CounterFair: Group Counterfactuals for Bias Detection, Mitigation and Subgroup Identification

Alejandro Kuratomi[1], Zed Lee[1], Panayiotis Tsaparas[2], Guilherme Dinis Junior[1],
Evaggelia Pitoura[2], Tony Lindgren[1], and Panagiotis Papapetrou[1]

[1]Department of Computer and Systems Sciences, Stockholm University, Sweden
Email: {alejandro.kuratomi,zed.lee,guilherme,tony,panagiotis}@dsv.su.se
[2]Department of Computer Science and Engineering, University of Ioannina and Archimedes/Athena RC, Greece
Email: {tsap,pitoura}@uoi.gr

2024 IEEE International Conference on Data Mining (ICDM) | 979-8-3315-0668-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICDM59182.2024.00025

***Abstract*—Counterfactual explanations can be used as a means to explain a models decision process and to provide recommendations to users on how to improve their current status. The difficulty to apply these counterfactual recommendations from the users perspective, also known as burden, may be used to assess the models algorithmic fairness and to provide fair recommendations among different sensitive feature groups. We propose a novel model-agnostic, mathematical programming-based, group counterfactual algorithm that can: (1) detect biases via group counterfactual burden, (2) produce fair recommendations among sensitive groups and (3) identify relevant subgroups of instances through shared counterfactuals. We analyze these capabilities from the perspective of recourse fairness, and empirically compare our proposed method with the state-of-the-art algorithms for group counterfactual generation in order to assess the bias identification and the capabilities in group counterfactual effectiveness and burden minimization.**

***Index Terms*—Counterfactual explanations, Algorithmic Fairness, Group counterfactuals, Local explainability**

I. Introduction

Counterfactual explanations help us understand opaque machine learning (ML) models by exploring 'what-if' scenarios for individual instances [1]. The word *counterfactual* may be used as a noun to refer to the instances that are counterfactual points themselves, or as an adjective to describe the points, the explanations, or the reasoning itself. We will use the shorthand *CF* to abbreviate the word "counterfactual". Given a dataset and a trained classifier that maps input instances to class labels, CF explanations can highlight the relevant feature value changes for an instance of interest that would result in an alternative predicted class label [1]–[3]. Consequently, a CF is also known as a *recourse* [4], since it suggests actions to improve the situation of a given instance [1], [2], [4]–[6]. For example, CF explanations may highlight the changes on the features of an individual (e.g., marital status, habits, education, occupation) to obtain a positive answer on a loan application, or to move from a low-wealth to a high-wealth status [4], [5]. Usually, the closest point with the desired label is selected as a CF for the given instance [1], since that reduces the feature changes the instance must apply to reach the desired label.

Nonetheless, with sensitive features, such as gender, race, or age, the suggested changes may hide biases across the sensitive groups, which are not trivial to detect or measure.

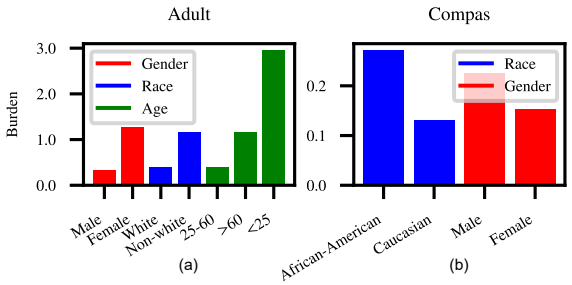


Fig. 1: Burden for sensitive groups (x-axis) belonging to different features (colors), showing biases on Adult and COMPAS.

These biases, if left undetected or unattended, could lead to unfair and harmful outcomes. The assessment of model biases or algorithmic fairness through the recommendations suggested by CFs is known as *counterfactual fairness* [6]–[9]. For example, consider the fairness assessment of two commonly used public datasets: Adult[1] and COMPAS[2]. For Adult, the class label indicates whether a person earns over $50K/year or not, whereas for COMPAS it indicates whether a person is a recidivist (a person recommitting crimes). Fig. 1 illustrates the average difficulty in achieving the desired state (i.e., high wealth or no recidivism) in the form of a measure called *burden* [4], [5]. Burden is the distance between an instance and its closest CF, and the figure shows its aggregated value per sensitive group. We observe a higher aggregated burden for females than for males in wealth prediction (Fig. 1a), implying that it is harder for females to achieve higher wealth. Moreover, we observe that it is harder for males than for females to not be recidivists (Fig. 1b). Similarly, it is harder for non-white people to achieve higher wealth, and harder for African-Americans to not be recidivists.

While the generation of individual CFs provides personalized and actionable recommendations, these CFs may include biases across sensitive feature groups, such as the ones observed in Fig. 1. Such biases cannot be easily mitigated when

[1]https://archive.ics.uci.edu/dataset/2/adult
[2]https://www.kaggle.com/datasets/danofer/compass

2374-8486/24/$31.00 ©2024 IEEE
DOI 10.1109/ICDM59182.2024.00025

generating individual CFs, since each individual CF ignores other groups, leading to biased CF recommendations.

This is a problem when trying to explain biased models that are already trained. For example, if a group of 10 males and a group of 100 females receive a loan rejection, and the average recommendation for males is to increase their salary by \$10K, while for females the recommended increase is by \$25K, the model is biased not only in its predictions but also in the CF recommendations. Hence, the model should be retrained to address these biases, so that these groups receive similar recommendations in relation to their gender or any other sensitive feature. To solve this, we consider not an individual but a group-based CF generation approach that permits the assessment of fairness across sensitive groups.

Existing group CF algorithms obtain CFs through rule mining and *effectiveness* maximization [7], [10], which is the ratio of instances in the group that can apply changes to their own features to reach the feature values of the group CF. These algorithms follow two steps: (1) mine the frequent subgroups of the undesired class label and (2) mine the frequent subgroups of the desired class. While these methods provide subgroup identification and group CFs for fairness assessment, they suffer from two drawbacks: (1) the problem of fair CF generation is not addressed and (2) the relevant subgroups are not selected through the CF generation process. As a result, the solution space is limited and the generated CFs have a high burden and low effectiveness. Existing attempts to address these drawbacks are constrained to linear or decision-tree based models [8] and by the inability to output fair CFs among sensitive groups. We propose a method that can address these two drawbacks while also providing fairness assessment.

In this work we make the following contributions:

- **Novelty:** We propose CounterFair, a model-agnostic CF generation algorithm for assessing group CF fairness and generating fair CF recommendations. CounterFair can prioritize either burden minimization, subgroup identification, or fair recourse generation, leading to either more granular, group-oriented, or fairness-oriented CFs.
- **Bias detection:** We demonstrate the bias detection ability of CounterFair. We further provide an analysis of the results for individual and group CFs, showing that the bias detection capacity is related to the number of distinct groups and is increased when minimizing for burden.
- **Actionability-oriented fairness:** We show the ability of CounterFair to generate fair CF recommendations by minimizing the burden variance among sensitive groups.
- **Evaluation:** Our experimental evaluation shows that CounterFair outperforms state-of-the-art group CF competitors on six public fairness-related datasets in terms of burden minimization and effectiveness maximization.

II. Related work

CF explanations are usually obtained for single instances [1], [2], [4], [11]. Different application scenarios are considered, including recommender system explanations for individual user-item combinations [12]. A few recent studies focused on group CFs [7], [8], [10]. There are several ways to generate group CFs. One way is to jointly generate a CF for each instance, to, for example, make their distribution similar to that of the dataset (*one-to-one* way) [8], [13]. Another way is to get several CFs for a single instance to maximize recourse diversity (*many-for-one* way) [14]. Finally, getting a CF for a group to characterize its instances (*one-for-many* way) [8].

Finding optimal group CFs is analogous to optimally locating facilities, such as hospitals or production plants [15]. This is known as the location analysis problem solved using mathematical programming (MP) [8]. CF algorithms, like the Actionable Recourse [11], use MP but are constrained to linear classifiers due to the difficulty in formulating the nonlinearities of highly accurate ML models. To preserve the formulation linearity and convexity, as well as the solutions optimality, one may apply a graph-oriented approach [16].

Other group CF approaches exist: Kavouras et al. [7] and Rawal and Lakkaraju [10] developed Fairness Aware Counterfactuals for Subgroups (FACTS) and Actionable Recourse Summaries (AReS), respectively, to generate group CFs through rule mining. The rules are in the form of a predicate and an action, e.g., *if gender == female then salary* $\geq$ *\$80K*. FACTS first finds subgroups from the undesired class label instances and sets of actions for these subgroups from the desired class using FP-growth. Then, for these subgroups, the algorithm finds their intersection, i.e., the subgroups or feature-value combinations that are common across them. These common subgroups are then used to find, from the space of actions, a set of valid, effective actions that have the same cost for the individuals on each subgroup. FACTS then uses a set of measures to establish whether there is a bias given the found CFs among the different sensitive groups. AReS is similar to FACTS but it extracts both predicates and actions from the training dataset. The recourse rules are then selected using an optimization procedure that aims to maximize the *correctness* (the fraction of instances for which the recourse rules effectively create a CF), the coverage (the amount of instances for which the "if" conditions apply) and the interpretability (the amount of recourse rules, their length and the number of subgroups). Particularly, these two methods also allow the user to identify subgroups of interest inside the sensitive groups (e.g. in the females sensitive group, those from the EU who are divorced) via the identified and stored predicates for the CF rules. Kavouras et al. [7], and Rawal and Lakkaraju [10] also discuss quality measures to assess the group CFs and the algorithmic fairness based on the CFs.

There are many quality measures to assess CF explanations [1], [4], [17]. These measures include proximity and likelihood (how likely is the given CF regarding the dataset [13]) among others. Likewise, there are many algorithmic fairness measures. These may benefit from CF reasoning. For example, predictive equality (the false negative ratio of a sensitive group), which assesses biases jointly using the prediction and ground truth labels across groups, may miss the potential bias detection capability through the CF recommendations provided by the CFs [18]. Sharma et al. [5] propose CF burden

as a fairness proxy, while Kuratomi et al. [6] weighted the sensitive group burden with its predictive equality, combining accuracy-based and CF fairness. Group CFs quality may also be measured through effectiveness [7].

III. Preliminaries

Let $\mathcal{X}$ be a heterogeneous feature space with binary, categorical, ordinal, and continuous features. A dataset $\mathcal{D}$ is a collection of n pairs of (X, y) where X is a data sample (i.e., instantiation) of $\mathcal{X}$ and y is its corresponding binary class label $y \in \{\text{“}-\text{”}, \text{“}+\text{”}\}$. $\mathcal{D}$ is divided into a training and a test set, denoted as $\mathcal{D}_{Train}$ and $\mathcal{D}_{Test}$, respectively.

Moreover, let $\mathcal{S} \subseteq \mathcal{X}$ be a set of sensitive features in $\mathcal{X}$, such as *sex* or *race*. Each sensitive feature $s \in \mathcal{S}$ may be used to define different *sensitive groups* of data samples. A sensitive group of feature s is denoted as s_k, where k defines a condition on that feature, which is denoted by function $cond(\cdot)$. If s satisfies condition k then *cond(s,k) == 'true'*. For example, sensitive feature *sex* may be used to define two sensitive groups, i.e., s_{female} and s_{male}, corresponding to data samples for which *sex*==‘female’ and *sex*==‘male’, respectively. Given a classifier $f(\cdot)$, we define the set of false-negative test instances in sensitive group s_k as:

$$\mathcal{D}^{s_k}_{TestFN} = \{(X, \text{“}+\text{”}) | f(X) = \text{“}-\text{”}, cond(s,k), X \in \mathcal{D}_{Test}\}$$

Additionally, we introduce the property of *feasibility*. A CF is *feasible* with respect to an instance of interest if it complies with the properties of *mutability*, *directionality*, and *plausibility*. A CF complies with the mutability property if only mutable features are changed from its corresponding instance in $\mathcal{D}_{TestFN}$. In the same manner, it complies with directionality if the features are changed only in possible directions, e.g., age or education cannot decrease. Finally, plausibility indicates that the CF feature values have all physically possible values.

For each instance $X_i \in \mathcal{D}^{s_k}_{TestFN}$ we use a CF generator to get its CF, X'_i. Let us now define the set of possible CFs for the instances in $\mathcal{D}_{TestFN}$ as $\mathcal{Q}$ and the function $F(\mathcal{D}_{TestFN}, \mathrm{Q})$ as an indicator of the instances in $\mathcal{Q}$ that comply with the feasibility condition with respect to the instances in $\mathcal{D}_{TestFN}$. We now introduce the general problem formulation.

Problem 1 (Bias detection, mitigation & Identification of relevant subgroups): Given a classifier $f(\cdot)$ and the set of false-negative test instances $\mathcal{D}_{TestFN} = \bigcup_{s_k} \mathcal{D}^{s_k}_{TestFN}$, we want to obtain the set of CFs $\mathcal{D}'$ as follows:

$$\begin{aligned}\mathcal{D}' = \arg\min_{\mathrm{Q}} \{ & w_1 C_{burden}(Q) + w_2 C_{fair}(Q) + \\ & w_3 C_{groups}(Q) \mid F(\mathcal{D}_{TestFN}, \mathrm{Q})\},\end{aligned} \tag{1}$$

where w_1, w_2 and w_3 are the weights for the costs associated to: (1) the aggregated CFs burden (C_{burden}), (2) bias mitigation or fairness (C_{fair}) and (3) the number of relevant subgroups identified (C_{groups}), respectively.

The formulation in problem 1 enables a flexible cost function definition to allow for the extraction of CFs that optimize different objectives. When prioritizing C_{burden}, burden is minimized, and when aggregated by sensitive groups, this measure elicits the biases among sensitive groups, indicating which groups require a higher effort to achieve the desired label. When prioritizing C_{fair}, the differences in burden across sensitive groups is reduced. This provides fair CF recommendations across different groups, since these would have similar application difficulty, as measured by burden. When prioritizing C_{groups}, a set of CFs that is minimal in size is obtained, which forces the CFs to be *shared* among the instances of interest, i.e., each of the false negative instances will be subgrouped together with other instances, based on the shared CF, generating group CFs and identifying subgroups of interest simultaneously. We now explain CounterFair, the instantiation of the cost functions, C_{burden}, C_{fair} and C_{groups} used, and how CounterFair solves problem 1.

IV. CounterFair

CounterFair is an MP-based algorithm that attains feasible and optimal group CFs in terms of a given cost function. This cost function is adaptable and can be defined in different ways. In our case, we define so that burden (leading to bias detection), burden differences (leading to bias mitigation) or the number of different CFs (indetifying relevant subgroups) are minimized. We first provide an outline of the main steps of CounterFair, the instantiation of the cost function for CounterFair, and finally its MP formulation.

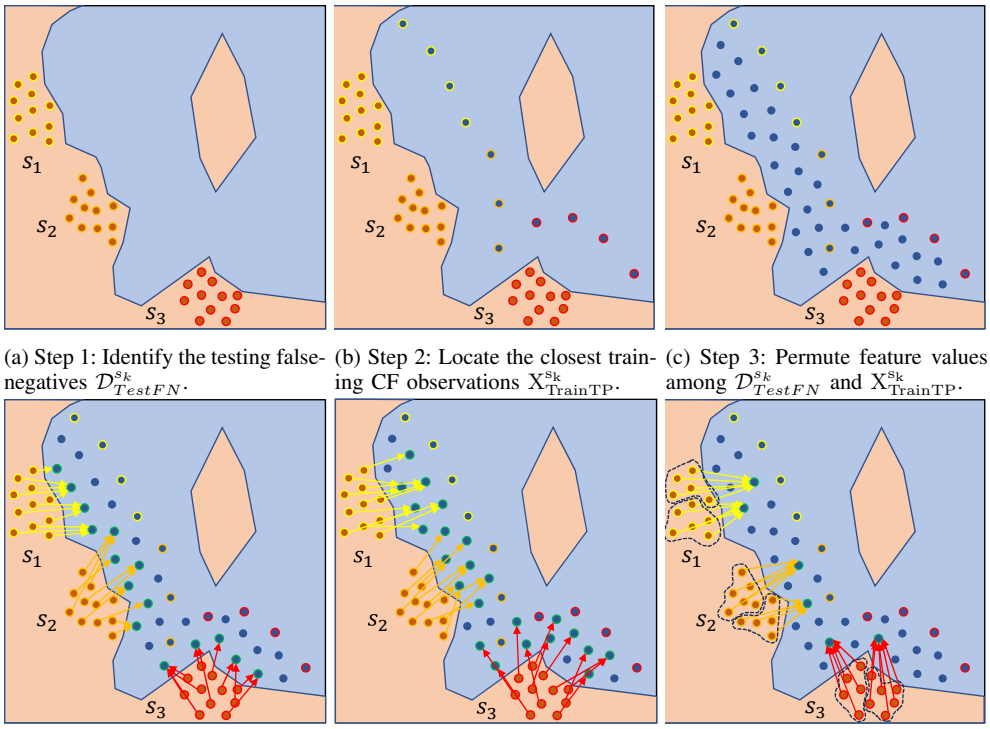
A. Outline

CounterFair creates a set of *points* from which it selects an optimal set of CFs given a cost function, following four steps: (1) obtain the sets of false-negative test instances $\mathcal{D}^{s_k}_{TestFN}$ and true-positive training instances $\mathcal{D}^{s_k}_{TrainTP}$ per sensitive group; (2) obtain the nearest neighbor training CF for each instance in $\mathcal{D}^{s_k}_{TestFN}$ from the instances in $\mathcal{D}^{s_k}_{TrainTP}$, which are then stored in set $\mathrm{CF}^{\mathrm{s_k}}_{\mathrm{Train}}$; (3) find all the combinations of the feature values between each $X \in \mathcal{D}^{s_k}_{TestFN}$ and every CF in $\mathrm{CF}^{\mathrm{s_k}}_{\mathrm{Train}}$ to generate a cloud of points, $\mathcal{P}$, which are the potential CFs; (4) solve the MP to select the best CFs simultaneously for every $X \in \mathcal{D}^{s_k}_{TestFN}$ using $\mathcal{P}$.

The steps of CounterFair are detailed in Alg. 1 and depicted in Fig. 2. In step 1 (Fig. 2a), a ML model separates the undesired class (orange-shaded region) from the desired class (blue-shaded region). The orange points in the undesired region represent the false-negatives. Each sensitive group s_1, s_2, and s_3 is outlined in yellow, orange and red, respectively.

In step 2, for each $X \in \mathcal{D}^{s_k}_{TestFN}$, the *Nearest Neighbor* (NN) is used to find the closest training CF observation from $\mathcal{D}^{s_k}_{TrainTP}$ (blue-colored points in Fig. 2b) and store it in $CF^{s_k}_{Train}$. The set $CF^{s_k}_{Train}$ is filtered using: (1) the closest percentage Ω of CFs from the centroid of the set of false-negatives instances of each s_k, and (2) a critical distance d to the false-negative instances. The Ω value depends on the dataset but is usually 100% (see Appendix A), i.e., all the training CFs in $CF^{s_k}_{Train}$ are considered. The distance d is the maximum of the distance averages of each sensitive group.

In step 3 (Fig. 2c), the cloud of blue points $\mathcal{P}$ is generated using all the possible combinations of the feature values



(a) Step 1: Identify the testing false-negatives $\mathcal{D}^{s_k}_{TestFN}$.

(b) Step 2: Locate the closest training CF observations $\mathrm{X}^{\mathrm{s_k}}_{\mathrm{TrainTP}}$.

(c) Step 3: Permute feature values among $\mathcal{D}^{s_k}_{TestFN}$ and $\mathrm{X}^{\mathrm{s_k}}_{\mathrm{TrainTP}}$.

(d) Step 4.1: Solve the MP with a higher weight on CFs burden.

(e) Step 4.2: Solve the MP minimizing burden variance.

(f) Step 4.3: Solve the MP with a higher weight on group formation.

Fig. 2: 2-dimensional example of CounterFair. Steps 2d, 2e and 2f are obtained by prioritizing C_{burden} (minimizing burden), C_{fair} (minimizing burden differences) and C_{groups} (minimizing the set of CFs), respectively.

Algorithm 1: CounterFair

input : D, f, s, $\vec{mut}$ (mutability vector), $\vec{dir}$ (directionality vector), $\vec{pla}$ (plausibility vector), Ω (closest training percentage).

output: $CF_{CounterFair}$

$\mathcal{D}^{s_k}_{TestFN}, \mathcal{D}^{s_k}_{TrainTP} \leftarrow$ `TestFNTrainTP` (D, f, s)

$CF^{s_k}_{Train} \leftarrow$ `NN` $(\mathcal{D}^{s_k}_{TestFN}, \mathcal{D}^{s_k}_{TrainTP}, \Omega)$

$\mathcal{P}, C_{X_in}, F_{X_in} \leftarrow$ `points` $(\mathcal{D}^{s_k}_{TestFN}, CF^{s_k}_{Train}, f, \vec{mut}, \vec{dir}, \vec{pla})$

$CF^{X_in}_{CounterFair} \leftarrow$ `solveMP` $(\mathcal{P}, C_{X_in}, F_{X_in})$

return $CF_{CounterFair}$

between each $X_i \in \mathcal{D}^{s_k}_{TestFN}$ and the CF observations in $\mathrm{CF}^{\mathrm{s_k}}_{\mathrm{Train}}$. All the continuous features are discretized using an equal frequency binning (details of this discretization are shown in Appendix A). The generated points are stored in $\mathcal{P}$, if they: (1) are feasible CFs with respect to the instance from which they are generated, (2) lie inside the critical distance d with respect to this instance. Finally, for each $X_i \in \mathcal{D}^{s_k}_{TestFN}$ and $n \in \mathcal{P}$, two parameters are calculated: (1) a cost parameter C_{X_in}, representing the cost of using point n as the CF for X_i and (2) a feasibility parameter F_{X_in} indicating whether point n is a feasible CF for X_i.

In step 4, the MP is solved in three separate ways, shown in Figures 2d, 2e and 2f, by: (1) minimizing the aggregated burden, (2) minimizing the burden differences and (3) minimizing the number of distinct CFs. Fig. 2d illustrates 15 unique CFs, outlined in green, presenting the lowest sensitive group-aggregated burden (the length of the arrows is minimized). Fig. 2e shows CFs having similar distances from their respective instances of interest, minimizing the burden differences among sensitive groups (the difference in length of the arrows is minimized). In Fig. 2f there are six distinct, shared CFs. There are two unique subgroups per sensitive group (enclosed in the dashed lines), which are the relevant subgroups.

B. Cost function instantiation

To solve problem 1, we instantiate the cost functions C_{burden}, C_{fair} and C_{groups} and describe the MP formulation.

a) C_{burden}*:* In order to define C_{burden}, we hereby introduce the measure of accuracy weighted burden (AWB).

Accuracy Weighted Burden: the AWB measure, introduced in [6], is the product of predictive equality (the false negative ratio) and the average burden per sensitive group. The AWB measure uses a distance function $d(X_i, X'_i)$. This distance is the burden incurred by instance X_i in trying to attain the feature values of X'_i, and it is a combination of the $L1$-norm

and the $L0$-norm (see details in Appendix C). Eq. 2 indicates how to calculate the AWB measure.

$$\mathrm{AWB}^{s_k} = \frac{\sum\limits_{X_i \in \mathcal{D}^{s_k}_{TestFN}} d(X_i, X_i')}{|\{(X,y) \in \mathcal{D}_{Test} | cond(s,k), y = \text{“}+\text{”}\}|}, \quad (2)$$

where the denominator is the amount of true positives in the sensitive group s_k. Eq. 2 indicates that a higher number of false negatives, or a higher distance between each instance and its CF in the s_k sensitive group, make the AWB^{s_k} burden higher. Then, we define C_{burden} as the total AWB:

$$C_{burden} = \sum_{s_k} \mathrm{AWB}^{s_k} \quad (3)$$

b) C_{fair}: We may define the cost associated to the presence of biases by estimating the differences of the burden among the sensitive groups. To do this, we define $\mathrm{AWB}_{min} = \min \mathrm{AWB}^{s_k}$ and $\mathrm{AWB}_{max} = \max \mathrm{AWB}^{s_k}$ as the minimum and maximum burden, respectively. C_{fair} is then defined as the absolute difference between these two terms:

$$C_{fair} = \mathrm{AWB}_{max} - \mathrm{AWB}_{min} \quad (4)$$

c) C_{groups}: To define the cost associated to the number of distinct CFs, we define a variable and a set: $l_n, \forall n \in \mathcal{P}$ as a variable that indicates whether a point $n \in \mathcal{P}$ is selected as a CF for any of the instances $X_i \in \mathcal{D}_{TestFN}$ and $\mathcal{I} = \mathcal{D}_{TestFN} = \bigcup_{s_k} \mathcal{D}^{s_k}_{TestFN}$ as the set of all false-negative instances. Therefore, the cost C_{groups} is defined as:

$$C_{groups} = \frac{\sum\limits_{n \in \mathcal{P}} l_n}{|\mathcal{I}|} \quad (5)$$

In the worst case scenario, every instance $X_i \in \mathcal{I}$ will have its own unique CF, making the cost $C_{groups} = 1$. We now continue with the MP formulation of the CounterFair algorithm and show how it solves problem 1.

C. CounterFair MP Formulation

To solve problem 1, we split the implementation of CounterFair into two: one main implementation focusing on minimizing C_{burden} and C_{groups}, and another focusing on minimizing C_{fair}, the latter requiring additional variables and constraints over the main implementation. In the main implementation, the MP uses only integer decision variables, making the MP an integer program. In the second implementation, a set of continuous decision variables must be added to the main implementation, which makes the MP a mixed integer linear program. We proceed to present the main formulation and then describe the added variables and constraints for the second.

We define the set of binary decision variables $p_{X_i n}, \forall X_i \in \mathcal{I},\ n \in \mathcal{P}$. These variables indicate whether the point n is selected as a CF for the instance X_i. In order to relate the C_{burden} cost with the decision variable, we introduce the parameter $\mathrm{AWB}^{s_k}_{X_i n}$, which is the added burden when selecting point n for the instance X_i as a CF, i.e. when $p_{X_i n} = 1$:

$$\mathrm{AWB}^{s_k}_{X_i n} = \frac{d(X_i, n)}{|\{(X,y) \in \mathcal{D}_{Test} | cond(s,k), y = \text{“}+\text{”}\}|}, \quad (6)$$

for all $X_i \in \mathcal{I}$ and for all $n \in \mathcal{P}$. Then, multiplying $\mathrm{AWB}^{s_k}_{X_i n}$ with the decision variable $p_{X_i n}$:

$$\mathrm{AWB}^{s_k} = \sum_{X_i \in \mathcal{I}} \sum_{n \in \mathcal{P}} \mathrm{AWB}^{s_k}_{X_i n} \cdot p_{X_i n}, \quad (7)$$

and then C_{burden} can be rewritten as:

$$C_{burden} = \sum_{s_k} \mathrm{AWB}^{s_k} = \sum_{s_k} \sum_{X_i \in \mathcal{I}} \sum_{n \in \mathcal{P}} \mathrm{AWB}^{s_k}_{X_i n} \cdot p_{X_i n} \quad (8)$$

For the C_{groups} term, we use the binary decision variables $l_n, \forall n \in \mathcal{P}$, and the C_{groups} cost remains as defined in 5. We define the objective function for the main implementation as:

$$Z_1 = \alpha C_{burden} + (1 - \alpha) C_{groups}, \quad (9)$$

where the weight $\alpha \in [0, 1]$. When weight $\alpha \approx 0$, C_{groups} is prioritized and the number of distinct CFs will be minimized. This will force the *sharing* of CFs among similarly-distanced instances, automatically identifying relevant subgroups via these shared CFs. Equivalently, when $\alpha \approx 1$, the total aggregated burden will be minimized, optimizing the recommendations found for each instance. In this scenario, the models biases will be observed as a higher relative aggregated burden for some of the sensitive groups. We now define the block $\mathcal{R}$ of constraints as follows:

$\mathcal{R}$:

$$p_{X_i n} \leq F_{X_i n},\ \forall i \in \mathcal{I}, n \in \mathcal{P}, \quad (10)$$

$$\sum_{n \in \mathcal{P}} p_{X_i n} = 1, \forall X_i \in \mathcal{I}, \quad (11)$$

$$p_{X_i n} \leq l_n\ \forall X_i \in \mathcal{I}, \forall n \in \mathcal{P}, \quad (12)$$

$$p_{X_i n}, l_n \in \{0, 1\}\ \forall X_i \in \mathcal{I}, \forall n \in \mathcal{P}, \quad (13)$$

Constraint (10) guarantees that the selected points n are feasible for their instances X_i, while constraint (11) forces the selection of a single point n per instance X_i. Finally, constraint (12) requires all $p_{X_i n}$ variables to be less than or equal to the limiter l_n, with constraints (13) forcing $p_{X_i n}$ and l_n to be binary. Finally, the main MP formulation is:

$$\min Z_1,\ \text{subject to } \mathcal{R} \quad (14)$$

We now describe the second implementation, which aims at mitigating the biases among sensitive groups. To do this, we take the previously defined variables: AWB_{max} and AWB_{min}, and add them as continuous decision variables. Then we add the following set of constraints to the block $\mathcal{R}$:

$$\mathrm{AWB}_{min} \leq \sum_{X_i \in \mathcal{I}} \sum_{n \in \mathcal{P}} \mathrm{AWB}^{s_k}_{X_i n} \cdot p_{X_i n} \leq \mathrm{AWB}_{max}, \forall s_k, \quad (15)$$

which bounds the aggregated burden per sensitive feature. Then, we define the cost function as:

$$Z_2 = \text{AWB}_{max} - \text{AWB}_{min}, \tag{16}$$

which matches Eq. 4. When minimizing Z_2 the difference between the maximum and minimum burden is reduced down to zero. Given that AWB_{max} and AWB_{min} work as bounds on the burden, it then forces the selection of CFs that have equal burden across sensitive groups and thereby effectively decreasing the biases obtained from the recommendations.

By now, it is possible to conceive other cost functions and potential adaptations to the algorithm to achieve other objectives through the CounterFair MP formulation. Besides the objectives of bias detection, mitigation and subgroup identification, we consider CF effectiveness as a measure to optimize for in group CF generation. We now continue with the formulation to maximize group CF effectivenes.

As previously mentioned, CF effectiveness is the ratio of total instances that have a feasible CF. We define parameter $e_n, \forall n \in \mathcal{P}$, as the effectiveness associated to point n with respect to the instances of interest:

$$e_n = \frac{\sum\limits_{X_i \in \mathcal{I}} F(X_i, n)}{|\mathcal{I}|}, \forall n \in \mathcal{P}, \tag{17}$$

where $F(X_i, n)$ is the feasibility indicator function between the instance X_i and the point n. This parameter, which can be estimated for every point $n \in \mathcal{P}$, indicates the ratio of the instances in the set of false-negatives that can reach the n CF point. Then, the cost function associated to effectiveness in the MP is defined as:

$$Z_3 = C_{eff} = -\sum_{X_i \in \mathcal{I}} \sum_{n \in \mathcal{P}} e_n \cdot p_{X_i n}. \tag{18}$$

We now proceed to discuss the complexity of CounterFair. *Complexity:* The most complex step of the CounterFair algorithm is step 4: the solution of the MP. In general, an integer program formulation, which is harder to solve than a mixed integer linear program formulation, is classified as a NP-complete problem [19]–[21] with a complexity determined by the number of rows (constraints) and columns (variables). Based on the block $\mathcal{R}$ of constraints, the number of constraints is $3(|\mathcal{I}| \cdot |\mathcal{P}|) + |\mathcal{I}| + |\mathcal{P}|$, whilst the number of variables, v, is $|\mathcal{I}| \cdot |\mathcal{P}| + |\mathcal{P}|$. The solution is usually obtained through a branch-and-bound approach, which requires the solution of a relaxed linear program at every node of the branch-and-bound tree. Each linear program solution is estimated to have a complexity of $\mathcal{O}(v^{2.5})$ [22]. The minimum number of nodes in a branch-and-bound tree is determined by $2^{\lfloor \frac{v}{2c} \rfloor}$, where c is the maximum number of variables in any given constraint [23]. In this case, since $c = |\mathcal{P}|$ (see constraint 11) then $2^{\lfloor \frac{v}{2c} \rfloor} = 2^{\lfloor \frac{|\mathcal{I}| \cdot |\mathcal{P}| + |\mathcal{P}|}{2|\mathcal{P}|} \rfloor} = 2^{\lfloor \frac{|\mathcal{I}|+1}{2} \rfloor}$. Therefore, the total complexity of CounterFair is $\mathcal{O}(2^{\lfloor \frac{|\mathcal{I}|+1}{2} \rfloor}(|\mathcal{I}| \cdot |\mathcal{P}| + |\mathcal{P}|)^{2.5})$.

V. Empirical evaluation

In this section, we illustrate the experimental setup by describing the CF evaluation measures, the datasets and the classification performance achieved. We then evaluate CounterFair and compare it with AReS [10] and FACTS [7].

A. Experimental Setup

We compare the aggregated AWB values of the sensitive groups using Eq. 7 and the number of subgroups obtained by summing the limiter variable l_n for each of the sensitive groups. We define the set of points in the cloud of points $\mathcal{P}$ that belong to sensitive group s_k as $\mathcal{P}^{s_k}$. Then, we define $L^{s_k} = \sum\limits_{n \in \mathcal{P}^{s_k}} l_n$ as the number of distinct points selected as CFs for s_k. We calculate the effectiveness of the CFs per sensitive group, which is defined as $E^{s_k} = \frac{|\{X_i \in \mathcal{D}^{s_k}_{TestFN} | F(X_i, X_i')\}|}{|\{X_i \in \mathcal{D}^{s_k}_{TestFN}\}|}$.

Moreover, we used six binary classification datasets. These datasets cover different application domains and sensitive groups, focusing mainly on gender, age, and race [24]. All datasets have been preprocessed and stored in our GitHub repository[3]. The preprocessing is based on Karimi et al. [2] and Le Quy et al. [24]. Further details about the CounterFair parameters and the features for each dataset may be found in Appendix A and the repository.

Additionally, we trained a Random Forest (RF) and a Multi-Layer Perceptron (MLP) classifier and tuned their parameters using a 70%/30% train/test split and a grid search tuning on the training set. We used the F1 score as our classification metric. Details on performance, computing unit used and structure of the classifiers are provided in Appendix B and the repository.

Finally, we benchmark CounterFair on three scenarios, one for each of the described cost functions in IV-C: (1) with cost function Z_1 and three different values of α: $\alpha = [0.1, 0.5, 1.0]$ for all datasets. and obtain the AWB^{s_k} and L^{s_k} scores for each sensitive group s_k; (2) with cost function Z_2, to reduce the biases based on the aggregated burden; (3) with cost function Z_3 to showcase the adaptability of CounterFair and its performance when optimizing for effectiveness, as a group CF measure that has been previously prioritized by other methods. We compare the performance in terms of burden and effectiveness with AReS and FACTS.

B. Results

We present here the results of the experiments carried out with respect to 5 elements: (1) burden minimization for bias detection, (2) minimization of burden differences among sensitive groups for bias mitigation and fair recommendations, (3) impact of minimizing differences of burden and differences of distance on the group CF recommendations, (4) minimization of distinct CFs for relevant subgroup identification and (5) comparison of CounterFair with AReS and FACTS in burden, effectiveness and run times.

[3]https://github.com/alku7660/CounterFair

1) Burden minimization: in the first experiment, we ran CounterFair to minimize cost function Z_1 with $\alpha = [0.1, 0.5, 1.0]$, i.e., starting with a low weight of 0.1 for the C_{burden} cost and a high weight of 0.9 for the C_{groups} cost, and ending with a high weight of 1.0 for C_{burden} and 0.0 for C_{groups}. Fig. 3 shows the aggregated burden per sensitive group as the bars when α increases for each dataset. The burden decreases as α increases for all the datasets, and shows the differences in burden among different sensitive groups.

The differences in burden across sensitive groups is best evidenced when using the highest $\alpha = 1.0$ because the nearest and easiest CF is selected for each instance. Since the aggregated burden for each sensitive group is calculated through Eq. 7, a higher number of false-negative instances (the number in parenthesis in the legend of each plot) would normally portray a higher burden for a given group. However, this is not always the case. For example, prioritizing AWB ($\alpha = 1.0$) in the German dataset does not show the same relative AWB behavior among genders as in the other two values for α and, even though there are less than half as many females as males, females present a higher AWB.

2) Minimization of burden differences among sensitive groups: in this experiment we ran CounterFair to minimize cost function Z_2, i.e., the differences in burden among the CFs. Fig. 3 shows the result of this experiment in the last set of bars on each plot, over the *Fair* x-axis label. Note that the obtained CFs show an equal burden as measured by AWB^{s_k} among the sensitive groups for each dataset, effectively eliminating the biases in burden and producing group CF recommendations that are fair across these groups.

3) Impact of minimizing the differences of burden among groups (AWB^{s_k}): we illustrate the impact of the minimization of burden differences in the CounterFair CF recommendations for the false-negative instances in the German dataset and compare it to the minimization of distance differences, i.e., not considering the false negative ratio of the model. We randomly pick a male and a female from the $\mathcal{D}_{TestFN}$ set. We then run CounterFair on this dataset minimizing instead the differences in distance cost $\sum_{\mathcal{D}_{TestFN}} d(X_i, X_i')$ and extract the CFs for the selected male and female. The CFs (minimizing AWB differences and distance differences) are shown on table I. In the CFs minimizing AWB differences, the credit change is larger for the female than for the male, compensating for the higher amount of false-negative males (there are 8 false-negative females and 28 false-negative males). When minimizing for distance differences, there is no consideration of the false negative ratio, so there is no compensation for the bias in accuracy, and the credit change is smaller for the females, although the rate also decreased.

We highlight the importance of having both the bias detection-oriented CF generation first, as this would allow the users to detect potential sensitive feature biases present in the trained model, and then, if required, generate fair CFs by minimizing the burden in deployed models. This is important, since only running CounterFair for bias mitigation (although beneficial for the fairness in the recommendations to users) might hide the models algorithmic biases. We recommend using CounterFair in the following manner: first identify the biases and then obtain the bias-mitigating CFs if needed, so that the recommendations are fair across groups.

4) Minimization of distinct CFs: We now show the structure of the relevant subgroups identified when minimizing cost function Z_1 with $\alpha = 0.1$. Fig. 4 illustrates the number of relevant CFs and subgroups found for each sensitive group with the diamonds plotted using the secondary y-axis. Note that, as burden is increasingly prioritized, the number of subgroups increases. Fig. 4 shows the details of the seven relevant subgroups identified in the Compas dataset. The red subgroup shows caucasian males with at least a felony and 15 priors, older than 45. The other subgroups are characterized by priors around 6, ages between 25 and 45. These are obtained by aggregating the instances that share the same group CF, as output by CounterFair. In the case of the Compas dataset, the identified subgroups may provide further data to analyse the people that are being misclassified as recidivists and why. For example, in the case of the red group (caucasian males with a felony, almost 15 priors, older than 45), only 22 false-negative instances were found, while for the dark green (caucasian males with a felony, around 6 priors, between 25-45 years of age) there were 210, indicating an almost 10 times larger false negative ratio for the latter. This is interesting, since there are more false negatives in the younger age group, even as there are less prior counts of crimes committed, indicating that the model might have an age bias. The remaining datasets identified subgroup detail plots are uploaded in the repository.

5) Comparison of CounterFair with AReS and FACTS with respect to burden, effectiveness and run time: we ran the experiments of AReS and FACTS with two considerations: (1) following the authors recommended support threshold of 1% [7], [10] and (2) limiting the execution time to maximum 1 week per dataset. However, the threshold had to be modified to run within the time limit, but at most to 10% (beyond this point, the performance significantly degrades). Fig. 5 shows the AWB, effectiveness and run times of CounterFair, AReS and FACTS. CounterFair mostly outperforms AReS and FACTS in burden and effectiveness. AReS and FACTS beat CounterFair in AWB in the Dutch dataset Females (FACTS beats it on Males and Females). AReS also beats CounterFair in Males and Females in AWB in the Athlete dataset. CounterFair significantly beats them in effectiveness in all cases. For AReS this can be explained by the lack of feasibility constraints on the CFs, leading sometimes to infeasible CFs. Timewise, AReS is the fastest, and CounterFair is at least 10 times faster than FACTS except in the Dutch dataset (ran with 10%). We excluded Adult and Student since the recommended threshold of 1% overshot the run time beyond the week, or it had to be raised beyond 10%, hindering the performance.

VI. Conclusions

We propose CounterFair, an MP-based, model-agnostic CF generation algorithm that can detect biases, mitigate them,

	Sex	Single	Unemployment	Purpose	Rate	Housing	Age	Credit	Duration
$X1$	Male	No	No	Electronics	3	Rent	22	1331	1.2
${X1'}_{Fair}^{AWB}$	Male	No	No	Electronics	3	Rent	22	1845	4.5
${X1'}_{Fair}^{dist.}$	Male	No	No	Electronics	3	Rent	22	1871	4.5
$X2$	Female	No	No	Car	4	Owns	34	1842	3.6
${X2'}_{Fair}^{AWB}$	Female	No	No	Car	4	Owns	34	866	15
${X2'}_{Fair}^{dist.}$	Female	No	No	Car	1	Owns	34	1490	15

TABLE I: Instances and their CFs obtained through CounterFair when mitigating biases across sensitive groups in the German dataset. The recommended changes are larger for the female since the model is biased in accuracy favoring females.

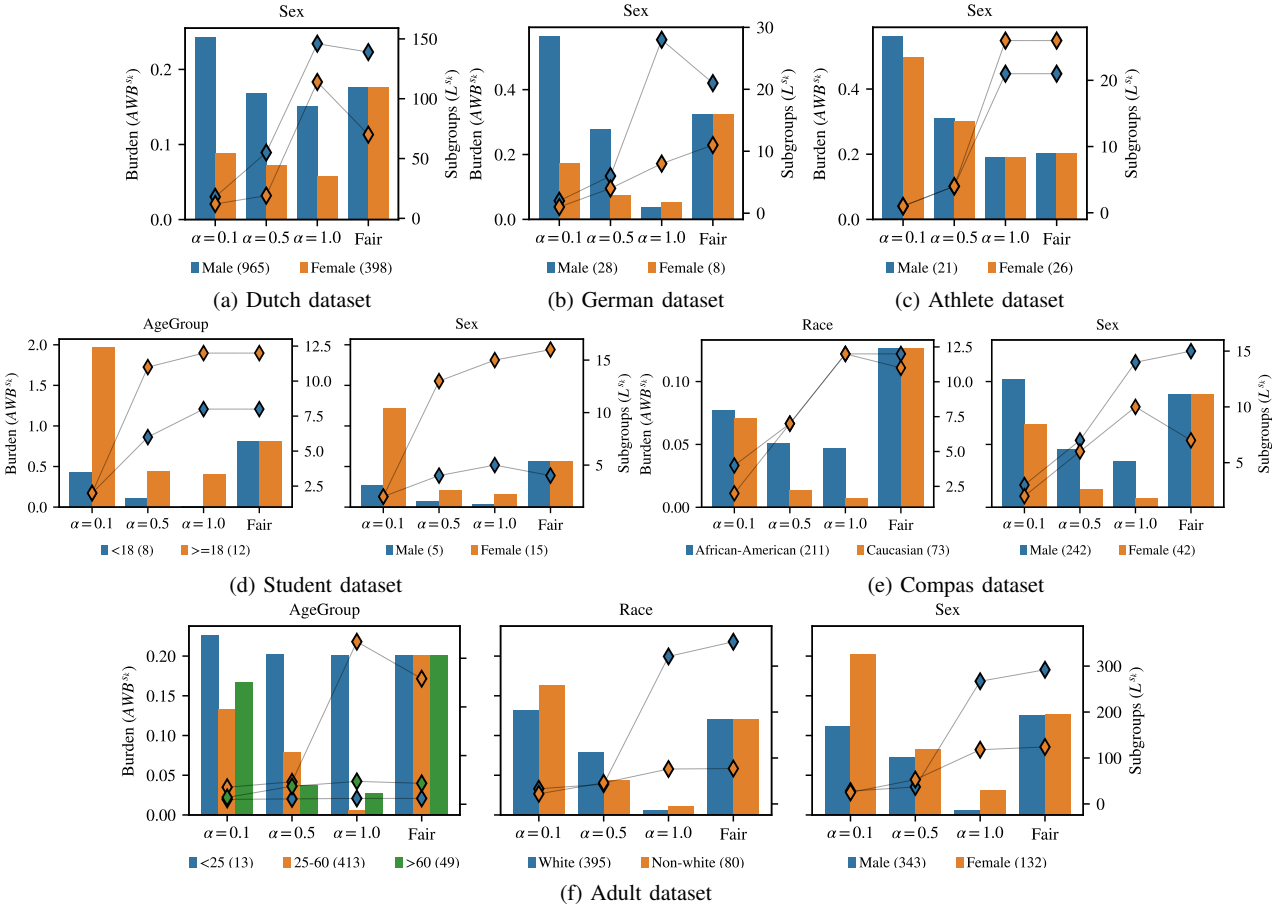


Fig. 3: Aggregated burden, identified subgroups and fair CFs burden output. Each plot has four points in the x-axis: three for the α values of 0.1, 0.5 and 1.0 using cost function Z_1, and one using cost function Z_2. The bars show the burden (on the left y-axis), while the diamonds the number of distinct subgroups for each sensitive group (on the right y-axis). The legends indicate the sensitive groups and their number of false-negatives in parenthesis.

and identify relevant subgroups in the data, all via group CF generation. The generation of group CFs requires only the input of the feature properties of mutability, directionality and possible values. CounterFair is, as demonstrated, adaptable to generate CFs based on different cost functions thanks to its flexibility in cost and constraints definitions. An example is analyzed with group effectiveness, and it is the only group CF generation method, to the best of our knowledge, that is also able to reduce the burden biases among sensitive groups by selecting CFs that decrease the difference in aggregated burden among them. From a holistic perspective, having a tool that is not only able to detect biases, but also extract fair recommendations based on the trained ML models is useful for scientists and developers but also useful for users who are looking to find ways to improve their condition without them turning to be unfair with their peers. As part of future work, other cost functions could be formulated based on the literature on CF explanations quality measures, such as likelihood or

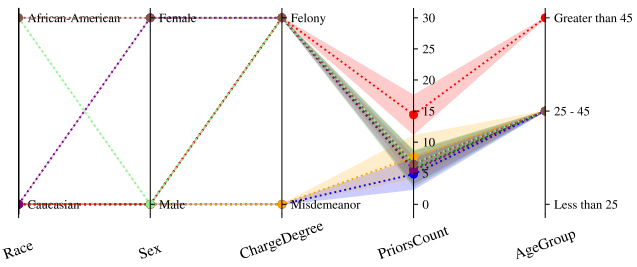


Fig. 4: Compas dataset subgroup details with $\alpha = 0.1$. The shaded regions have a width equal to one standard deviation of the features values of each subgroup.

sparsity, as well as the usage of other commonly used fairness measures. Additionally, the introduction of intersectional fairness: the study of fairness across the specific found subgroups of interest, is a natural step forward. Moreover, the inclusion of the classifiers as nonlinear constraints in the mathematical programming formulations could be researched. Finally, the consideration of non-binary datasets, which should be easy to tackle using, for example, a one-versus-the-rest approach, is also a logical progression, while the scalability and complexity is a good topic to focus on.

Acknowledgements: This work has been supported by project MIS 5154714, and under framework of the H.F.R.I call "Basic Research Financing" (H.F.R.I. Project Number: 016636), both under the National Recovery and Resilience Plan "Greece 2.0" funded by the European Union - NextGenerationEU.

REFERENCES

[1] R. Guidotti, "Counterfactual explanations and how to find them: literature review and benchmarking," *Data Mining and Knowledge Discovery*, pp. 1–55, 2022.

[2] A.-H. Karimi, G. Barthe, B. Balle, and I. Valera, "Model-agnostic counterfactual explanations for consequential decisions," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 895–905.

[3] C. Molnar. (2021) Interpretable machine learning: A guide for making black-box models explainable. [Online]. Available: https://christophm.github.io/interpretable-ml-book/limo.html

[4] A.-H. Karimi, G. Barthe, B. Schölkopf, and I. Valera, "A survey of algorithmic recourse: contrastive explanations and consequential recommendations," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–29, 2022.

[5] S. Sharma, J. Henderson, and J. Ghosh, "CERTIFAI: Counterfactual Explanations for Robustness, Transparency, Interpretability, and Fairness of Artificial Intelligence models," *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 166–172, Feb. 2020, arXiv: 1905.07857. [Online]. Available: http://arxiv.org/abs/1905.07857

[6] A. Kuratomi, E. Pitoura, P. Papapetrou, T. Lindgren, and P. Tsaparas, "Measuring the burden of (un) fairness using counterfactuals," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2022, pp. 402–417.

[7] L. Kavouras, K. Tsopelas, G. Giannopoulos, D. Sacharidis, E. Psaroudaki, N. Theologitis, D. Rontogiannis, D. Fotakis, and I. Emiris, "Fairness aware counterfactuals for subgroups," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[8] E. Carrizosa, J. Ramírez-Ayerbe, and D. R. Morales, "Mathematical optimization modelling for group counterfactual explanations," *European Journal of Operational Research*, 2024.

[9] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," *Advances in neural information processing systems*, vol. 30, 2017.

[10] K. Rawal and H. Lakkaraju, "Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 187–12 198, 2020.

[11] A. Spangher, B. Ustun, and Y. Liu, "Actionable recourse in linear classification," in *Proceedings of the 5th workshop on fairness, accountability and transparency in machine learning*, 2018.

[12] X. Wang, Q. Li, D. Yu, Q. Li, and G. Xu, "Counterfactual explanation for fairness in recommendation," *ACM Transactions on Information Systems*, vol. 42, no. 4, pp. 1–30, 2024.

[13] M. Pawelczyk, K. Broelemann, and G. Kasneci, "Learning model-agnostic counterfactual explanations for tabular data," in *Proceedings of The Web Conference 2020*, 2020, pp. 3126–3132.

[14] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 2020, pp. 607–617.

[15] C. S. ReVelle and H. A. Eiselt, "Location analysis: A synthesis and survey," *European journal of operational research*, vol. 165, no. 1, pp. 1–19, 2005.

[16] P. Avella, A. Sassano, and I. Vasil'ev, "Computational study of large-scale p-median problems," *Mathematical Programming*, vol. 109, pp. 89–114, 2007.

[17] S. Verma, J. Dickerson, and K. Hines, "Counterfactual Explanations for Machine Learning: A Review," *arXiv:2010.10596 [cs, stat]*, Oct. 2020, arXiv: 2010.10596. [Online]. Available: http://arxiv.org/abs/2010.10596

[18] A. Coston, A. Mishler, E. H. Kennedy, and A. Chouldechova, "Counterfactual risk assessments, evaluation, and fairness," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. Barcelona Spain: ACM, Jan. 2020, pp. 582–593. [Online]. Available: https://dl.acm.org/doi/10.1145/3351095.3372851

[19] H. W. Lenstra Jr, "Integer programming with a fixed number of variables," *Mathematics of operations research*, vol. 8, no. 4, pp. 538–548, 1983.

[20] R. Kannan and C. L. Monma, "On the computational complexity of integer programming problems," *Optimization and Operations Research*, pp. 161–172, 1978.

[21] C. H. Papadimitriou, "On the complexity of integer programming," *Journal of the ACM (JACM)*, vol. 28, no. 4, pp. 765–768, 1981.

[22] M. B. Cohen, Y. T. Lee, and Z. Song, "Solving linear programs in the current matrix multiplication time," *Journal of the ACM (JACM)*, vol. 68, no. 1, pp. 1–39, 2021.

[23] A. Basu, M. Conforti, M. Di Summa, and H. Jiang, "Complexity of branch-and-bound and cutting planes in mixed-integer optimization," *Mathematical Programming*, pp. 1–24, 2022.

[24] T. Le Quy, A. Roy, V. Iosifidis, W. Zhang, and E. Ntoutsi, "A survey on datasets for fairness-aware machine learning," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 12, no. 3, p. e1452, 2022.

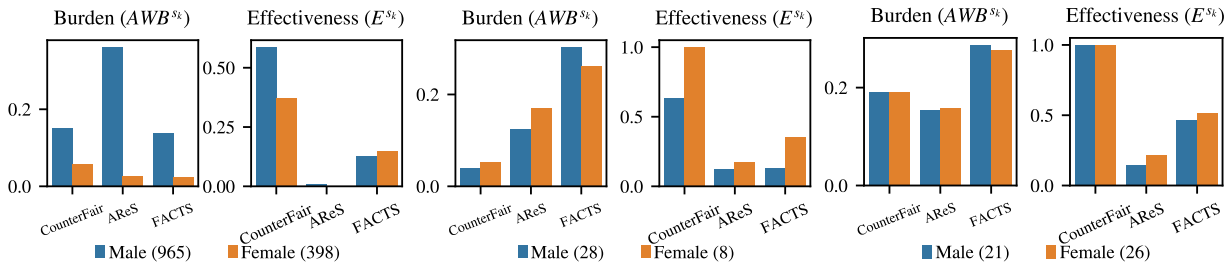
APPENDIX

A. Datasets

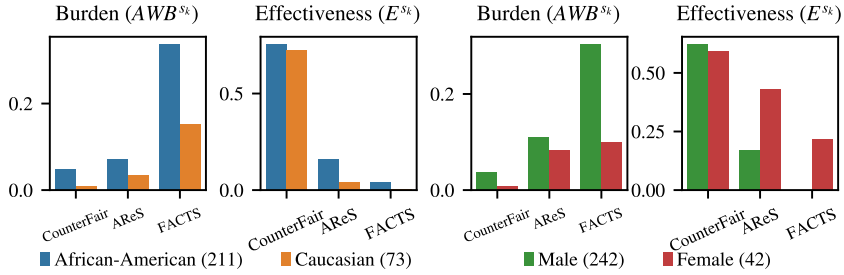
The datasets and their description may be found at the UCI Machine Learning repository[4] unless a different URL is specified. Additionally, you may find them at the repo[5].

The discretization of continuous features is done as follows: for each continuous feature the values found in the training set are used as steps. If there are more than 10 values between the values of the instance and the CF, then a normal distribution with the mean and standard deviation of the continuous feature in the training set is estimated and the bins are defined as having each 10% of this distribution, providing equal frequency of points in each bin. The values of the closest training percentage, Ω and the number of bins used for the discretization of continuous features are specified in Table II.

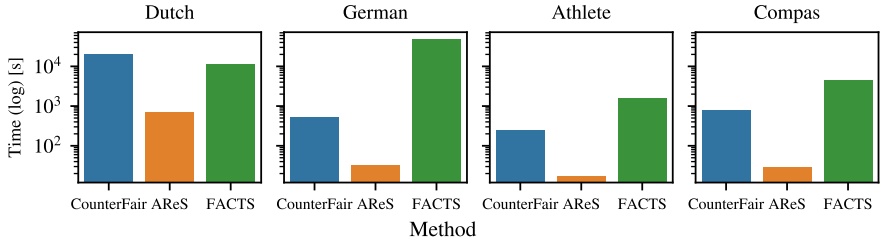
[4]https://archive.ics.uci.edu/ml/index.php
[5]https://github.com/alku7660/CounterFair



(a) Dutch dataset (Sex) (b) German dataset (Sex) (c) Athlete dataset (Sex)



(d) Compas dataset (Race & Sex)



(e) Run times for CounterFair, AReS and FACTS

Fig. 5: CounterFair, AReS and FACTS performance. Lower AWB, higher effectiveness and lower times are better. CounterFair is run with Z_1 and $alpha = 1.0$ for AWB, and with Z_3 for effectiveness.

The properties of mutability and directionality for each of the datasets can be found at the repo.

B. Classifiers

The classifiers performance is shown in Table III.

C. Distance function

The distance function used is:

$$d(X_i, X_i') = \frac{ord + con}{w} l1(X_i, X_i') + \frac{bin + cat}{w} match(X_i, X_i'), \tag{19}$$

where bin, cat, ord and con are the binary, categorical, ordinal and continuous features, respectively ($w = bin + cat + ord + con$). The $match(a, b)$ function is a matching function between categorical features (based on [2], [5]).

	Adult	Athlete	Compas	Dutch	German	Student
Ω	30%	100%	100%	30%	100%	100%
Bins	5	10	10	5	10	10

TABLE II: Closest training percentage and number of bins used for the discretization of continuous features.

Dataset	Model	Hyperparameters	F1-score
Adult	RF	$depth_{max} = 10$, $leaf_{minsize} = 1$, $split_{minsize} = 1$, $n = 100$	0.83
Athlete	MLP	$act. = tanh$, $layers = (20, 50, 10)$, $solver = sgd$	0.71
Compas	MLP	$act. = tanh$, $layers = (20, 10, 10)$, $solver = adam$	0.68
Dutch	RF	$depth_{max} = 10$, $leaf_{minsize} = 3$, $split_{minsize} = 5$, $n = 50$	0.84
German	MLP	$act. = ReLU$, $layers = (100, 10)$, $solver = sgd$	0.70
Student	RF	$depth_{max} = 2$, $leaf_{minsize} = 5$, $split_{minsize} = 2$, $n = 200$	0.70

TABLE III: Selected classifiers and test performance.

1) Parameters: The support threshold of the frequent feature values per dataset are indicated in Table IV:

	Athlete	Compas	Dutch	German
AReS	1%	1%	1%	1%
FACTS	1%	1%	10%	1%

TABLE IV: Support threshold for AReS and FACTS.