairness in rankings. *arXiv preprint arXiv:1805.01788*, 2018.

[5] Christopher J. C. Burges. From ranknet to lambdarank to lambdamart: An overview, 2010.

[6] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, 21(2):277–292, 2010.

[7] Alexandra Chouldechova and Aaron Roth. The frontiers of fairness in machine learning. *arXiv*, 2018.

[8] Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023*, 2018.

[9] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[10] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.

[11] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.

[12] Moritz Hardt, Eric Price, Nati Srebro, et al. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pages 3315–3323, 2016.

[13] Harold Jeffreys. *The theory of probability*. OUP Oxford, 1998.

[14] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1):1–33, 2012.

[15] E. Robert Kass and Adrian E. Raferty. Bayes factors. *Journal of the American Statistical Association*, 90(430):773,795, 1995.

[16] Hang Li. A short introduction to learning to rank, 2011.

[17] Filip Radlinski and Thorsten Joachims. Query chains: Learning to rank from implicit feedback. In *In ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD*, 2005.

[18] Ashudeep Singh and Thorsten Joachims. Fairness of exposure in rankings. *arXiv preprint arXiv:1802.07281*, 2018.

[19] Ashudeep Singh and Thorsten Joachims. Policy learning for fairness in ranking. *CoRR*, abs/1902.04056, 2019.

[20] Ke Yang and Julia Stoyanovich. Measuring fairness in ranked outputs. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, page 22. ACM, 2017.

[21] 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. ACM, 2017.

[22] Meike Zehlike and Carlos Castillo. Reducing disparate exposure in ranking: A learning to rank approach. *arXiv preprint arXiv:1805.08716*, 2018.

A Fairness Selection Test

In this section, we present the different models d for the phase 1 of the fairness test. We introduced three different models which all employ the 4/5th rule. In the first model, v1, we define the alternate (12) and null (13) model as shows below

$$\begin{aligned} \mu_1 &\sim Beta\,(1,1) \\ X_i &\sim Bern(\mu_1) && i \in [N] \\ \mu_2 &\sim Beta\,(1,1) \\ Y_i &\sim Bern(\mu_2) && i \in [N] \end{aligned} \tag{12}$$

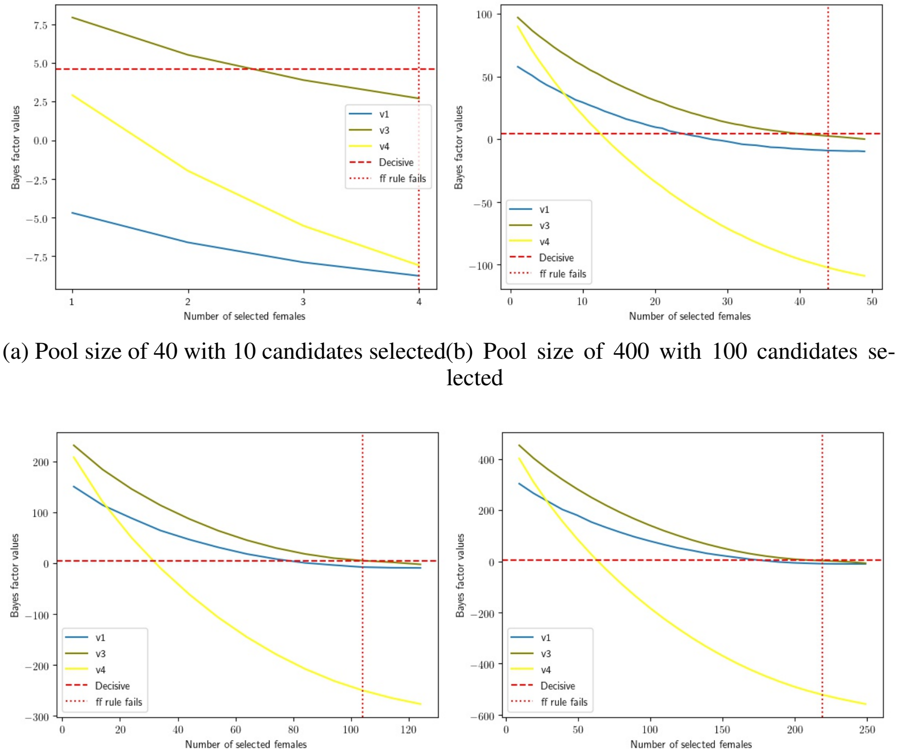
$$\begin{aligned} \mu_1 &\sim U\,(1,1) \\ X_i &\sim Bern(\mu_1) && i \in [N] \\ \mu_2 &\sim U\left(\frac{4}{5}\mu_1, 1\right) \\ Y_i &\sim Bern(\mu_2) && i \in [N] \end{aligned} \tag{13}$$

The second model, v3, is explained in selection fairness section. The last model, v4, alternate (14) and null (15) and is formulated as below

$$\begin{aligned} \mu &\sim U\,(1,1) \\ Y_i &\sim Bern(\mu) && i \in [N] \end{aligned} \tag{14}$$

$$\begin{aligned} \mu &\sim U\left(\frac{4}{5}\mu^*, 1\right) \\ Y_i &\sim Bern(\mu) && i \in [N] \end{aligned} \tag{15}$$

To explore these models, we designed a set of experiments. We fixed the total number of candidates with the same number male and female candidates. In the selection process, we incrementally add females to completely biased recommendation, e.g only male candidates were selected. The number of selected candidates is fixed in thus by adding female candidates we remove male candidates from the selected group. Figure 6 shows the behavior of these models in this controlled scenario. The horizontal dashed line indicates the line were the test has seen decisive amount of evidence for discrimination and the vertical dotted line represents the point were 4/5th rule fails for lower number of selected female candidates.



(a) Pool size of 40 with 10 candidates selected

(b) Pool size of 400 with 100 candidates selected

(c) Pool size of 1000 with 250 candidates selected

(d) Pool size of 2000 with 500 candidates selected

Figure 6: Behavior of

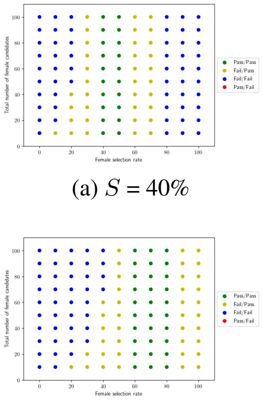
If we consider plot (b) in Figure 6, in the fair scenario, 50 candidates would have been selected from each gender. The 4/5th rule fails when less than 45 selected candidates are female. v3 fails when the number of selected females are less than 40, v1 fails at 23 and v4 fails only when number of selected females is 12. This indicates that v3 is an agreement with 4/5th rule but considers the lack of observations into account. Despite the fact that v1 and v4 follow the same logic, they respond to higher level of discrimination.

B In-Depth Evaluation of the Tests

B.1 Additional Experiments for the Selection Test

Here we include some additional results for Section 3. Figures7 & 8 extend the selection test experiments with additional values for the selection rate of males. We can see that the conclusions drawn previously still hold for these additional selection rates.

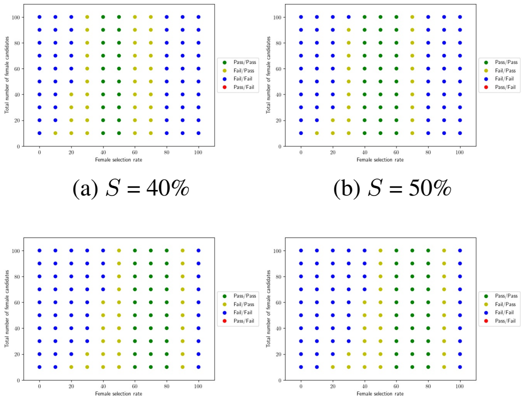
Figures9 & 10 extend for the ranking test experiments with additional values for the number of observations. Again, we see the conclusions discussed in Section 3 still hold.



(a) $S = 40\%$ (b) $S = 50\%$

(c) $S = 70\%$ (d) $S = 80\%$

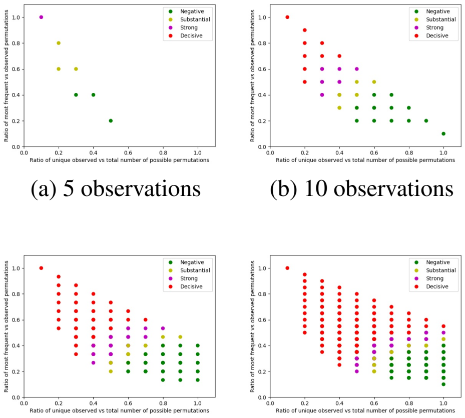
Figure 7: $T = 100$



(a) $S = 40\%$ (b) $S = 50\%$

(c) $S = 70\%$ (d) $S = 80\%$

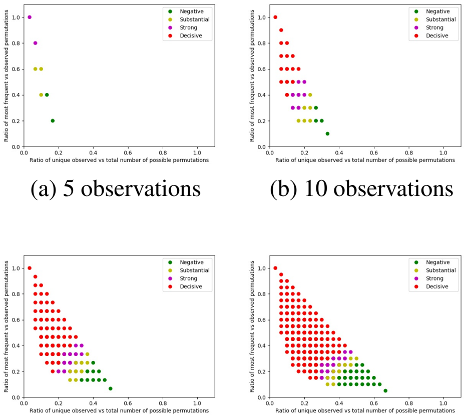
Figure 8: $T = 500$



(a) 5 observations (b) 10 observations

(c) 15 observations (d) 20 observations

Figure 9: 2 categories



(a) 5 observations (b) 10 observations

(c) 15 observations (d) 20 observations

Figure 10: 3 categories

B.2 Additional Experiments for Permutation Test

This section elaborates on the results reported in Section 3.2 and also includes additional experimental conditions.

How is this test behaving? Can the test detect unfairness from just N observed permutations? To quantify unfairness of the set of N permutations we define two indicators. If many of the observed permutations are actually identical, then the ratio of unique permutations to number of samples is low, and this might be an indication of unfairness. Similarly, if the most frequently observed permutation is a large fraction of the total observed permutations, then this is an indication of unfairness. In Figure 9, we look at the specific case of a top-K ranking that contains 2 categories, say male and female, with 3 males and 2 females, for which there are 10 possible permutations. We evaluate the test on N observed rankings that have different values of these indicators with the former of the two on the x axis and the latter on the y. Thus, points in the lower right correspond sets of permutations that are fair, and as we get further from this locus, unfairness increases and becomes most severe for the points in the upper left. We use colors yellow, purple, and red if the log-Bayes factor is greater than log(2), log(10), and log(100) respectively. Otherwise, we report green. These values are commonly used in analyzing Bayes factors. Each subplot reports on a different amount of observed data. Figure 10 reports on the same type of experiments but where we instead use 3 categories with occurrence 2, 2, and 1.

As we can see, the test behaves as expected, detecting unfairness only in the appropriate regions of this space; for example, towards upper left corner. As more specific example, if we look at the sole red dot at $x = 1.0$ in Figure 9d, we see that the test classifies it as decisively unfair, which makes sense because even though all the permutations are observed, the most frequently observed permutation constitutes more than half of them. More generally, we see that the test indeed accounts for the amount of evidence of permutation discrimination.

B.3 Understanding the failure modes

To understand the "failure mode" in our proposed fair ranking test, we designed two set of tests. For the simplicity, we present the results on the binary protected attribute like gender. In the first test we exclude female candidates from the top positions in the ranks. The fair ranking test is designed to react to condensation of observed ranking patterns in few possible permutations. Thus, if the observed patterns are uniformly distributed among the possible permutations, the ranking test will likely pass them. In this experiment we exclude female candidates from top 2, 3, 4 and 5 positions and distribute the number of observed patterns among the possible permutations evenly. The purpose of this test to see how many observations are required for the fair ranking test to eventually identify the discriminatory behavior.

Table 1: Minimum number of observations required for the test to find evidence of discrimination. In the first column the numbers provided in parentheses are all possible permutations in this *group*. In other columns, these numbers reflect all possible permutations within the restriction imposed on them.

group	Top 2 exclusion	Top 3 exclusion	Top 4 exclusion	Top 5 exclusion
9M, 1F (10)	83 (8)	48 (7)	32 (6)	23 (5)
8M, 2F (45)	107 (28)	56 (21)	34 (15)	22 (10)
7M, 3F (120)	128 (56)	61 (35)	33 (20)	19 (10)
6M, 4F (210)	116 (70)	50 (35)	24 (15)	12 (5)

In the second test, we examine the overall behavior of the test. As mentioned before, the fair ranking test does not consider position in the rank into consideration. Thus, this experiment is aiming to uncover underlying discrimination on the overall view that will not be visible to the test. We consider a scenario in which we are required to recommend 10 workers for 500 tasks. These workers are selected from different pool of 50 candidates and the on average 40% of the candidates in the pools are females. The fair ranking test is run on these rankings to ensure the fairness of them. This experiment is repeated 50 times and the results presented in the table 2 are the average of these runs.

The experiment includes four different cases. In the first case (column 2), for all the tasks, the score assigned to these candidates are random. Next (column 3), we consider a case were all the scores were skewed close to 0.1 in favor of male candidates. In the third case (column 4), for each pool we randomly skew the scores in favor of males but the average skew is close to 0. The fourth case is the same as the third one but the average skew is close to 0.1. For each case we calculated the probability of female being assigned to each position.

Table 2: 500 tasks, pool size of 50 for each task with 40% Females

Position	skew:0 mean $P(F\|p_i)$	skew:0.1 mean $P(F\|p_i)$	mean skew:0 mean $P(F\|p_i)$	mean skew:0.1 mean $P(F\|p_i)$
1	31%	4%	27%	16%
2	31%	13%	30%	18%
3	31%	17%	29%	20%
4	31%	22%	29%	24%
5	30%	27%	30%	26%
6	31%	29%	30%	27%
7	31%	28%	31%	28%
8	31%	29%	32%	27%
9	31%	23%	33%	29%
10	30%	26%	31%	28%

C ILP Feasibility Conditions

Below we discuss the conditions for the ILP 10 to be feasible in the case of gender as the protected group. The amount of deviation ϵ in the second constraint in 10 allows us to utilize the output of the tests as feedback and control the amount of post-processing changes. For example, $\epsilon_1 = 0$ allows us to enforce the 4/5 condition most aggressively, whenever it is feasible (i.e., whenever there are sufficient number of candidates in the pool that belong to the minority category). When this is not the case, the constraint 10 can not be satisfied even when either a) all the applicants in the minority group are moved into the top k recommendation or b) all the candidates in top k recommendations belong to the minority group, depending on the particular situation. Accordingly, we assign an appropriate value for ϵ_1 in order to address the infeasibility in the above situations. More specifically, consider the case when the number of minority group applicants in the pool is greater than k but is not sufficient to satisfy the 4/5 condition in aggregate. In this case, the best post-processing can do is to recommend the top k minority group candidates. To achieve this objective, we compute ϵ_{1alt} as the difference between 4/5th of the majority selection rate and the minority selection rate, where the top k recommendations belong to the minority group. Note that $\epsilon_{1alt} < 0$. Now consider the second possibility in which the number of minority group candidates in the current ranking is less than k. In this case, the post-processing step should bring in all the minority group candidates into the top k recommendations. Accordingly, we compute ϵ_{1alt} as the difference in selection rates in this situation. As a consequence, in order to execute aggressive post-processing, we have the maximum value of ϵ_1: ϵ_{1max} as either equal to ϵ_{1alt} or 0. Additionally, observe that if we do not intend to perform any post-processing at this stage, we simply set ϵ_1 to the observed difference in the constraint in 10, i.e., $\epsilon_1 =$ minority selection rate - $4/5$ (majority selection rate). This discussion can be extended to multiple categories in the protected group.

D Post-processor Evaluation

D.1 Datasets

German Credit The German credit dataset [3] contains financial information about 1000 individuals who are classified into groups of good and bad credit risk. The good credit group contains 699 individuals.

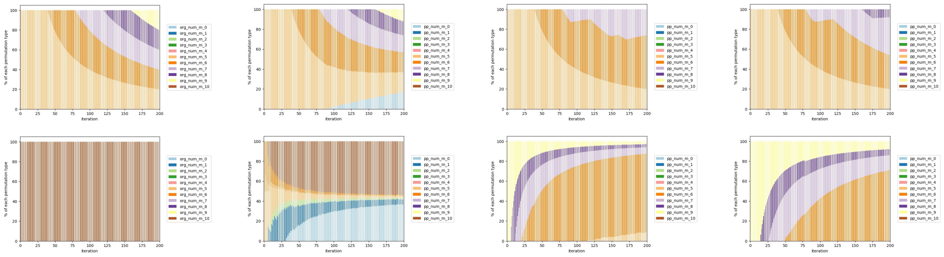
Census Income The census income dataset [3] extracted from the 1994 Census database contains demographic information about 48842 American adults. The prediction task is to determine whether a person earns over $50K a year. The dataset contains 16, 192 females (33%) and 32, 650 males (67%).

D.2 Decision on the degree of aggression in post-processing

In each experiment, we execute the post-processor with varying degree of aggression, which is decided by choosing appropriate values for the ϵ parameters. More precisely, in each iteration we compute $\epsilon = f(k_{ab})$ as the following. We compute the ϵ_{min} and ϵ_{max} as the parameter values corresponding to absent and maximum post-processing (as described in section C). We set the range of the k_{ab} values as $[\log 100, \theta]$ and define $\epsilon = f(k_{ab})$ as a linear mapping based on the above range of values. Increasing θ results in decreased intensity of the post-processor. Accordingly, we denote the most aggressive case as 'Aggr' ($\epsilon = \epsilon_{max}$), the moderate case as 'Mod' ($\theta = 1000$) and the lowest value as 'Low' ($\theta = 5000$).

D.3 Additional experiments

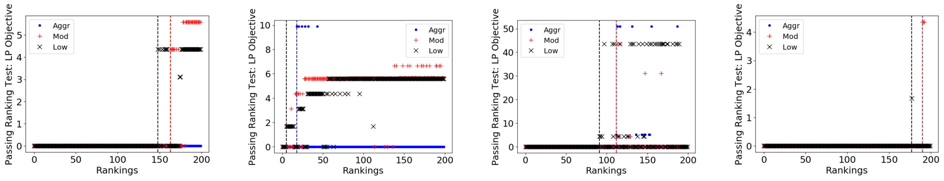
To further analyze the post-processor, we display the number of male applicants (recall that the males form the majority group in the simulations) in the top k recommendations before and after post-processing in Figure 12. The top row corresponds to the increasing bias case and the bottom row is for the extreme bias case. The first plot in each row shows the original top k recommendations in the 200 ranking samples. The next three plots show the recommendations as a result of the aggressive, moderate and low versions of the post-processor respectively. In each case, as the intensity of the post-processor increases, the number of minority group applicants (i.e., the females) in the top k recommendations increases and therefore, the number of males in top k decreases.



(a) (b) (c) (d)

Figure 11: Evaluation of the post-processor in terms of the number of male applicants in the top k recommendations for simulated rankings with extreme bias (top row) and linearly increasing bias (bottom row). The original top k recommendations are shown in (a). The recommendations as a result of the aggressive, moderate and low versions of the post-processor are shown in (b),(c) and (d) respectively.

In a second set of experiments in Figure 12, we plot the objective in the LP 11 vs. the ranking iterations. As expected, with increasing intensity of the post-processor, the solution to the LP is closer to the uniform distribution resulting in an increased magnitude of the objective (reported in iterations where the ranking test detects discrimination).



(a) (b) (c) (d)

Figure 12: Evaluation in terms of the LP objective in the post-processing step for the ranking test on (a) simulated rankings with extreme bias, (b) simulated rankings with linearly increasing bias, (c) rankings from German credit and (d) Census income data. A vertical line indicates the first iteration when the test(s) detect discrimination, for each version of the post-processor.

Finally, we elaborate on some details in Figure 5 (d). Here unprocessed ('Aggr unp', 'Mod unp' and 'Low unp') indicates the result of the ranking test on the set of observed rankings (which might have been post-processed) and the current ranking (which is not post-processed). Accordingly, the ranking test detects discrimination at different iterations depending on the type of the post-processor. In each case, post-processing (second phase) brings down the log Bayes factor for the ranking test to values smaller than the decisive threshold.