University of Padova

Department of Department of Mathematics

Master Thesis in Data Science

Fairness in Rankings via Randomized Algorithms

Supervisor
Tomaso Erseghe
University of Padova

Master Candidate
Andrii Kliachkin

Academic Year
2022-2023

Dedication.

Abstract

As machine-learning algorithms proliferate, there are growing concerns regarding their fairness. Can we stop AI, trained on real-world data, from reproducing and exacerbating real-world biases? There exists a growing body of work on fairness in AI, but often focussing rather narrowly on classification problems. Online advertising and job-candidate rankings, for example, utilize ranking algorithms instead of classification algorithms. The objectives of this work are to (a) describe the theoretical basis of fairness in ranking and the metrics used to evaluate it, (b) explore several existing post-processing algorithms for fairness in ranking, (c) introduce a novel randomized algorithm based on Mallows distribution that offers a tradeoff between fairness and accuracy, and (d) compare its performance in terms of obtained fairness and loss of ranking accuracy to such of existing deterministic algorithms.

Contents

Listing of figures

Listing of tables

Listing of acronyms

ML	Machine Learning
NDCG	Normalized Discounted Cumulative Gain
WAE	We are All Equal
WYSIWYG	What You See Is What You Get
CS	Construct Space
OS	Observed Space

1
Introduction

1.1 Motivation for fairness

Today, it is no secret that machine learning (ML) algorithms are not always impartial decision-makers. From the widely known case of the COMPAS recidivism risk prediction software in the USA being heavily biased against black offenders [2], to sexist biases in word2vec embeddings [3], there have been numerous precedents for AI models exhibiting prejudices.

Biases in ML algorithms can have different origins, but mainly, they can be divided into:

- Pre-existing: biases that are reproduced by models learning from biased data. The bias in the data can be representative of real-world prejudices (e.g. reflecting the wage disparity between men and women), or it can be produced by a flawed data collection process (e.g. picking a non-representative sample for a group of the population);

- Technical: biases that arise from technical constraints. These may be caused by the algorithm itself adding bias that was absent from the training data, or by the conditions in which the algorithm is used. An example of the latter may be the size of a web page that displays the results of a ranking algorithm (e.g. a web search); limited space allows the user to look only at the several highest-scoring results.

- Emergent: biases that arise from user interaction. Coming back to the example above, the users clicking more on the highest-scoring results will only increase their perceived relevance, and thus push the lower-scoring ones even deeper into obscurity.

Among those, arguably the most dangerous one is pre-existing bias, as it reinforces real-world injustices. An important point to state is that it can not be rectified through ”unawareness”: we can not eliminate bias by simply removing the sensitive information from the data. This is because, in most cases, it can still be inferred from the other attributes. For example, an individual's race may be highly correlated with their address, their media preferences, their political views and so on. A very recent example of this phenomenon comes from the GPT4 large language model. The model was given several texts in standard English and in African American Vernacular English (AAVE) – a dialect used predominantly by the black population of the US. When asked its thoughts on the authors of the texts, the model's responses to the AAVE speakers were overwhelmingly negative. At the same time, when asked openly about race, GPT4 did not exhibit any harmful prejudices [4].

In addition to the moral argument for fairness, there exists a legal one. The European Union ”Artificial Intelligence Act” adopted in March of 2023 is partly concerned with possible discrimination performed by machine learning models [5]. As such, it is likely that future ML products developed or deployed in the EU will have to comply with some kind of non-discrimination or fairness regulations.

The existing works on the topic of algorithmic fairness will be discussed shortly in subsection 1.2 and in more depth in chapter 3.

1.1.1 Blind fairness

In algorithmic fairness, we divide the data entries into groups defined by some attributes – race, gender, etc. However, what happens if we do not know which attributes to select?

In some cases, the definitions may seem obvious – for example, race, gender, age bracket and so on. However, often the bias can come from an unexpected angle: take, for example, the International Baccalaureate incident of 2020, where, due to COVID-19 restrictions, the final school exam was substituted with an automatic grade-prediction algorithm. After the grades have been issued, the algorithm was found to overwhelmingly discriminate based on which school each student attended [6].

Another issue with choosing the protected attributes is that of intersectionality: the discrimination may take place over several factors at once. For example, an algorithm may produce fair results for white adults, white children, and black adults, but not for black children. As such, a fairness intervention with an insufficient group definition (in this case, only race or only age) is not guaranteed to remove discrimination.

Finally, there is a privacy concern: individuals may not want their sensitive information, such as race, gender, address, etc. to be available to a third party – even if it is, supposedly, to ensure fairness.

This leads us to the problem of "fairness without demographics" – or ensuring fairness when the group definitions or individual group membership is not known – and a relaxed version of this task, when information about group membership is noisy. This topic has attracted some attention in recent years, but the focus of most of those works is on classification tasks [7], [8], [9]. In [10], the authors introduce an algorithm to detect groups with biased representation in ranking; I will discuss this work shortly.

1.2 Ranking: preliminaries and notation

In a fair learning-to-rank (LtR) setting, we are given a set C of "candidates" or "individuals" – items to be ranked. Each candidate is described by a set X of attributes (or "features") and a score s. Each candidate belongs to a group defined by a combination of sensitive features (sensitive attributes) $A \subseteq C$. These may be binary (e.g. "income greater than $35000"), or multinary (e.g. race, non-binary gender identity, city). The protected group is the one we suspect the algorithm is biased against – often a historically disadvantaged demographic or a minority (in the latter case, the bias may come from lack of data). We denote item at position i in the ranking π with π_i (sometimes also seen as $\pi(i)$) and use $\pi^{-1}(i)$ to denote the position of item i in π.

A common metric used to measure fairness in rankings is the Spearman footrule distance. Given two rankings π and σ, $|\pi| = |\sigma| = k$, it measures the total number of positions the elements of π were moved in σ:

$$F(\pi, \sigma) = \sum_{i \in (\pi \cap \sigma)} |\pi^{-1}(i) - \sigma^{-1}(i)|$$

Another metric is the Kendall-Tau (Kendall Tau, Kendall's Tau, KT) distance, which measures the number of discordant pairs between π and σ:

$$d_{KT}(\pi, \sigma) = \sum_{i,j \leq k, i<j} 1\{(\pi(i) - \pi(j))(\sigma(i) - \sigma(j)) < 0\}$$

As the goal of many rankings is to put the most "relevant" results on top, a measure of relevance, or utility, is required. The most widespread is the Discounted Cumulative Gain:

$$DCG(\pi) = \sum_{i=1}^{k} \frac{s(\pi(i))}{\log(1+i)},$$

It can also be normalized against the DCG of a "perfect" ranking – one where items are sorted according to their descending scores. The latter is called the Ideal DCG (IDCG), and the Normalized DCG is defined as:

$$NDCG(\pi) = \frac{DCG(\pi)}{IDCG(\pi)}$$

1.3 Existing solutions

1.3.1 ...for ranking

The task of learning-to-rank itself is not the focus of this work because, as I will show shortly, many FairML algorithms are focused on either modifying the data or modifying the output of the machine learning model. However, it is useful to give a short overview of this task and the models used for solving it.

The core components of an LtR problem are [11]:

- A set of queries $Q = q_{i i=1}^{|Q|}$;
- A set of documents $D_q = d_{q,i i=1}^{n_q}$ associated with each query q;
- A set of feature vectors $x_{q,j} = \langle \psi_1(q, d_{q,j}), ..., \psi_M(q, d_{q,j}))\rangle \in \mathbb{R}^M$ associated with each query-document pair; each element $\psi_i(...)$ is usually a score produced by a simpler ranking algorithm, e.g. cosine similarity between the tf-idf encondings of the query and the document, BM25 score of the pair, etc.;
- A set of relevance labels $l_{q,j}$ associated with each query-document pair.

This way, a dataset for such a problem consists of a set of feature vectors $x_{q,j}$ and corresponding labels $l_{q,j}$.

The goal of LtR models is to learn a function f that minimizes the loss function L over the training set. The choice of the loss function divides the models into three main categories:

- Pointwise: the loss function is calculated on each query-document pair individually;

- Pairwise: for each query, the model learns to predict pairwise preferences between pairs of documents, transforming the problem into a classification one; examples are Microsoft's RankNet [12], LambdaRank [13] and LambdaMART [14];

- Listwise: for each query, the loss function is calculated on the entire list: for an example, consider ListNet [15] or SoftRank [16].

The loss function usually, in some way, aims to optimize NDCG of the resulting ranking.

Comparing the models is beyond the scope of this thesis; in previous research, however, listwise [17] and pairwise [12] models have been shown to outperform pointwise ones.

1.3.2 ...FOR FAIRNESS IN RANKING

There is a growing body of work on fair ranking. The methods used for this task are classified into pre-processing, in-processing, and post-processing. Pre-processing algorithms focus on eliminating bias in the data by learning (or otherwise producing) a "fair" representation of each individual that eliminates information about their group membership, like ifair [18]. In-processing approaches introduce regularization to the algorithm's loss function that penalizes discrimination: DELTR [19] and Fair-PG-Rank [20] for listwise ranking models, and Beutel et al. [21] introduced a regularization term for pairwise ones. Finally, post-processing algorithms rerank the model's output to make it more fair [22], [23].

This thesis focuses on post-processing; I justify this choice in more detail in section 2. Some advantages include explainability of post-processing algorithms, ease of implementation into existing pipelines, and good performance in terms of fairness.

However, the overwhelming majority of FairML algorithms assume perfect information about group membership of candidates. A notable exception is the work of Moskovitch et al. [10], in which the authors attempt to achieve fairness without group information, improving on a previous work by Pastor et al. [24]. The algorithm they introduce aims to detect the widest collection of sensitive attribute values such that a group defined by those values is underrepresented in top-k; the lower bound of acceptable representation, as well as k, are defined by the user. These group definitions may then be used as input to a post-processing algorithm, or otherwise utilized in order to improve fairness.

Yet this work, in turn, assumes that the user knows the values of the sensitive attributes of each candidate – and, as discussed earlier, that information is not always available. In the next subsection, I will describe our approach to completely group-oblivious fairness.

1.4 Our contribution

1.4.1 A group-oblivious fairness algorithm

Our approach to fairness without demographics is inspired by work on Differential Privacy by Dwork et al. [25], where noise is admixed into database records to protect the users' privacy. Similarly, we "mix up" a ranking (which may be the output of an ML model) by sampling the Mallows' distribution [26]; by changing the variance of the latter, we can achieve different results in terms of fairness.

This does not require any information on group membership; in fact, the only necessary information is the order in which the candidates are ranked. Furthermore, this process is computationally fast: it just needs one sample from a random distribution.

The research this thesis is based on is due to be presented at the 1st International Workshop on Fairness in AI (`https://fairnessinai.github.io/`); the workshop paper preprint is available at `https://arxiv.org/abs/2403.19419v1`.

1.4.2 Contribution to AIF360

As part of my internship and thesis work, I implemented one of the state-of-the-art algorithms for fairness in rankings used in this thesis, DetConstSort [22], as well as 3 other algorithms described in the paper (DetGreedy, DetConservative, and DetRelaxed), and contributed them to the IBM-sponsored AI Fairness 360 Python package (`https://github.com/Trusted-AI/AIF360/pull/461`). I also implemented unit tests and a usage demo in Jupyter Notebook for the algorithms, several metrics for dealing with fairness in rankings, and added other supporting code. AI Fairness 360 is a widely-used open-source Python package that provides tools for detecting and eliminating algorithmic bias with over 1000 citations [27].

1.5 Thesis structure

This thesis is organized as follows:

- In Chapter 2, I will describe the theoretical basics of algorithmic fairness, as well as the metrics for quantifying it.
- In Chapter 3, I will lay out two state-of-the-art algorithms used for achieving fairness, and then introduce our Mallows algorithm.

- In Chapter 4, I will present an experimental comparison of the algorithms from Section 3 in a setting with perfect, noisy, and absent group information.

1.6 Code availability

The full code used for the Experiments section of this thesis can be accessed at `https://github.com/andrewklayk/fairness_with_mallows_distribution`.

2

Background on Fairness in Rankings

As discussed in the Introduction, the goal of algorithmic fairness is to make sure that the decision-making algorithm does not discriminate against protected subgroups of the population. But that is a rather nebulous definition: what do we mean by «discriminate»? How do we tell if an algorithm is «fair» or not?

In section 2.1, I will attempt to answer these questions with arguments based mainly the work of Friedler et al. [1] and Zehlike et al. [28], [29]. Then, in section 2.2, I will describe and contrast the two approaches that divide the field of FairML: Individual vs. Group Fairness. Finally, in section 2.3, I will lay out the mathematical formulation of some metrics used to quantify fairness in ranking.

2.1 Worldviews

As we shall see in the next section of this chapter, there are many quantitative measures of algorithmic fairness. However, [1] notes that they are oftentimes introduced «ad-hoc», without stating the assumptions underlying them. The authors introduce the idea of three spaces, such that any measure of fairness may be classified according to the assumptions it makes about interactions between them.

Those are the Construct space (CS), the Observed space (OS) and the Decision space (DS). A representation of them is shown on Fig. X. The Construct space represents the true properties of an individual in which the decision-maker is interested –e.g., their intelligence. However,

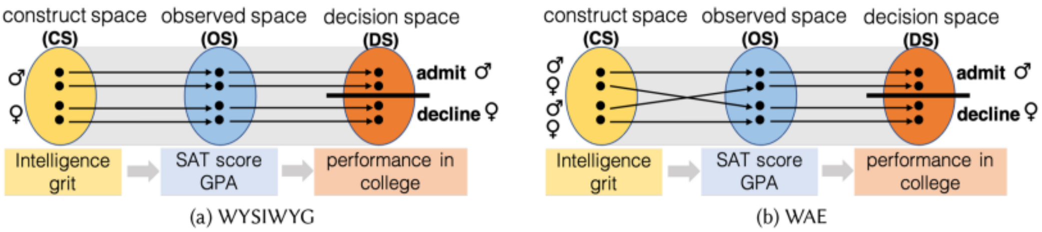


Figure 2.1: An illustration of worldviews from Frieder et al. [1]. WYSIWYG worldview assumes minimal disortion between the construct space (CS), that contains actual quantites of interest - e.g. intelligence and the observed space (OS), that contains observations - e.g. test scores. WAE worldview assumes that the mapping from the CS and OS treats different groups differently to a non-negligible degree - e.g. women's test scores being artificially lowered compared to men's.

those properties are unobservable directly – in the real world, we have to rely on proxies, like, for example, test scores; those lie in the Observed space. The Decision space is made up of the decisions made by the model – binary «yes/no» in case of classification, a number in case of regression, and the individual's position in case of ranking.

As the paper notes, any fairness metric implies assumptions about how those spaces interact. Those can be divided into two worldviews: «What You See Is What You Get» (WYSIWYG) and «We are All Equal» (WAE).

WYSIWYG denotes the belief that the differences between the OS and the CS are minimal, thus it is possible to accurately compare each candidate's true quantity of interest (e.g. a university applicant's ability) using just the observations (their exam scores).

WAE worldview, on the other hand, assumes that in the CS, all groups look essentially the same – all qualities relevant to the decision-making process are distributed nearly identically within each group. The reason WAE gives for observed differences is that the transformation from the CS to the OS is inaccurate and treats different groups in different ways. In this worldview, OS is an unreliable representation of the CS: for example, a physics exam taken in English may be representative of the physics knowledge of an English-speaking student, but not of a non-English-speaking one.

When designing a fairness-oriented algorithm, it is necessary to choose between the two worldviews. In some ways, they correspond to different concerns a model designer might have: roughly speaking, WYSIWYG means that we trust the data to provide an accurate representation of the CS, and are concerned with the model itself learning to discriminate on the unwanted attributes from data; while WAE means that we think the data is biased due to real-world issues and want the model to correct for that.

While the question of worldviews may seem purely theoretical, it is necessary for a deeper understanding of a much more practical choice in the field of Fair ML: Individual versus Group fairness.

2.2 Individual vs. Group Fairness

The first definition to arise in the field of Fair ML was **group fairness**. Group fairness is simple: it asks for the model output to be independent of group membership; given the model output R and group membership A, it requires that

$$P(R = 1|A = a) = P(R = 1|A = b)$$

It is also known as demographic parity, statistical parity, or disparate impact. Under the last name, and with a relaxed formulation, it is widely employed in legal settings - take, for example, the "four-fifths" rule in the USA that requires the job acceptance rate for each significant protected group to be at least 4/5 of that of the most frequently accepted group.

Individual fairness, on the other hand, aims for similar individuals to be treated similarly. It was introduced into the field of Fair ML in [30]; there, the similarity between individuals is measured with a user-defined metric. The authors recognize that designing a «fair» metric is a difficult problem, as there might be structural bias present in the data.

A widespread set of fairness criteria that aims for individual fairness is equalized odds and equality of opportunity, which require, respectively, equal error rates and equal false positive rates across groups. They are based on either full or relaxed statistical notion of separation: the output R is separated from the group membership A by the ground truth Y if $R \perp\!\!\!\perp A|Y$. It makes intuitive sense in a classification setting if we consider Y as a measure of merit: equally qualified individuals should be accepted or refused with equal rates, regardless of their group membership.

It is easy to see how the individual and group fairness notions map closely to the WYSIWYG vs. WAE worldviews, respectively. More accurately, as pointed out in [1], individual fairness requires the user to assume the WYSIWYG worldview to guarantee fairness, and vice versa (note that, unlike the paper, in this thesis I do not make a distinction between "fairness" and "non-discrimination", because this section is intended only to provide a basic theoretical background).

There are apparent tradeoffs to choosing between individual and group fairness, which have roots in the conflict between WYSIWYG and WAE. Using our running example, if we assume that the students' test scores accurately represent their intelligence (WYSIWYG), adhering to individual fairness criteria would make sure that students with similar scores would have the same probability of acceptance, regardless of race; on the other hand, if we know the tests to favor one group over the others (WAE), individual fairness would just reinforce that difference.

Similarly, the disadvantages of group fairness under WYSIWYG are apparent: by forcing the model to produce similar outputs regardless of group membership, it would unfairly «elevate» black students with lower scores. Group fairness constraints can even be used insidiously: a recruiter may hire candidates from the privileged and protected groups at the same rate, but purposefully select less qualified members of the latter; this would lead to worse performance from the protected group, which can be used to justify further discrimination [31]. In the real world, this is known as the «glass cliff» phenomenon.

2.3 Measuring Fairness in Rankings

Having established the theoretical basis behind different fairness criteria, we can finally move on to their mathematical formulation. As stated already, most work on fairness focuses on the problem of classification, and thus, a wide range of metrics has been introduced for this task, focusing on both group (positive classification rate) and individual (false positive rate ratio, false negative rate ratio, error rate ratio, and so on) fairness [32].

Much less work has been put into the problem of rankings, however. Biega et al. [33] choose the individual approach (WYSIWYG) and aim to ensure that the user attention received by each item is proportional to the item's relevance to a given query. As a proxy for attention, they use position-based discounts – a quantity that decreases as the item is placed further down in the ranking, e.g. $p(1-p)^{(j-1)}$ for a user-set parameter p and position j. The authors argue that their constraints are unlikely to be achieved in any single ranking, so they attempt to optimize cumulative attention received by items over multiple successive rankings (e.g. multiple repeated responses to the same internet search query, or even to different queries – in which case the items' relevance scores will change depending on the query):

$$\frac{\sum_{l=1}^{m} a_{i1}^{l}}{\sum_{l=1}^{m} r_{i1}^{l}} = \frac{\sum_{l=1}^{m} a_{i2}^{l}}{\sum_{l=1}^{m} r_{i2}^{l}}, \ \forall u_{i1}, u_{i2}.$$

, where u_{i1}, u_{i2} are a pair users, a_{i1}^{l}, a_{i2}^{l} are the normalized attention values received by the two users in ranking l, and r_{i1}^{l}, r_{i2}^{l} are the normalized relevance values of the two users in the ranking l.

In [34], the authors introduce constraints for both individual and group fairness based on the notion of exposure:

$$\text{Exposure}(d_i|\mathbf{P}) = \sum_{j=1}^{N} \mathbf{P}_{i,j} v_j$$

Where X_a is the item, $P^{\tau}_{a,i}$ is the probability of the model placing the candidate a at rank i in ranking τ, and $v(i)$ is the position discount.

The average exposure of a group G_k is defined as:

$$\text{Exposure}(G_k|\mathbf{P}) = \frac{1}{|G_k|} \sum_{d_i \in G_k} \text{Exposure}(d_i|\mathbf{P}),$$

In the individual case (WYSIWYG), the aim is to guarantee that the average exposure of each group is proportional to the average utility of its items; in the group case (WAE), the average exposure is required to be equal across all groups.

Finally, [35], [22] and [23] place firm constraints on the group structure of the rankings. These works aim to achieve equal representation of groups in the top k entries – also known as proportional, or proportionate, fairness.

For each position i and protected group g, [35] introduces a lower bound α_i such that the proportion of members of protected groups does not fall too far below a user-set percentage p. As the main contribution of the paper is an algorithm that constructs a sub-ranking of size k out of a larger set of scored candidates, the fairness of the result is evaluated using the percent of protected candidates in the sub-ranking. While this approach is intuitive, it does not account for the positions of the candidates: for example, all individuals belonging from the protected group may be situated at the bottom.

[22] introduced the InfeasibleIndex metric, defined as the number of positions at which the lower representation bound is violated:

$$\text{InfeasibleIndex}(\pi) = \sum_{k=1}^{|\pi|} 1(\exists G_i \in G,\ s.t.\ \text{count}_k(G_i, \pi) < \lfloor \alpha_i \cdot k \rfloor). \tag{2.1}$$

where τ_r is the ranking, G is the set of groups, α_i is the lower constraint on the proportion of items of group G_i in the ranking set by the user.

[23] improved on that by also considering the upper bound in the "Percentage of P-Fair Positions" – or the percentage of positions in the ranking at which none of the requirements

are violated. The authors claim that introducing an upper representation constraint in addition to the lower one provides for more robust fairness guarantees.

In this work, I use the latter metric, as well as a modification of InfeasibleIndex that also counts the upper bound violations:

$$\text{LowerViol}(\pi) = \sum_{k=1}^{|\pi|} 1(\exists G_i \in G,\ s.t.\ \text{count}_k(G_i, \pi) < \lfloor \alpha_i \cdot k \rfloor$$

$$\text{UpperViol}(\pi) = \sum_{k=1}^{|\pi|} 1(\exists G_i \in G,\ s.t.\ \text{count}_k(G_i, \pi) > \lceil \beta_i \cdot k \rceil$$

where $\text{count}_k(G_i, \pi)$ is the number of elements of group G_i in the top k positions of ranking π; α_i, β_i are the lower and upper constraints on the proportion of items of that group in the ranking. Then,

$$\text{TwoSidedInfInd}(\pi) = \text{LowerViol}(\pi) + \text{UpperViol}(\pi)$$

Depending on the choice of the proportions for each group, these metrics can describe both individual and group fairness, as noted in [22] and [23]. In this thesis, the focus will be placed on group fairness.

3 Algorithms

There are several ways to classify algorithms used for mitigation of bias in rankings. Most commonly, they are categorized into pre-, in-, and post-processing. Pre-processing methods seek to mitigate bias in the data before it is used for training a ranking model. In-processing methods modify the ranking model (usually, by altering the loss function) so that the outcome is bias-free. Post-processing methods reorder the output ranking so that it satisfies fairness constraints.

In this thesis, I focus on post-processing algorithms. Such methods offer several advantages, as noted in [29]:

- Most of them come with firm theoretical guarantees on the fairness of the output – which is not the case with in-processing algorithms that are intended to produce a tradeoff between fairness and accuracy;
- Their effect is much more clear than that of other methods: while pre- and in-processing algorithms intervene in some way on the training data or the loss function, post-processing methods «simply» reorder the output, usually according to easily understandable rules. This makes them more easily analyzable and explainable – the latter being a very important property of decision-making algorithms, as noted in chapter WHICH of [31].

Those advantages, however, come with greater loss in accuracy than that of pre- and in-processing methods. On the other hand, if bias is present in the scoring function, the decrease in accuracy may not be at all representative of the real-life impact [29].

In addition to those points, I suggest another quite apparent benefit: most post-processing algorithms do not require access to either the training data or the ranking model; just the scores produced by the latter. As such, they are easy to introduce into the pipeline of a ranking system – as it was done, for example, at LinkedIn [22].

The same paper also classifies algorithms according to their underlying worldview – either WYSIWYG or WAE, described in Chapter 2, and the type of bias the algorithm can mitigate (pre-existing, technical, or emergent).

3.1 State-of-the-art post-processing algorithms

3.1.1 DetConstSort

Geyik et al. in [22] introduce four post-processing ranking algorithms that intervene on the ranked output: DetGreedy, DetCons, DetRelaxed and DetConstSort. Among those, the only algorithm the authors prove the theoretical feasibility for is DetConstSort. Every algorithm introduced in the paper intervenes on the output ranking by introducing minimum and maximum representation constraints for each group at each k:

$$\forall k \leq |\tau_r| \, \& \, \forall a_i \in A, \; count_k(a_i) \leq \lceil p_{a_i} \cdot k \rceil \;, \; and, \tag{3.1}$$

$$\forall k \leq |\tau_r| \, \& \, \forall a_i \in A, \; count_k(a_i) \geq \lfloor p_{a_i} \cdot k \rfloor \;, \tag{3.2}$$

where p_{a_i} represents the proportion of the group defined by the protected attribute value a_i; this value is set by the user.

The DetConstSort algorithm is presented in Algorithm 3.1. In a nutshell, the algorithm does the following:

1. Increase the value of the counter k until the minimum representation requirement at position k is increased for at least one group; pick one candidate from each group with the best score and order them according to their descending score;

2. For each such candidate:

   (a) insert the candidate into the last empty position in the recommendation list;

   (b) swap the candidate towards the top position until either:

      - The score of the candidate above is greater than the score of the inserted candidate, or

- The minimum representation requirement for the candidate above would be violated due to the swap.

Algorithm 3.1 Feasible Mitigation Algorithm Based on Interval Constrained Sorting (DetConstSort)

```
foreach $a_i \in a$, counts[$a_i$] $:= 0$
foreach $a_i \in a$, minCounts[$a_i$] $:= 0$
rankedAttList := [];   rankedScoreList := [];   maxIndices := []
lastEmpty := 0;   k := 0
while lastEmpty $\leq k_{max}$
    k++
    foreach $a_i \in a$, tempMinCounts[$a_i$] $:= \lfloor k \cdot p_{a_i} \rfloor$
    changedMins := $\{a_i$ : minCounts[$a_i$] $<$ tempMinCounts[$a_i$]$\}$
    if changedMins $\neq \emptyset$
        ordChangedMins := sort changedMins by $s_{a_i,\, counts[a_i]}$ descending
        for $a_i \in$ ordChangedMins (chosen in the sorted order)
            rankedAttList[lastEmpty] := $a_i$
            rankedScoreList[lastEmpty] := $s_{a_i,\, counts[a_i]}$
            maxIndices[lastEmpty] := k
            start := lastEmpty
            while start > 0 and maxIndices[start - 1] $\geq$ start and rankedScoreList[start-1] < rankedScoreList[start]
                swap(maxIndices[start - 1], maxIndices[start])
                swap(rankedAttList[start - 1], rankedAttList[start])
                swap(rankedScoreList[start - 1], rankedScoreList[start])
                start−−
            end while
            counts[$a_i$]++
            lastEmpty++
        end for
        minCounts := tempMinCounts
    end if
end while
return [rankedAttList, rankedScoreList]
```

Any algorithm that allows the user to set the desired proportion of groups seemingly permits a continuous shift between WAE and WYSIWYG worldviews. However, [29] notes that the DetGreedy algorithm from [22] is incompatible with WAE, as it always picks the highest-scoring available candidate at each step, and in this way introduces a measure of utility into the fairness constraints. Evidently, DetConstSort does this as well, choosing higher-scoring candidates when ties in the representation constraints are present.

Furthermore, in [23], authors show that DetConstSort does not always produce the closest fair ranking to the initial one; however, I have found that the paper misrepresents the performance of the algorithm due to an error in the code used for running the experiments – DO I ADD THIS (maybe in appendix? seems important enough).

3.1.2 Approximate Multi-Valued IPF

The paper [23] introduces three algorithms to solve the problem of constructing a fair ranking from a possible unfair one of the same size – what the authors call the Individual p-Fair Ranking problem (IPF). The first algorithm, GrBinaryIPF, is only viable for the case of 2 groups and follows the same principle as the DetGreedy algorithm from [22]. Like with DetGreedy from the last paper, the authors provide a proof that GrBinaryIPF produces a fair ranking in the case of ≤2 groups; moreover, the latter is proven to produce the optimal solution to the IPF problem in the case of ≤2 groups.

Here I consider the algorithm the authors call Approximate Multi-Valued IPF. As this algorithm is based on minimum weight perfect matching in graphs, let me give a brief overview of the necessary concepts:

A graph $G(V_t, E)$ consists of a set of vertices V_t and a set of pairs of vertices called "edges" E. An edge $e_{vy} = (v, y), v, y \in V_t$ is called incident to the vertices v and y; v and y are called its endpoints. A graph is called bipartite if V_t can be divided into two sets V and Y such that no edge in E has both endpoints in either V or Y. Such a graph is also denoted as $G(V, Y, E)$. A matching $M \subseteq E$ is a collection of edges s.t. every vertex in V_t is incident to at most one edge in M. A perfect matching is a matching where every vertex in V_t is incident to exactly one edge in M.

Approximate Multi-Valued IPF exploits the fact that a perfect matching in a bipartite graph produces a one-to-one correspondence between the vertices of the two partition sets. First, the algorithm calculates the lower and upper bound of positions for each candidate of the ranking based on their group membership. It then represents the original score-based ranking π_0 and the output fair ranking π (to be found) as two parts V and Y of a bipartite graph $G(V, Y, E)$: the nodes in V represents the items and the nodes in Y represent their potential position. Each edge e_{vy} represents an assignment of a candidate $v \in V$ to a position $y \in Y$; as y is supposed to be a fair ranking, an edge e_{vy} exists only if y lies between the lower and upper position bound for that candidate. The edges are then given weights $w(e_{vy})$ equal to the Spearman's footrule distance between y and the position of v in π.

The algorithm then finds a minimum weight perfect matching between V and Y - a matching M that minimizes the sum of the weights of the edges in it. This is a well-researched problem that can be solved in polynomial time [36].

The pseudocode of Approximate Multi-Valued IPF, as it is given in the original paper, is given in Algorithm 3.2.

Algorithm 3.2 ApproxMultiVauedIPF(G)

```
for $e_{vy} \in E$
    $w(e_{vy}) \leftarrow |\pi_0(v) - y|$
end for
$M \leftarrow find\_minimum\_weight\_perfect\_matching(G)$
for $e_{vy} \in M$
    $\pi(y) = \pi_0(v)$
end for
return $\pi$
```

This algorithm was not analyzed in [29], but it is evident that it also conforms to the WYSIWYG worldview for the same reason as DetConstSort: aiming to minimize the distance between the initial score-based ranking and a fair one, it will place higher-scoring candidates higher in the ranking. The authors prove that Approximate Multi-Valued IPF admits a 2-approximation factor for the IPF problem in terms of the Kendall-Tau distance to the initial ranking.

3.2 Baseline: Integer Linear Programming

In order to provide an additional baseline, we introduced an integer linear problem formulation of the p-fair ranking problem. Given a set of lower and upper representational bounds α_g, β_g for each group $g \in G$, the optimal p-fair top-k ranking can be computed from an initial ranking π_0 as follows:

$$
\begin{aligned}
\max \sum_{i\in[k]} \sum_{j\in[k]} & s(i)c(j)x_{ij} && \\
\text{s.t.} \sum_{i\in[k]} & x_{ij} = 1 && \forall j \in [k] \\
\sum_{j\in[k]} & x_{ij} \leq 1 && \forall i \in [k] \\
\lfloor \alpha_g \ell \rfloor \leq & \sum_{i:\pi_0(i)\in g} \sum_{j=1}^{\ell} x_{ij} \leq \lceil \beta_g \ell \rceil && \forall \ell \in [k], \forall g \in G \\
& x_{ij} \in \{0,1\} &&
\end{aligned}
$$

3.3 Mallows algorithm for group-oblivious fairness

In this section I will introduce the algorithm developed and analyzed by us for fairness without group information in top-k rankings. This algorithm is simple: starting from a central ranking, it "shuffles" the candidates randomly according to the Mallows distribution. This, however, produces impressive results in terms of fairness, and addresses more than one fairness concern, as will be discussed later. The pseudocode is given in Algorithm 3.3

The Mallows distribution was introduced in [26] for the problem of modelling rankings. Given a center ranking π_0 and a decay parameter θ, the model is:

$$\mathbb{P}_{\pi\sim\mathcal{M}(\pi_0,\theta)}[\pi] = \frac{e^{-\theta d(\pi,\pi_0)}}{Z_k(\theta)}$$

The probability of obtaining a given ranking π decreases with increasing KT-distance between π and π_0.

This distribution belongs to the location-scale family; its expectation is the center ranking π_0 and its variance is θ.

Algorithm 3.3 Mallows Algorithm

```
$\pi_0 \leftarrow choose\_central\_ranking()$
$\pi \leftarrow \mathcal{M}(\pi_0, \theta)$
return $\pi$
```

The choice of π_0 and θ is a crucial point of this method. With $\theta = 0$, equal probability is given to all possible permutations; higher values means less changes in the ranking. Varying this parameter allows the user to achieve a tradeoff between fairness and utility.

Similariy, choosing the way in which the central ranking is constructed will also impact the results. The most intuitive approach is to use the score-based ranking of top-k highest-scoring candidates; this allows to achieve the highest utility and puts the method firmly in the WYSIWYG worldview. Another way is to produce a ranking in which each group is represented to some desired degree; this is the approach I use in Chapter 4 4, as it produces a certain tradeoff between the individual and group fairness approach. Finally, one may sample a permutation on the entire dataset and take the top-k candidates: with $\theta = 0$, this would fully conform to the WAE worldview.

Along with not requiring information about group membership, another advantage of this approach over many other reranking methods is the randomization, which helps combat position bias. Studies show that top-ranked candidates tend to receive disproportionately more attention than even slighty lower ranked ones [37] [38]. In deterministic algorithms, items with lower scores will **always** be presented below higher-scoring ones, bar fairness constraints. This, in turn, may lead to emergent bias - less clicks on a lower-scoring search result will further lower its perceived relevance. Alleviating these effects was the primary concern in the work of Biega et. al. [33]; here it is a useful side effect.

4 Experiments

In this chapter, I will describe the experiments we conducted to evaluate our randomized approach to group-oblivious fairness in rankings and compare it to the aforementioned state-of-the-art algorithms, and lay out their results.

4.1 Implementation

All algorithms were implemented in Python. The implementation of ApproximateMultiValuedIPF was the one provided with the original paper. The implementation of DetConstSort was based on the AI Fairness 360 toolkit with minor adjustments.

Sampling from the Mallows distribution was done using the code provided with [39] at `https://github.com/ekhiru/top-k-mallows/`. All code is available at `https://github.com/andrewklayk/fairness_with_mallows_distribution`.

4.2 Experimental results

4.2.1 Mallows model and Infeasible Index

The first experiment aims to evaluate the impact of Mallows randomization on the Infeasible Index of a ranking. The experimental setup and results follow.

Experiment 1: Setup

I analyze a scenario of ten candidates, who belong to two equal-sized groups and create multiple rankings, denoted as σ_{II}, by adjusting the placement of candidates from each group to produce diverse values of the Infeasible Index (II). I then sample rankings from a Mallows distribution using the σ_{II} ranking as the central permutation and varying values of θ, observing the resulting Infeasible Index in the samples.

Experiment 1: Results

Fig. 4.1 shows that as the dispersion parameter increases, the Infeasible Index of samples drawn from Mallows model converges to the Infeasible Index of the central permutation. When the central permutation's Infeasible Index is small, and as $\theta \to 0$, the Infeasible Index of the samples drawn from Mallows' model is higher, but without a significant difference. However, when the central permutation's Infeasible Index is large, there is a significant drop in the Infeasible Index of the samples drawn from the Mallows' model as $\theta \to 0$.

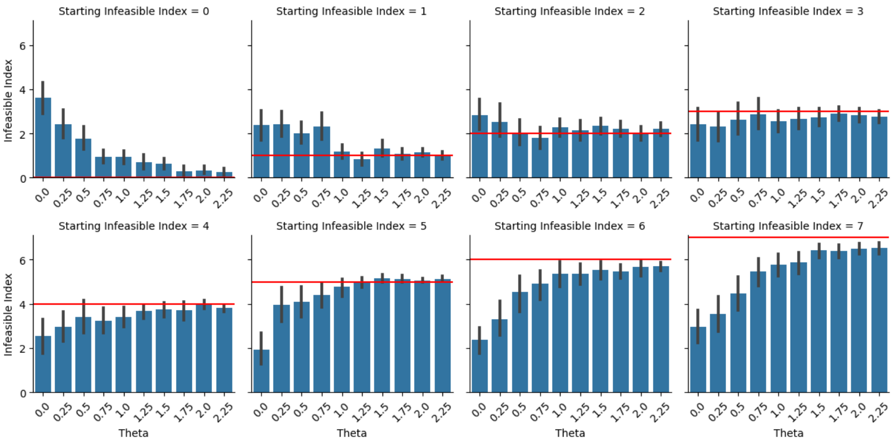


Figure 4.1: Mallows distribution and Infeasible Index (Experiment 1, Subsection 4.2.1). Each subplot corresponds to a different value of the central ranking's Infeasible Index, which is shown as a red line. The bar plots depict the mean value of the Infeasible Index of the samples from the Mallows distribution centered on the initial ranking with two groups. Confidence intervals were obtained via bootstrapping ($n = 1000$).

4.2.2 Mallows model and NDCG

The aim of the second experiment is to evaluate the impact of Mallow's randomization on both fairness and ranking utility in a simple ranking setup.

Experiment 2: Setup

I consider two equal-sized groups of five individuals each, where the candidates in the first group are assigned scores $S_1 \sim \mathcal{U}(0,1)$, and the candidates in the second group are assigned scores $S_2 \sim \mathcal{U}(0+\delta, 1+\delta)$, where $\delta = \{0.0, 0.1, \ldots, 0.9, 1.0\}$. I then sort the candidates according to their descending scores and sample the Mallows distribution centered on the sorted rankings with different values of θ, and evaluate the Infeasible Index and NDCG of the samples.

Experiment 2: Results

Figs. 4.3 and 4.4 depict the experimental results in terms of evaluating how the Infeasible Index and the NDCG change as we start from the corresponding Infeasible Index values as depicted in a rankings Fig. 4.2. As for the Infeasible Index, we notice similar behavior as in the Fig. 4.2.1. We can see that as the dispersion parameter increases the NDCG converges to that of the central ranking, which is 1. This illustrates that there is a trade-off between the NDCG and the Infeasible Index when sampling from Mallow's distribution with different values of the dispersion parameter.

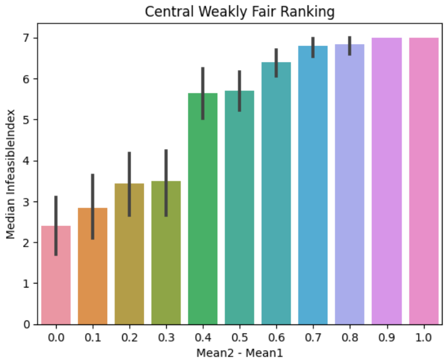


Figure 4.2: The Infeasible Index of the Central Ranking, as constructed by sampling from score distributions for each of the two groups (Experiment 2, Subsection 4.2.2). Specifically, the x-axis depicts the difference in means between the score distributions of the two groups. Confidence intervals were obtained via bootstrapping ($n = 1000$).

4.2.3 A real-world application: German Credit Dataset.

In the third experiment, we utilized a real-world dataset to evaluate how well the Mallows randomization method performs in a practical scenario, where we have partial knowledge regarding some of the protected attributes and aim to evaluate the result in terms of an unknown

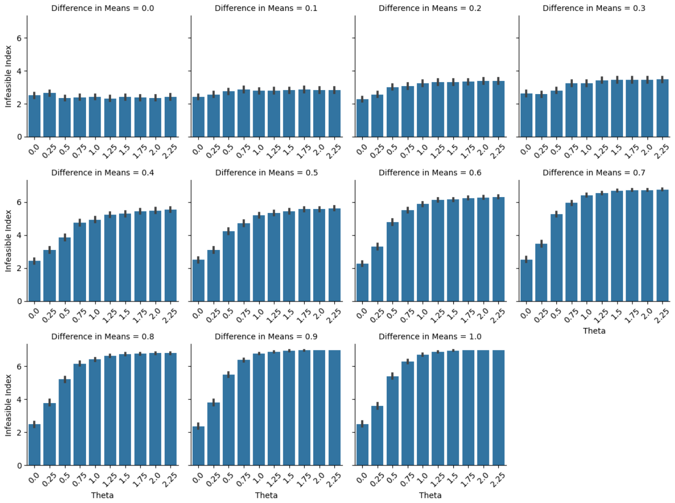


Figure 4.3: Mallows distribution and Infeasible Index in a simple score-based ranking setup (Experiment 2, Subsection 4.2.2). Each subplot corresponds to a difference in means between the score distributions of the two groups. We sample five individuals for each group, where the candidates in the first group are assigned scores $S_1 \sim \mathcal{U}(0,1)$, and in the second group - $S_2 \sim \mathcal{U}(0+\delta, 1+\delta)$, where δ is the difference in means. The subplots depict the mean value of the Infeasible Index of the samples from the Mallows distribution centered on the initial ranking. Confidence intervals were obtained via bootstrapping ($n = 1000$).

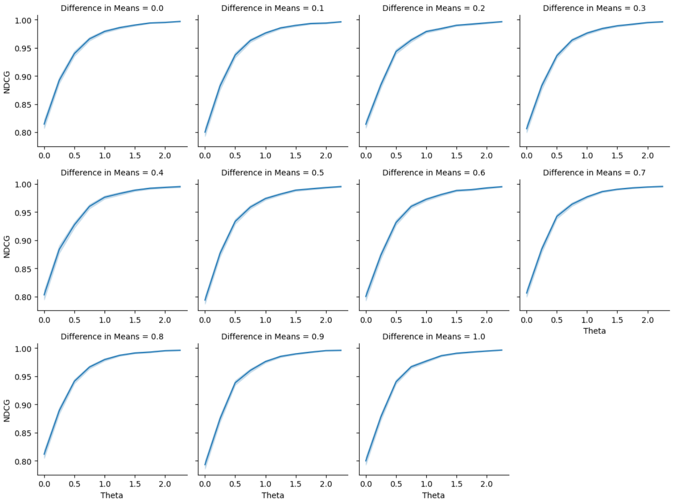


Figure 4.4: Mallows distribution and NDCG in a simple score-based ranking setup (Experiment 2, Subsection 4.2.2). Each subplot corresponds to a difference in means between the score distributions of the two groups. We sample five individuals for each group, where the candidates in the first group are assigned scores $S_1 \sim \mathcal{U}(0,1)$, and in the second group - $S_2 \sim \mathcal{U}(0+\delta, 1+\delta)$, where δ is the difference in means. The subplots depict the mean value of NDCG of the samples from the Mallows distribution centered on the initial ranking. Confidence intervals were obtained via bootstrapping ($n = 1000$).

protected attribute. We conduct a thorough experimental analysis with several state-of-the-art postprocessing algorithms that are designed to ensure fairness in rankings regarding specific attributes. To emulate real-world conditions, we introduce noise into their fairness constraints to simulate imperfect knowledge about group membership.

Dataset Description

We utilize the German Credit dataset [40], following the work of [41, 23]. For the ranking of the candidates, we use the Credit Amount attribute. We aggregate the binary attributes Sex and Age into the $Sex - Age$ protected attribute with four values and consider the information of this attribute as known, with little or no noise. We evaluate the fairness of the algorithms in terms of a third attribute, named $Housing$, with three values. We regarded the $Housing$ attribute as unknown; therefore, it could not be used as information for any algorithm. The distribution of the groups defined by these attributes is shown in Table 4.1.

Age-Sex	Housing			Total
	free	own	rent	
< 35 - female	2	131	80	213
< 35 - male	23	261	51	335
≥ 35 - female	17	65	15	97
≥ 35 - male	66	256	33	355
Total	108	713	179	1000

Table 4.1: Distribution of groups defined by Age, Sex, and Housing in the German Credit dataset.

Experiment 3: Setup

We executed the state-of-the-art ApproxMultiValuedIPF [23], DetConstSort [22] as well as the ILP algorithm, using as input a weakly-p-fair ranking with respect to the combined $Sex - Age$ protected attribute. The algorithms were run in their vanilla version and with some noisy representation constraints on the combined $Sex - Age$ protected attribute. Specifically, we introduced noise into the calculations of the constraints by each of the aforementioned algorithms in the following ways:

- ApproxMultiValuedIPF: we added an independent sample from $N(0, \sigma)$ to each of the weights at the weight calculation step (Algorithm 2, line 2 of [23])
- DetConstSort: we added an independent sample from $N(0, \sigma)$ to each of tempMinCounts (Algorithm 3, line 7 of [22])

- ILP: given $X_{ij}, Y_{ij} \sim |N(0,\sigma)|$, we modified the calculation of constraints for each group G_p such that:

$$\lfloor \beta_p \ell \rfloor - X_{ij} \leq \sum_{i \in G_p} \sum_{j=1}^{\ell} x_{ij} \leq \lceil \alpha_p \ell \rceil + Y_{ij} \qquad \forall \ell \in [k]$$

  This was done to lessen the probability of making the problem infeasible, while still retaining noise.

We repeated this step 15 times to reliably measure the effect of the noise.

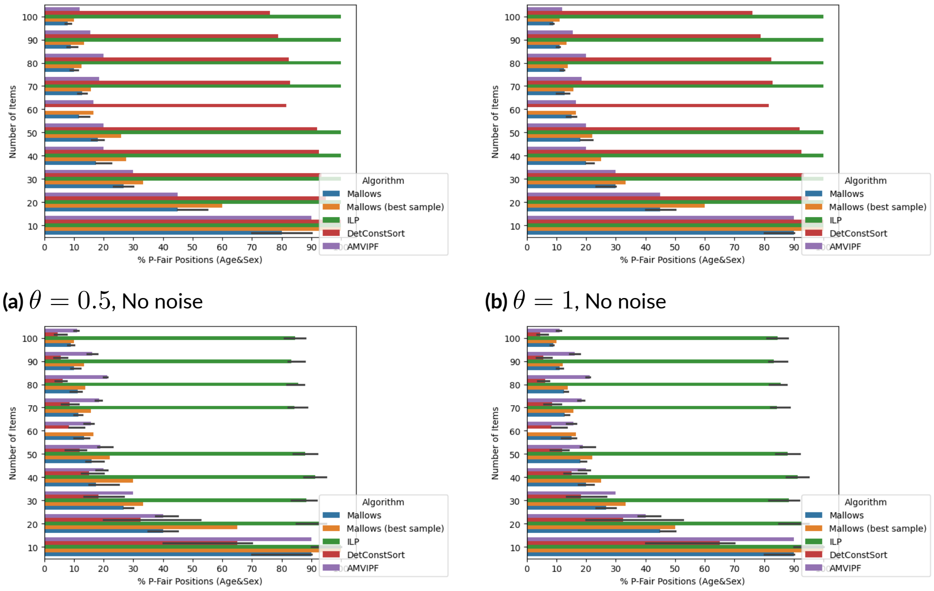
We also run the Mallows algorithm on the same weakly-p-fair ranking, using the weakly-p-fair ranking as a central ranking and dispersion parameters of 0.5 and 1, taking 1 or the best of 15 samples.

We evaluate the fairness of the output rankings using the Infeasible Index with respect to the $Housing$ attribute and the utility of the output rankings using NDCG. The experiment is repeated for rankings of size 10, 20, 30, 40, 50, 60, 70, 80, 90, 100.

Experiment 3: Results

The experimental results are presented in Figs. 4.6 and 4.7. Fig. 4.6 shows the median percentage of positions satisfying P-fairness with respect to the $Housing$ protected attribute. Since DetConstSort, ApproxMultiValuedIPF and the ILP use the $Age - Sex$ protected attribute to create the fair ranking, the resulting ranking fairness has to do with another attribute's distribution, therefore we cannot have any guarantees. For comparison only, we include Fig. 4.5, so that it is clear how the same ranking can have different fainress scores according to different attribute. We argue that the addition of noise can improve the results regarding different protected attributes, leading to a more balanced output, regardless of the target protected attribute. This approach seems like a compromise among all the different protected attributes that may be present and about which we have no knowledge.

As shown in Fig. 4.5, under noisy conditions, the Mallows algorithm performs well compared to state-of-the-art algorithms (DetConstSort and ApproxMultiValuedIPF). Fig. 4.7 shows the mean NDCG (as a solid line) and a 95% confidence interval (shaded region). We can notice that as the number of items increase, the NDCG score increases for Mallows, and the performance of the best sample from Mallows approaches the NDCG curve for the ILP.



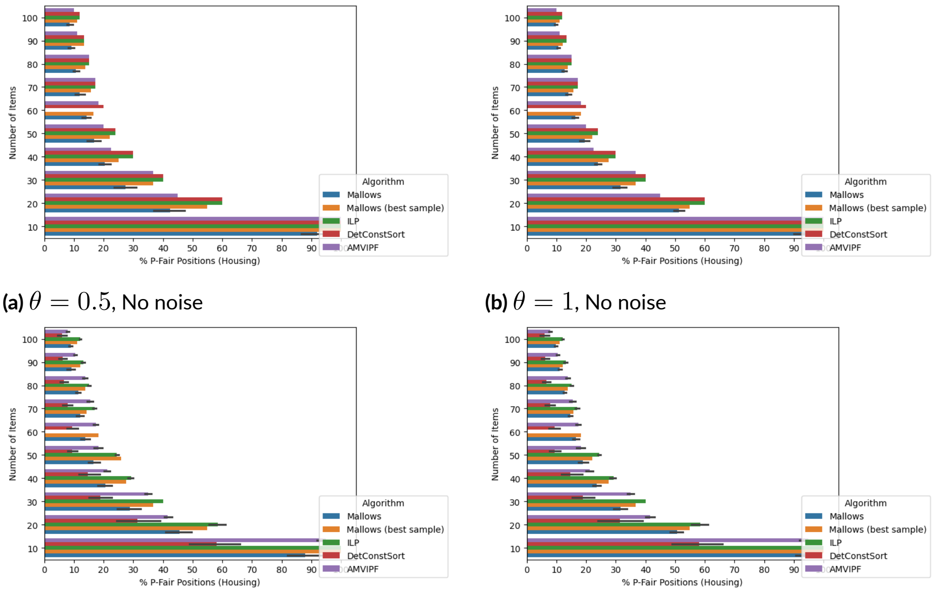
(a) $\theta = 0.5$, No noise

(b) $\theta = 1$, No noise

(c) $\theta = 0.5, \sigma = 1$

(d) $\theta = 1, \sigma = 1$

Figure 4.5: Rankings constructed with noisy representation constraints on the combined $Age - Sex$ protected attribute from an initial weakly-p-fair ranking with respect to the combined $Age - Sex$ protected attribute. **The plots show the median percentage of positions satisfying P-fairness w.r.t. the $Age - Sex$ protected attribute**. Confidence intervals were obtained via bootstrapping ($n = 1000$). In Subfigure (a) the θ parameter of the Mallows distribution is set to 0.5, and no noise is added to the constraints. In Subfigure (b) $\theta = 1$ and no noise is added to the constraints. In Subfigure (c) $\theta = 0.5$ and Gaussian noise $\xi \sim \mathcal{N}(0, 1)$ is added to the constraints. In Subfigure (d) $\theta = 1$ and Gaussian noise $\xi \sim \mathcal{N}(0, 1)$ is added to the constraints.



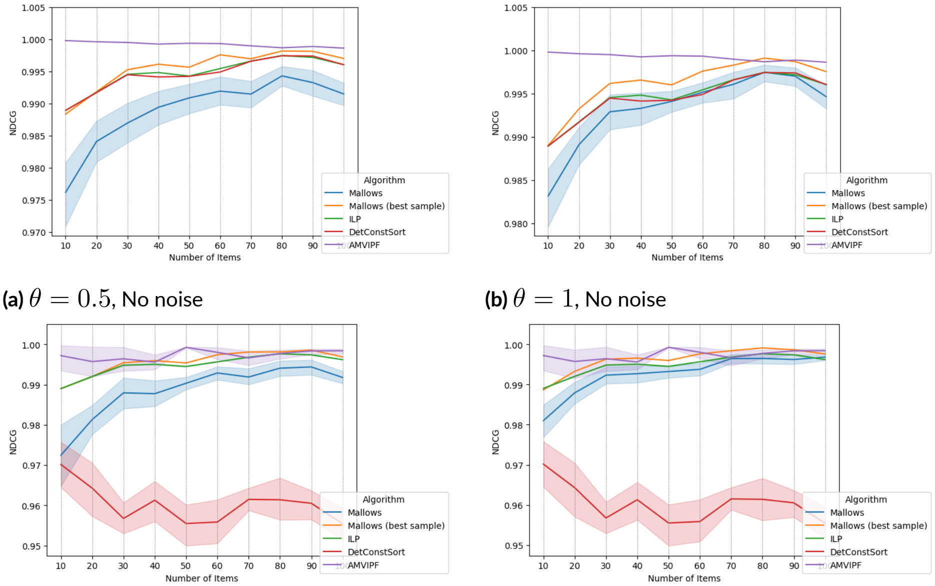
(a) $\theta = 0.5$, No noise

(b) $\theta = 1$, No noise

(c) $\theta = 0.5, \sigma = 1$

(d) $\theta = 1, \sigma = 1$

Figure 4.6: Rankings constructed with noisy representation constraints on the combined $Age - Sex$ protected attribute from an initial weakly-p-fair ranking with respect to the combined $Age - Sex$ protected attribute. **The plots show the median percentage of positions satisfying P-fairness w.r.t. the $Housing$ protected attribute**. Confidence intervals were obtained via bootstrapping ($n = 1000$). In Subfigure (a) the θ parameter of the Mallows distribution is set to 0.5, and no noise is added to the constraints. In Subfigure (b) $\theta = 1$ and no noise is added to the constraints. In Subfigure (c) $\theta = 0.5$ and Gaussian noise $\xi \sim \mathcal{N}(0, 1)$ is added to the constraints. In Subfigure (d) $\theta = 1$ and Gaussian noise $\xi \sim \mathcal{N}(0, 1)$ is added to the constraints.



(a) $\theta = 0.5$, No noise

(b) $\theta = 1$, No noise

(c) $\theta = 0.5, \sigma = 1$

(d) $\theta = 1, \sigma = 1$

Figure 4.7: Mean NDCG (solid line) and a 95% confidence interval (shaded region) of the output rankings. Confidence intervals were obtained via bootstrapping ($n = 1000$). In Subfigure (a) the θ parameter of the Mallows distribution is set to 0.5, and no noise is added to the constraints. In Subfigure (b) $\theta = 1$ and no noise is added to the constraints. In Subfigure (c) $\theta = 0.5$ and Gaussian noise $\xi \sim \mathcal{N}(0, 1)$ is added to the constraints. In Subfigure (d) $\theta = 1$ and Gaussian noise $\xi \sim \mathcal{N}(0, 1)$ is added to the constraints.

5 Conclusion

In this work, I have described a novel randomized post-processing algorithm for fairness in rankings that does not use any information about the group membership of the candidates or the values of the protected attributes. The algorithm demonstrated its ability to balance fairness across multiple protected attributes and achieved comparable performance to state-of-the-art when the information about group membership is corrupted by noise. It may be implemented into a ranking pipeline or serve as a proof of concept for achieving fairness through noise injection.

One direction for future work is a rigorous theoretic analysis of the effects of Mallows noise injection on fairness, providing a bound for the values of the most common fairness metrics. This would allow users to tune the parameters of the model depending on the distribution of the values of the protected attributes, improving the algorithm's performance without requiring information about individual candidates' group membership.

References

[1] S. A. Friedler, C. Scheidegger, and S. Venkatasubramanian, "On the (im)possibility of fairness," 2016.

[2] J. Angwin, J. Larson, S. Mattu, and P. Lauren Kirchner, "Machine bias," https://web.archive.org/web/20240322030010/https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing, [Online; Accessed: 2024-25-03].

[3] T. Bolukbasi, K.-W. Chang, J. Zou, V. Saligrama, and A. Kalai, "Man is to computer programmer as woman is to homemaker? debiasing word embeddings," 2016.

[4] V. Hofmann, P. R. Kalluri, D. Jurafsky, and S. King, "Dialect prejudice predicts ai decisions about people's character, employability, and criminality," 2024.

[5] European Parliament, "European Parliament legislative resolution of 13 March 2024 on the proposal for a regulation of the European Parliament and of the Council on laying down harmonised rules on Artificial Intelligence (Artificial Intelligence Act) and amending certain Union Legislative Acts (COM(2021)0206 – C9-0146/2021 – 2021/0106(COD))," 2024.

[6] N. Y. T. Meredith Broussard, "When algorithms give real students imaginary grades," https://web.archive.org/web/20240223090531/https://www.nytimes.com/2020/09/08/opinion/international-baccalaureate-algorithm-grades.html, [Online; Accessed: 2024-25-03].

[7] A. A. Cabrera, W. Epperson, F. Hohman, M. Kahng, J. Morgenstern, and D. H. Chau, "Fairvis: Visual analytics for discovering intersectional bias in machine learning," in *2019 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, 2019.

[8] Z. Jin, M. Xu, C. Sun, A. Asudeh, and H. V. Jagadish, "Mithracoverage: A system for investigating population bias for intersectional fairness," in *Proceedings of the 2020 ACM*

SIGMOD International Conference on Management of Data, ser. SIGMOD '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 2721–2724.

[9] J. Li, Y. Moskovitch, and H. V. Jagadish, "Denouncer: Detection of unfairness in classifiers," *Proc. VLDB Endow.*, vol. 14, pp. 2719–2722, 2021.

[10] ——, "Detection of groups with biased representation in ranking," 2023.

[11] T. Qin, T.-Y. Liu, J. Xu, and H. Li, "Letor: A benchmark collection for research on learning to rank for information retrieval," *Information Retrieval*, vol. 13, pp. 346–374, 2010.

[12] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 89–96. [Online]. Available: https://doi.org/10.1145/1102351.1102363

[13] C. Burges, R. Ragno, and Q. Le, "Learning to rank with nonsmooth cost functions," in *Advances in Neural Information Processing Systems*, B. Schölkopf, J. Platt, and T. Hoffman, Eds., vol. 19. MIT Press, 2006. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2006/file/af44c4c56f385c43f2529f9b1b018f6a-Paper.pdf

[14] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao, "Adapting boosting for information retrieval measures," *Information Retrieval*, vol. 13, no. 3, pp. 254–270, Jun 2010. [Online]. Available: https://doi.org/10.1007/s10791-009-9112-1

[15] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: from pairwise approach to listwise approach," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 129–136.

[16] M. Taylor, J. Guiver, S. Robertson, and T. Minka, "Softrank: Optimising non-smooth rank metrics," in *WSDM 2008*, February 2008. [Online]. Available: https://www.microsoft.com/en-us/research/publication/softrank-optimising-non-smooth-rank-metrics/

[17] M. Ibrahim and M. Carman, "Comparing pointwise and listwise objective functions for random-forest-based learning-to-rank," vol. 34, no. 4, aug 2016.

[18] P. Lahoti, K. P. Gummadi, and G. Weikum, "ifair: Learning individually fair data representations for algorithmic decision making," 2019.

[19] M. Zehlike, T. Sühr, C. Castillo, and I. Kitanovski, "Fairsearch: A tool for fairness in ranked search results," in *Companion Proceedings of the Web Conference 2020*, ser. WWW '20. ACM, Apr. 2020. [Online]. Available: http://dx.doi.org/10.1145/3366424.3383534

[20] A. Singh and T. Joachims, "Policy learning for fairness in ranking," 2019.

[21] A. Beutel, J. Chen, T. Doshi, H. Qian, L. Wei, Y. Wu, L. Heldt, Z. Zhao, L. Hong, E. H. Chi, and C. Goodrow, "Fairness in recommendation ranking through pairwise comparisons," 2019.

[22] S. C. Geyik, S. Ambler, and K. Kenthapadi, "Fairness-aware ranking in search & recommendation systems with application to linkedin talent search," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 2221–2231. [Online]. Available: https://doi.org/10.1145/3292500.3330691

[23] D. Wei, M. M. Islam, B. Schieber, and S. Basu Roy, "Rank aggregation with proportionate fairness," in *Proceedings of the 2022 International Conference on Management of Data*, ser. SIGMOD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 262–275. [Online]. Available: https://doi.org/10.1145/3514221.3517865

[24] E. Pastor, L. de Alfaro, and E. Baralis, "Identifying biased subgroups in ranking and classification," 2021.

[25] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[26] C. L. Mallows, "Non-null ranking models. I," *Biometrika*, vol. 44, no. 1/2, pp. 114–130, 1957.

[27] R. K. E. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilović, S. Nagar, K. N. Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, K. R. Varshney, and Y. Zhang, "Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias," *IBM Journal of Research and Development*, vol. 63, no. 4/5, pp. 4:1–4:15, 2019.

[28] M. Zehlike, K. Yang, and J. Stoyanovich, "Fairness in ranking, part i: Score-based ranking," *ACM Comput. Surv.*, vol. 55, no. 6, dec 2022. [Online]. Available: https://doi.org/10.1145/3533379

[29] ——, "Fairness in ranking, part ii: Learning-to-rank and recommender systems," *ACM Comput. Surv.*, vol. 55, no. 6, dec 2022. [Online]. Available: https://doi.org/10.1145/3533380

[30] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," 2011.

[31] S. Barocas, M. Hardt, and A. Narayanan, *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, 2023.

[32] A. Chouldechova and A. Roth, "The frontiers of fairness in machine learning," 2018.

[33] A. J. Biega, K. P. Gummadi, and G. Weikum, "Equity of attention: Amortizing individual fairness in rankings," *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018.

[34] A. Singh and T. Joachims, "Fairness of exposure in rankings," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '18. ACM, Jul. 2018. [Online]. Available: http://dx.doi.org/10.1145/3219819.3220088

[35] M. Zehlike, F. Bonchi, C. Castillo, S. Hajian, M. Megahed, and R. Baeza-Yates, "Fa*ir: A fair top-k ranking algorithm," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. ACM, Nov. 2017. [Online]. Available: http://dx.doi.org/10.1145/3132847.3132938

[36] W. Cook and A. Rohe, "Computing minimum-weight perfect matchings," *INFORMS Journal on Computing*, vol. 11, no. 2, pp. 138–148, 1999. [Online]. Available: https://doi.org/10.1287/ijoc.11.2.138

[37] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey, "An experimental comparison of click position-bias models," in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, ser. WSDM '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 87–94. [Online]. Available: https://doi.org/10.1145/1341531.1341545

[38] T. Joachims and F. Radlinski, "Search engines that learn from implicit feedback," *Computer*, vol. 40, no. 8, pp. 34–40, August 2007. [Online]. Available: https://www.microsoft.com/en-us/research/publication/search-engines-that-learn-from-implicit-feedback/

[39] C. Fabien and I. Ekhine, "Concentric mixtures of mallows models for top-k rankings: sampling and identifiability," 2020.

[40] H. Hofmann, "Statlog (German Credit Data)," UCI Machine Learning Repository, 1994, DOI: https://doi.org/10.24432/C5NC77.

[41] K. Yang and J. Stoyanovich, "Measuring fairness in ranked outputs," 2016.

Acknowledgments

This is the acknowledgments section.