FAIR: Fair Adversarial Instance Re-weighting

Andrija Petrović[a], Mladen Nikolić[c], Sandro Radovanović[b], Boris Delibašić[b], Miloš Jovanović[b]

[a] *Singidunum University - Technical Faculty, Danijelova 32, Belgrade, Serbia*
[b] *University of Belgrade - Faculty of Organizational Sciences, Jove Ilica 154, Belgrade, Serbia*
[c] *University of Belgrade - Faculty of Mathematics, Studentski Trg 16, Belgrade, Serbia*

Abstract

With growing awareness of societal impact of artificial intelligence, fairness has become an important aspect of machine learning algorithms. The issue is that human biases towards certain groups of population, defined by sensitive features like race and gender, are introduced to the training data through data collection and labeling. Two important directions of fairness ensuring research have focused on (i) instance weighting in order to decrease the impact of more biased instances and (ii) adversarial training in order to construct data representations informative of the target variable, but uninformative of the sensitive attributes. In this paper we propose a Fair Adversarial Instance Re-weighting (FAIR) method, which uses adversarial training to learn instance weighting function that ensures fair predictions. Merging the two paradigms, it inherits desirable properties from both interpretability of reweighting and end-to-end trainability of adversarial training. We propose four different variants of the method and, among other things, demonstrate how the method can be cast in a fully probabilistic framework. Additionally, theoretical analysis of FAIR models' properties have been studied extensively. We compare FAIR models to ten other related and state-of-the-art models and demonstrate that FAIR is able to achieve a better trade-off between accuracy and unfairness. To the best of our knowledge, this is the first model that merges reweighting and adversarial approaches by means of a weighting function that can provide interpretable information about fairness of individual instances.

Keywords: Fairness, Adversarial training, Instance reweighting, Deep learning, Classification

⋆

Email address: apetrovic@singidunum.ac.rs (Andrija Petrović⋆)

1. Introduction

Machine learning algorithms have lead to many recent breakthroughs in different complex tasks that cannot be solved satisfactorily by domain specific algorithms, such as face detection [1], object detection [2], machine translation [3], facial expression recognition [4], sport prediction [5], etc. With this enormous success in practical applications and its growing presence in everyday life, social issues related to machine learning algorithms are becoming increasingly important. One of the most prominent issues is fairness of machine learning algorithms, related to discrimination and bias [6].

It is well known that in many applications data reflects intended or unintended biases of humans whose actions generated data. Salary prediction [7], credit risk prediction [8], medical prediction [9], personnel planning and recruiting forecasting methods [10], are just some of the examples where data, collected from societal interactions, is biased with respect to age, gender, or race. Therefore, machine learning algorithms will extract and learn biases that are present in the data and these can have a strong discriminative impact towards disadvantaged groups. Improving fairness of biased data and decision procedures based on that data is not only a problem of society, but also a problem of machine learning. It is critical to guarantee that the prediction obtained by machine learning algorithms is based on appropriate information and that the outcomes are not biased towards certain groups of population defined by sensitive features like race and gender [11].

Current techniques for improving fairness fall into three different groups: preprocessing techniques [12, 13], techniques based on optimization at training time [14, 15, 16, 17], and post-processing based ones [18, 19]. State-of-the-art techniques for mitigating bias by preprocessing are based on instance reweighing [20], a technique that assigns weights to instances as means of controlling their influence on the model during training. The good side of such methods is that weights that the method assigns can be interpreted as indicators of instance fairness. The downside is that the preprocessing procedure is oblivious to the properties of the downstream learning task, like loss function used, model architecture, etc. That may result in suboptimal weights with respect to that learning task.

Adversarial training has widely been used for finding Nash equilibrium in minimax (zero-sum) games [21, 22, 23]. Recently, adversarial framework became popular in debiasing deep learning models by introducing two networks, one for predicting output labels and one for predicting sensitive attributes [24, 25, 26, 27]. Both depend on the learnt feature space representation which allows fairly accurate prediction of the output label by the first network, while being maximally uninformative about the sensitive attributes, so that the second network has to fail in its task. While these

methods allow for end-to-end training, they do not provide interpretable information on instance fairness, which is desirable.

In this paper, we propose Fair Adversarial Instance Re-weighting (FAIR) – a novel model for mitigating bias in discriminative dataset by using an adversarial framework to learn an instance reweighing function instead of a new data representation as it is done in previous work. The weighting function can provide interpretable information on instance fairness. Also, FAIR does not perform weighting as preprocessing, but integrates it in the learning procedure so that the learning is performed end-to-end. FAIR consists of three neural networks: the first one is used for determining weights for each instance, the second one for predicting the sensitive attribute, and the third one for predicting the output label. FAIR comes in four variants differing in the weighting method. In the first method (FAIR-scalar), obtained scalar weights are used directly for weighting the log likelihood of corresponding instances, whereas in all other methods instance weights are modelled as random variables parametrized by the weighting network. In the second method (FAIR-Bernoulli) the weights are distributed according to Bernoulli distribution and during learning, score function is used to evaluate the expectation of the log likelihood. The other two methods rely on beta distribution, but they differ in evaluation of the expectation of the log likelihood – the third one (FAIR-betaSF) uses score function and the fourth one (FAIR-betaREP) relies on reparametrization. Additionally, we discuss how to reduce the variance of FAIR-Bernoulli and FAIR-betaSF using baseline functions. We evaluated our models on four different real-world datasets and compared them to the state-of-the-art techniques. The results demonstrate that FAIR achieved the best results, with respect to fairness and classification performance. Furthermore, to the best of our knowledge, this is the first model that merges reweighting and adversarial approaches relying on a weighting function that can provide interpretable information about fairness of individual instances.

To summaryze, the contributions of this work are as follows:

- We merge reweighing and adversarial approaches for mitigating bias in machine learning models, while keeping the best from both.
- The proposed method can provide interpretable information about fairness of individual instances.
- We provide theoretical analysis of properties of adversarial re-weighting.
- We explore several variants of instance weight estimation including probabilistic ones.

- We evaluate the method on four different real-world datasets, compared to the state-of-the-art techniques, and provide qualitative analysis.

The remainder of the paper is structured as follows. In section 2 the related work is reviewed. The proposed FAIR algorithm with different variants is described in 3. Experimental setup and results on real-world applications are shown in sections 4 and 5, respectively. Final conclusions are given in section 6.

2. Related Work

Notion of fairness. In context of decision-making, (un)fairness has several distinct notions, one of the most prominent being *disparate impact* [28]. It represents a situation in which decisions ($\hat{\mathbf{y}}$) made by classifier are disproportional between instances with different values of sensitive attributes ($\mathbf{s}$). We use three measures of disparate impact. First metric used is *absolute statistical parity difference*:

$$\textbf{ASD} = |P(\hat{\mathbf{y}} = 1|\mathbf{s} = 0) - P(\hat{\mathbf{y}} = 1|\mathbf{s} = 1)| \tag{1}$$

Low values of **ASD** mean that both groups have approximately the same probability of being labeled 1 (e.g., bank loan granted) by the model. In such case, the classifier is said to have statistical parity. Second metric we used is *absolute equal opportunity difference*:

$$\textbf{AEOD} = |TPR_{\mathbf{s}=0} - TPR_{\mathbf{s}=1}| \tag{2}$$

where TPR represents true positive rate (recall) of the prediction model. Recall reflects opportunity, so this measure can be interpreted as a difference of opportunities between unprivileged and privileged group. Value of AEOD close to 0 is desirable. The third metric that we used is *average odds difference*. Average odds difference can be formulated as:

$$\textbf{AOD} = \frac{1}{2}(|FPR_{s=unpriv} - FPR_{s=priv}| + |TPR_{s=unpriv} - TPR_{s=priv}|) \tag{3}$$

where FPR represent false positive rate (probability of false alarm), and TPR true positive rate (recall). Values of **AOD** close to zero are preferred.

Regarding fairness-aware machine learning algorithms, interested readers are referred to an extensive reviews presented in [29] and [30]. A naive approach to ensuring fairness would be to eliminate sensitive features from the dataset. However, the information contained in the sensitive features can often be approximated from other input features. For example, location of residence correlates with race, although it is

not obviously sensitive itself. Therefore, more sophisticated approaches are needed. We focus on two families of methods relevant for our work.

Instance Reweighting. Instance reweighting has shown impressive results, although the idea is relatively simple [31, 32, 33, 34, 35]. As a preprocessing technique, it was traditionally used for the class imbalance problem by assigning larger weights to instances of a lower cardinality class, so that the learning algorithm gives more importance to that class. This idea can be applied to the fairness problem as well – it assigns lower importance to unfair examples or removes them from the learning process. More specifically, those examples will have a lower impact on the likelihood function one tries to optimize. The simplest approach is to assign weights to instances so that sums of weights per value of a sensitive feature is the same and all instances from a group have the same weight [20]. That approach was improved in [32] by utilizing adaptive sensitive reweighting procedure. One can use variational fair auto encoder with Maximum Mean Discrepancy [36] which calculates distances between distributions using kernels. It is worth noticing that instance reweighting has shown to have lower disparate impact [31] compared to not applying any instance weighting strategy. An advantage of this class of methods is that the weights can be interpreted as indicators of individual instance fairness. However, the training is not end-to-end. To apply some instance reweighting strategy one needs to perform a two step procedure – first to obtain instance weights, and then to use the weights by the learning algorithm. This is a drawback of this approach since the weighting procedure is oblivious of the model representation and learning algorithm and therefore might choose suboptimal weights for them.

Our approach considerably differs from these approaches in that the weight estimation is integrated in the one step end-to-end learning procedure. Also, in our method the weights are not plain scalars, but the outputs of a neural network. Hence, for purposes of interpretability, such weights can also be estimated for new instances which were not included in the training without retraining the whole system.

Adversarial training. Adversarial training provides a framework for mitigating biases by learning new data representation from which it is possible to predict the target variable, but not possible to predict the sensitive attribute. This approach creates a trade-off between two goal functions and therefore reaches Nash equilibrium [21]. Adversarial training for fairness was first presented in [37]. Similar model was applied to recidivism prediction in order to remove racial bias [24]. A theoretical analysis of solving fairness problem via adversarial approach is presented in [25]. An important approach of such kind is Fair Adversarial Discriminative model (FAD) [15]. Moreover, theoretical analysis of the relationship between the label classifier performance and the adversary's ability to predict the sensitive attribute value is

provided. Also, in the same paper, a variation of the adversarial learning procedure is developed to increase diversity among elements of each mini-batch of the gradient descent training, in order to achieve a representation that does not suffer from mode collapse. Similarly, Zhao et. al [38] presented a algorithm for Conditional Learning of Fair Representations (CLFR) that can simultaneously mitigate two notions of disparity among different subgroups in the classification problems. Another adversarial approach focuses on learning to select non-sensitive features on per instance basis [11]. Ragonesi et. al [39] proposed optimization strategy (LURMI - Learning Unbiased Representations via Mutual Information backpropagation), which simultaneously estimates and minimizes the mutual information between the learned representation and specific data attributes by using adversarial framework. Cotter et. al [40] introduced an interesting new approach (PYCO) for solving constrainted optimization problem by introducing Langrangian (PYCO_diff) and proxy-Lagrangian (PYCO_non_diff) based min-max optimization methods for solving differentiable and non-differentiable constraint optimization problems, respectively. It is demonstrated that PYCO can be successfully used for solving group fairness constraint optimization problems.

The adversarial approach in general is employed to minimize the correlation between selected features and sensitive information. While adversarial approach enables end-to-end training it does not provide any interpretable information on the individual fairness of instances. Our approach differs from these approaches in that it provides interpretable information on instance fairness like reweighting approaches do. That way it tries to keep the best from both worlds.

3. Fair Adversarial Instance Re-weighting - FAIR

The dataset given by $D = \{(\mathbf{x}_i, \mathbf{y}_i, \mathbf{s}_i)\}_{i=1}^{N}$, consists of input features $\mathbf{x}$, the true label (or the target variable) $\mathbf{y}$ and sensitive features $\mathbf{s}$. It is generated by joint true underlying distribution $D \sim P(\mathbf{x}, \mathbf{y}, \mathbf{s})$. Unfairness which AI models learn is introduced through data instances containing unfair decisions. Therefore, we strive to recognize if a particular instance in a dataset is unfair. The main principle of FAIR is to reweight log likelihood of each instance, according to the trade-off between fairness and prediction performance, in order to obtain a fair and useful predictor of the target variable.

FAIR consists of three neural networks: the weighting network $f_\theta(\mathbf{x})$, the predictor network $g_\phi(\mathbf{x})$, and the sensitive network $h_\psi(\mathbf{x})$. For an instance $\mathbf{x}$ the weighting network outputs the weight of that instance $w_\mathbf{x} \in [0, 1]$, while the predictor network and the sensitive network output predictions of the output labels $\mathbf{y}$ and the sensitive

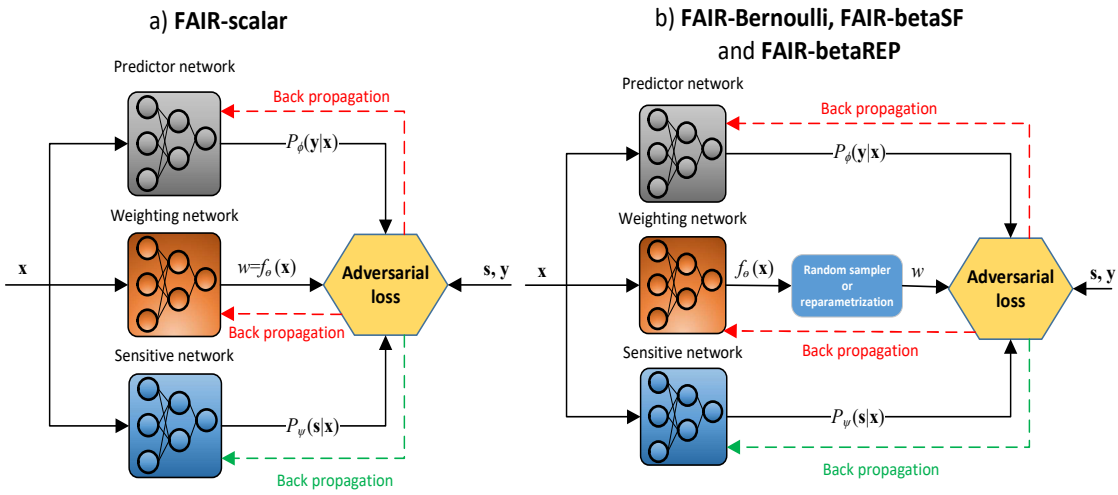

Figure 1: Graphical representations of FAIR with probabalistic and non-probabalistic frameworks

features $\mathbf{s}$, respectively. We denote probability functions modelled by these networks as $P_\phi(\mathbf{y}|\mathbf{x})$ and $P_\psi(\mathbf{s}|\mathbf{x})$. In order to incorporate the fairness objective, FAIR weights log likelihood of instances, so that the ones that are strongly informative of the sensitive features, but not of the target variable are assigned low weights and the ones that are informative of the target variable, but not of the sensitive attributes are assigned high weights. The weighting network is not used during inference, but can be helpful for assessing new instances.

Based on different weighting techniques, we present four different FAIR weighting methods. The first one, FAIR-scalar is based on non-probabilistic weighting framework, whereas FAIR-Bernoulli, FAIR-betaSF, and FAIR-betaREP are based on probabilistic framework. The graphical representation of FAIR with different weighting methods are given in Fig. 1.

3.1. FAIR – non-probablistic framework

Assume that each instance $\mathbf{x}$ is assigned a scalar weight $f_\theta(\mathbf{x}) \in [0, 1]$ by a weighting network. Then, FAIR-scalar adversarial problem is given by:

$$(\theta^*, \phi^*, \psi^*) = \arg\min_{\theta,\phi} \max_{\psi} \mathbb{E}_{\substack{\mathbf{x},\mathbf{y},\mathbf{s}\sim P(\mathbf{x},\mathbf{y},\mathbf{s}) \\ w=f_\theta(\mathbf{x})}} [w \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x}))] \qquad (4)$$

The hyperparameter α controls the trade-off between fairness and predictive performance of the predictor network, but this trade-off will be given further theoretical analysis.

3.2. FAIR – probabilistic framework

In the the case of FAIR with probabilistic approach to weighting, it is assumed that weights of instances are random variables. In contrast to FAIR-scalar, in probabilistic framework, output of the weighting network f_θ models a probability distribution of instance weights: $P(w_{\mathbf{x}}|\mathbf{x})$. Consequently, we can use different probability distribution models. Therefore, FAIR with probabilist approach introduced additional regularization through probability distribution of instance weights, and have more chance to reach mixed Nash equilibrium in the cases when pure Nash equilibrium does not exist. We consider Bernoulli (FAIR-Bernoulli) and beta distribution (FAIR-betaSF and FAIR-betaREP).

FAIR-Bernoulli assumes that log likelihoods of instances, with respect to sensitive features $\log P_\psi(\mathbf{s}|\mathbf{x})$ and labels $\log P_\phi(\mathbf{y}|\mathbf{x})$ are weighted by integers $w_{\mathbf{x}} \in \{0, 1\}$ such that it holds $P_\theta(w_{\mathbf{x}} = 1|\mathbf{x}) = f_\theta(x)$, meaning that the conditional probability of weights is a Bernoulli distribution $\mathcal{B}(f_\theta(\mathbf{x}))$. The FAIR-Bernoulli adversarial loss $\mathcal{L}_\alpha^{\mathcal{B}}(\theta, \phi, \psi)$ is given by:

$$\mathbb{E}_{\substack{\mathbf{x},\mathbf{y},\mathbf{s}\sim P(\mathbf{x},\mathbf{y},\mathbf{s}) \\ w\sim P_\theta(w|\mathbf{x})}} \Big[w \cdot (\alpha \log P_\psi \left(\mathbf{s}|\mathbf{x}\right) - \log P_\phi \left(\mathbf{y}|\mathbf{x}\right)) \Big] \tag{5}$$

and the corresponding adversarial problem is $(\theta^*, \phi^*, \psi^*) = \arg\min_{\theta,\phi} \max_\psi \mathcal{L}_\alpha^{\mathcal{B}}(\theta, \phi, \psi)$ where the superscript $\mathcal{B}$ emphasizes Bernoulli assumption.

In order to optimize the loss, gradients with respect to θ, ϕ, and ψ need to be computed. Gradients with respect to ϕ and ψ are computed by standard backpropagation. However, the gradient with respect to θ is trickier since θ defines the distribution of w over which the expectation is taken. Therefore, we derive the gradient of the adversarial loss $\nabla_\theta \mathcal{L}_\alpha(\theta, \phi, \psi)$ for FAIR-Bernoulli and FAIR-betaSF as follows:

$$\nabla_\theta \mathcal{L}_\alpha(\theta, \phi, \psi) = \nabla_\theta \mathbb{E}_{\substack{\mathbf{x},\mathbf{y},\mathbf{s}\sim P(\mathbf{x},\mathbf{y},\mathbf{s}) \\ w\sim P_\theta(w|\mathbf{x})}} \Big[w \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x})) \Big] \tag{6}$$

The gradient operator ∇_θ can be propagated through the expectation as:

$$\nabla_\theta \mathcal{L}_\alpha(\theta, \phi, \psi) = \mathbb{E}_{\mathbf{y},\mathbf{x},\mathbf{s}} \Big[\int_w \nabla_\theta P_\theta(w|\mathbf{x}) \cdot w \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi \left(\mathbf{y}|\mathbf{x}\right)) dw \Big] \tag{7}$$

Gradient of the distribution $P_\theta(w|\mathbf{x})$ can be transformed as:

$$\begin{aligned} \nabla_\theta P_\theta(w|\mathbf{x}) &= P_\theta(w|\mathbf{x}) \cdot \frac{\nabla_\theta P_\theta(w|\mathbf{x})}{P_\theta(w|\mathbf{x})} \\ &= P_\theta(w|\mathbf{x}) \cdot \nabla_\theta \log P_\theta(w|\mathbf{x}) \end{aligned} \tag{8}$$

Following this transformation, the final form of the gradient of the loss with respect to θ can be represented as:

$$\mathbb{E}_{\substack{\mathbf{x},\mathbf{s},\mathbf{y}\sim P(\mathbf{x},\mathbf{s},\mathbf{y}) \\ w\sim P_\theta(w|\mathbf{x})}}\Big[w \cdot \nabla_\theta \log P_\theta(w|\mathbf{x}) \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x}))\Big] \tag{9}$$

which is a suitable form as it allows the use of the stochastic gradient descent.

Next, we assume that weights $w_\mathbf{x}$ are random variables distributed according to the beta distribution which, in contrast to the case of FAIR-Bernoulli, takes any value from the interval $[0, 1]$. The outputs of the weighting network are the parameters $\alpha_\mathbf{x}$ and $\beta_\mathbf{x}$ of the beta distribution. The adversarial loss as defined by Eq. 5, but with beta distribution assumed instead of Bernoulli. We denote corresponding loss by $\mathcal{L}^\beta_\alpha(\theta, \phi, \psi)$ where β in the superscript emphasizes the assumed distribution. In optimization, the gradient $\nabla_\theta \mathcal{L}^\beta_\alpha(\theta, \phi, \psi)$ can be evaluated either by using score function as in Eq. 9 or by the reparametrization trick of beta distribution as shown in [41]. These two approaches we name FAIR-betaSF and FAIR-betaREP respectively.

Pseudocode of probabalistic FAIR with score function (FAIR-Bernoulli and FAIR-betaSF) is presented in Algortihm 1. FAIR losses are defined in terms of expectations. However, with finite samples, expectation is always approximated by sample mean, which we use in the algorithm. Incorporation of baseline functions for the reduction of variance of gradient estimate is discussed in Appendix A.

Algorithm 1 Probabilistic FAIR with score function

Input: learning rates $\gamma_\theta, \gamma_\phi, \gamma_\psi$, dataset D, hyperparameter α, probabilistic model $\mathcal{P}$ of instance weights, number of iterations M
Output: parameters θ, ϕ, ψ

Initialize θ, ϕ, ψ
for i = 1 to M **do**
  Sample a mini-batch $B \subseteq D$
  Sample $w_\mathbf{x} \sim \mathcal{P}(f_\theta(\mathbf{x}))$ for each $\mathbf{x}$ in B
  $d_\theta \leftarrow \gamma_\theta \frac{1}{|B|} \sum_{(\mathbf{x},\mathbf{y},\mathbf{s})\in B} \big[w_\mathbf{x} \nabla_\theta \log P_\theta(w_\mathbf{x}|\mathbf{x}) \cdot$
    $(\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x}))\big]$
  $d_\phi \leftarrow \gamma_\phi \nabla_\phi \mathcal{L}^\mathcal{P}_\alpha(\theta, \phi, \psi, B)$
  $d_\psi \leftarrow -\gamma_\psi \nabla_\psi \mathcal{L}^\mathcal{P}_\alpha(\theta, \phi, \psi, B)$
  $(\theta, \phi, \psi) \leftarrow (\theta, \phi, \psi) - (d_\theta, d_\phi, d_\psi)$

3.3. Theoretical analysis of model properties

In order to analyze properties of all our models in a uniform manner, we discuss instance weights as real values in the interval $[0, 1]$ and we emphasize dependence

of the weight on the instance as $w_{\mathbf{x}}$ without explicating specifics of the dependence. Vector of all such weights is denoted $\mathbf{w}$ and it is denoted $\mathbf{w}^*$ if it is a part of the optimal solution of the corresponding adversarial problem. In practice, expectations are approximated by sample means (or sums since outmost constant factors are irrelevant in optimization), and losses are regularized. Therefore we consider a regularized loss $\mathcal{L}_\alpha(\mathbf{w}, \phi, \psi)$:

$$\begin{aligned} \sum_{(\mathbf{x},\mathbf{y},\mathbf{s})\in D} & w_{\mathbf{x}}[\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x})] \\ & \text{s.t. } \|\theta\|_2^2 + \|\phi\|_2^2 + \|\psi\|_2^2 \leq \lambda \end{aligned} \tag{10}$$

where dependence of $w_{\mathbf{x}}$ on θ is not made explicit, but we are aware that it exists. To shorten the proofs, we formulate regularization in a constraint based manner [42], although it is more often formulated and implemented in a mathematically equivalent penalty based manner (note that the meaning of regularization parameter is reversed – in penalty based formulation case $\lambda = 0$ corresponds to an infinite value of λ in constraint based formulation).

Now we theoretically analyze the behaviour of our method. The focus of the analysis is on the following elements:

- the instance weights,
- the predictive performance of predictive and sensitive networks for a specific instance as measured by $\log P_\phi(\mathbf{y}|\mathbf{x})$ and $\log P_\psi(\mathbf{s}|\mathbf{x})$, which are related to instance predictive quality and its fairness,
- the hyperparameter α,

and their interdependence. The interdependence is most easily observed with respect to variation of the hyperparameter α, which is under direct control of the user. However, the interaction of all elements is relevant for the understanding of the method. First, we focus on how the variation of the hyperparamter reflects on the trade-off between fairness and the quality of prediction of the target variable. In a nutshell, extreme case $\alpha = 0$ represents extreme emphasis on fairness and $\alpha \to \infty$ represents extreme emphasis on quality of prediction and disregard for fairness. Please note that a superficial glance at the adversarial problem would suggest vice versa, but we stress that it is not the case. Second, we aim to understand how the hyperparameter α affects the optimal weights assigned to the instances. It turns out that under some (reasonable) conditions the optimal weights will tend to 0 and 1 and that the value of α controls the proportion of the two limiting values.

The effect of α on the weights and thus on model behvaiour is mediated by the ratio of performance of predictive and sensitive networks. These elements of the formal analysis also provide better intuitive understanding of the method. We provide such discussion after the theoretical results.

Lemma 1. *If λ is finite, there exist strictly negative constants c_ϕ, c'_ϕ, c_ψ, and c'_ψ such that it holds $c_\phi \leq \log P_\phi(\mathbf{y}|\mathbf{x}) \leq c'_\phi$ and $c_\psi \leq \log P_\phi(\mathbf{s}|\mathbf{x}) \leq c'_\psi$ for any $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{s}$, and any ϕ and ψ which satisfy regularization condition 10.*

Proof. Denote $\mathcal{B}$ the ball defined by $\|\theta\|_2^2 + \|\phi\|_2^2 + \|\psi\|_2^2 \leq \lambda$, representing the set of feasible solutions of the optimization problem. Denote $\bar{g}_\phi(\mathbf{x})$ the network $g_\phi(\mathbf{x})$ modelling $\mathbf{y}$ with sigmoid function at the output removed and $\bar{h}_\psi(\mathbf{x})$ the network $h_\psi(\mathbf{x})$ modelling $\mathbf{s}$ with sigmoid at the output removed. Since $\mathcal{B}$ is a compact set and $\bar{g}_\phi(\mathbf{x})$ and $\bar{h}_\psi(\mathbf{x})$ are continuous functions, they both attain their finite minimal and maximal values within $\mathcal{B}$. Since $\log P_\psi(\mathbf{s}|\mathbf{x})$ and $\log P_\phi(\mathbf{y}|\mathbf{x})$ are continuous functions of $\bar{h}_\psi(\mathbf{x})$ and $\bar{g}_\phi(\mathbf{x})$, respectively, which map the range of $\bar{h}_\psi$ and $\bar{g}_\phi$ from $(-\infty, \infty)$ to $(-\infty, 0)$, functions $\log P_\psi(\mathbf{s}|\mathbf{x})$ and $\log P_\phi(\mathbf{y}|\mathbf{x})$ attain their strictly negative and finite minimal and maximal values within $\mathcal{B}$. Therefore, the required constants exist, by which the lemma is proven. □

Theorem 1. *If λ is finite, for $\alpha = 0$ it holds $\mathbf{w}^* = \mathbf{0}$.*

Proof. By Lemma 1, $P_\psi(\mathbf{s}|\mathbf{x})$ is bounded, so for $\alpha = 0$ it holds:

$$\begin{aligned}(\mathbf{w}^*, \phi^*, \psi^*) &= \arg\min_{\mathbf{w},\phi} \max_{\psi} \mathcal{L}_\alpha(\mathbf{w}, \phi, \psi) \\ &= \arg\min_{\mathbf{w},\phi} - \sum_{(\mathbf{x},\mathbf{y},\mathbf{s})\in D} w_{\mathbf{x}} \cdot \log P_\phi(\mathbf{y}|\mathbf{x})\end{aligned} \tag{11}$$

By Lema 1, $\log P_\phi(\mathbf{y}|\mathbf{x})$ is strictly negative, so $-\sum_{(\mathbf{x},\mathbf{y},\mathbf{s})\in D} w_{\mathbf{x}} \cdot \log P_\phi(\mathbf{y}|\mathbf{x})$ is zero or positive. Therefore its minimal value is 0 for $\mathbf{w} = \mathbf{0}$ regardless of ϕ. Therefore, it holds $\mathbf{w}^* = \mathbf{0}$. □

Theorem 2. *For each instance $(\mathbf{x}, \mathbf{y}, \mathbf{s})$, it holds $w^*_{\mathbf{x}} = 1$ or $w^*_{\mathbf{x}} = 0$ or $\alpha \log P_{\psi^*}(\mathbf{s}|\mathbf{x}) = \log P_{\phi^*}(\mathbf{y}|\mathbf{x})$.*

Proof. Consider a partial derivative in the optimal solution:

$$\frac{\partial \mathcal{L}_\alpha}{\partial w_{\mathbf{x}}}(\mathbf{w}^*, \phi^*, \psi^*) = \alpha \log P_{\psi^*}(\mathbf{s}|\mathbf{x}) - \log P_{\phi^*}(\mathbf{y}|\mathbf{x}) \tag{12}$$

If the derivative is negative, then there exists $d > 0$ such that it holds

$$\mathcal{L}_\alpha(\mathbf{w}^* + d\mathbf{e}_{\mathbf{x}}, \phi^*, \psi^*) < \mathcal{L}_\alpha(\mathbf{w}^*, \phi^*, \psi^*) \tag{13}$$

where $\mathbf{e}_{\mathbf{x}} = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^{|D|}$ where 1 is at the coordinate corresponding to $w_{\mathbf{x}}$. Therefore, if it holds $w^*_{\mathbf{x}} < 1$, $w^*_{\mathbf{x}}$ can be increased in order to decrease the loss and $(\mathbf{w}^*, \phi^*, \psi^*)$ is not an optimal solution, which is a contradiction. Therefore, it has to hold $w^*_{\mathbf{x}} = 1$. If the derivative is positive, $w^*_{\mathbf{x}} = 0$ is proven in an analogous manner. If the derivative is 0, the theorem holds due to its third case. □

In the following propositions, we explicitly denote dependence of the optimal solution on α.

Lemma 2. *If λ is finite, for each instance $(\mathbf{x}, \mathbf{y}, \mathbf{s}) \in D$ it holds*

$$\frac{\partial \mathcal{L}_\alpha}{\partial w_{\mathbf{x}}}(\mathbf{w}^*_\alpha, \phi^*_\alpha, \psi^*_\alpha) \to -\infty \quad \textit{as} \quad \alpha \to \infty \tag{14}$$

Proof. Consider a partial derivative with respect to $w_{\mathbf{x}}$ in an optimum:

$$\frac{\partial \mathcal{L}_\alpha}{\partial w_{\mathbf{x}}}(\mathbf{w}^*_\alpha, \phi^*_\alpha, \psi^*_\alpha) = \alpha \log P_{\psi^*_\alpha}(\mathbf{s}|\mathbf{x}) - \log P_{\phi^*_\alpha}(\mathbf{y}|\mathbf{x}) \tag{15}$$

According to Lemma 1, for any feasible ψ and ϕ there exists constants $c'_\psi < 0$ and c_ϕ such that it holds $\log P_\psi(\mathbf{s}|\mathbf{x}) \leq c'_\psi$ and $\log P_\phi(\mathbf{y}|\mathbf{x}) \geq c_\phi$. Therefore, the first term goes to $-\infty$ as $\alpha \to \infty$ and the second term is bounded, so the limit of the partial derivative is $-\infty$. □

Theorem 3. *If λ is finite, for each instance $(\mathbf{x}, \mathbf{y}, \mathbf{s}) \in D$, it holds $w^*_{\mathbf{x},\alpha} \to 1$ as $\alpha \to \infty$.*

Proof. According to Lemma 2, the limit of the values of the partial derivative $\frac{\partial \mathcal{L}_\alpha}{\partial w_{\mathbf{x}}}(\mathbf{w}^*_\alpha, \phi^*_\alpha, \psi^*_\alpha)$ in optima as $\alpha \to \infty$ is negative. Then, by the definition of the limit, there exists $\alpha_0 \in \mathbb{R}$ such that for all $\alpha > \alpha_0$ it holds:

$$\frac{\partial \mathcal{L}_\alpha}{\partial w_{\mathbf{x}}}(\mathbf{w}^*_\alpha, \phi^*_\alpha, \psi^*_\alpha) < 0 \tag{16}$$

For each such α, since derivative with respect to $w_{\mathbf{x}}$ is negative, by the same argument as in the proof of Theorem 2, it holds $w^*_{\mathbf{x},\alpha} = 1$. Hence, we can conclude that for each $\varepsilon > 0$, there exists α_0 such that for all $\alpha > \alpha_0$ it holds $w^*_{\mathbf{x},\alpha} > 1 - \varepsilon$ (since $w^*_{\mathbf{x},\alpha} = 1$). Therefore, by the definition of the limit, we conclude that it holds $w^*_{\mathbf{x},\alpha} \to 1$ as $\alpha \to \infty$. □

3.4. Intuitive discussion of theoretical properties

Provided theorems explain the way the crutial elements of the model interact when set in motion by varying the hyperparameter α. It turns out that the hyperparameter can be understood as a threshold on the ratio of instance "predictiveness" and instance fairness based on which the model decides if the instance should be discarded or exploited in learning. If it holds:

$$\frac{\log P_\phi(\mathbf{y}|\mathbf{x})}{\log P_\psi(\mathbf{s}|\mathbf{x})} < \alpha \tag{17}$$

intuitively, the instance is fair enough considering its "predictiveness" (with respect to the target variable). Namely, for the ratio to be low, its "predictiveness" should be high (reflected by small negative value of log likelihood in the numerator) and its unfairness should be low (reflected by the large negative value of log likelihood in the denominator). In the extreme case of $\alpha = 0$ no instance is considered fair enough, since neither the log likelihood in the numerator can be exactly zero, nor the log likelihood in the denominator can be infinite. According to Theorem 1, in that case, all instances are discarded. In the other extreme, according to Theorem 3, as α tends to infinity, fairness is disregarded and all instances are used for learning. For values of α in between some instances are disregarded and some are used. Note that this insight is not only about the hyperparameter α, but also about the way the model judges predictive qualities of the instance and their trade-off and how, based on that, it weights them differently.

Besides this central insight, other aspects of the tehoretical analysis merit a comment. In case of infinite λ, overfitting might falsify our proof of Lemma 1 and in that case for some instance $\mathbf{x}$ it might hold $w^*_{\mathbf{x},\alpha} \to 0$ as $\alpha \to \infty$. However, this suggests an interesting diagnostic property – if for ever larger values of α one obtains $w_\mathbf{x} = 0$ for some $\mathbf{x}$, one has reasons to suspect overfitting. Also, finite capacity of the network might make regularization unnecessary in practice. However, theoretical analysis was easier under the assumption of explicit regularization.

Also note that the model of instance weights does not need allow values 0 and 1. Nevertheless, the provided theorems inform us that the gradients will push the weights towards these values. Still, our probabilistic approaches might provide additional regularization by giving nonzero probability to other weight values except the optimal ones.

4. Experimental Setup

Datasets. The proposed framework was tested on four datasets, three of which are commonly used benchmarks. Two datasets (German credit and Adult income)

come from the UCI ML repository [43]. To our knowledge the Hospital readmission dataset was used in this paper for the first time in the context of fairness.

The first, the *Adult income* dataset [44] represents a binary classification task of predicting whether an income is greater than 50K dollars. The dataset contains 45,222 instances described by 14 features and including the sensitive attribute Gender. The atributes used in the dataset describes the individual's education level, age, gender, occupation, workclass, martial-status, relationship, capital loss and etc [45]. After applying dummy coding, total number of features was 93. Total numbers of instances used in training, validation and testing are 31,655, 6,783, and 6,784, respectively.

Second dataset we used is the *Hospital readmission* dataset [46]. It represents a binary classification task where label 1 means that patient is readmitted within 30 days. The dataset consists of 66,994 instances and 931 attributes, including sensitive attribute Gender. Total number of instances used in training, validation and testing are 46,895, 10,049 and 10,050, respectively.

The third dataset, named *Hospital Expenditures*, comes from [47]. It represents a binary classification task of predicting whether a person would have high or low utilization of medical expenditures. The sensitive attribute is Race. Dataset contains 15,830 instances and 133 attributes, after dummy codding, total number of attributes used in this dataset was 138. For training, validation, and testing, we used 11,081, 2,374 and 2,375 instances respectively.

As a fourth dataset, we used *German credit* dataset. *German credit* dataset has 1,000 instances where the task is to classify bank account holders into classes good or bad. The total number of attributes used in the dataset, after applying dummy coding is 58, including sensitive attributes. Following the definition of fairness from [12] for German credit dataset, there are two sensitive attributes, one being Gender and other being Age (≥ 25 is considered as privileged class, and < 25 as unprivileged class). Total numbers of instances used in training, validation and testing was 700, 150, and 150, respectively.

Models. The results obtained by FAIR models are compared with ten related and state-of-the-art algorithms: FAD, reweighing preprocessing technique from [20] combined with the random forest classifier (Reweighing - RF) and with neural networks (Reweighing - NN), disparity impact remover [31] combined with random forest (DI - RF) and neural networks (DI - NN), prejudice remover [17] (PR), models based on conditonal learning of fair representations [38] (CLFR), learning unbiased representation via mutual information [39] (LURMI), optimization model for differentiable and non-differentiable constraints [40] (PYCO_diff and PYCO_non_diff). In the case of PYCO_non_diff model, all fairness metrics were directly optimized, whereas in

the case of PYCO_diff, approximation of fairness constraints is used. Architecture, number of epochs in early stopping procedure, and learning rates were empirically determined as to optimize the performance of each model, by varying design choices of the architectures described in the literature. Detailed specifications can be found in Appendix B. We did not use explicit regularization in our experiments since the capacity of the models can also be controlled through the choice of architecture and early stopping.

Optimization. For optimization of all neural network based models we use Adam optimizer [48]. During optimization of FAIR and FAD models, early stopping was used. In the early stopping procedure, the min and max objectives of adversarial training procedure on validation set were monitored. In case when there were no improvements in either of these two metrics for a given number of epoch (provided in Appendix B), the training procedure is stopped.

Metrics. Classification performance of all presented classifiers is quantified by the accuracy (ACC), which is calculated for the target variable ($\mathbf{y}$) and the sensitive attribute ($\mathbf{s}$). Therefore, we present $\text{ACC}_{\mathbf{y}}$ and $\text{ACC}_{\mathbf{s}}$ for the target variable and the sensitive attribute, respectively. If subscript is omitted, then $\text{ACC}_{\mathbf{y}}$ is presented. As fairness metrics we use ASD, AEOD, and AOD defined by Eqs. 1, 2, and 3, respectively.

Evaluation procedure and presentation of results. The evaluated models (both FAIR and the baselines) have hyperparameters which affect the trade-off between fairness and predictive performance of the classifiers. Note that such hyperparameters do not control model capacity. Therefore, we do not tune them to obtain maximal perfomance (like one might tune regularization hyperparameters). Instead, we vary them in order to illustrate model behaviour for different trade-offs. The hyperparameters α of FAIR, CLFR, LURMI and FAD models were varied in range $[0, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3]$, whereas in the case of other models, hyperparameters were varied in range $[0, 10^{-3}, 10^{-2}, 10^{-1}, 1]$ (since 1 is the maximal value for these methods). For each such value, evlauation metrics were computed. PYCO_diff and PYCO_non_diff models were optimized for each fairness metric separately. We call a set of models obtained from one model kind (FAD, FAIR, etc.) by varying the fairnes related hyperparamter, a *model family*. For instance, all FAD models trained for different values of α constitute a FAD model family.

Since the models are evaluated by two criteria (predictive performance of the target variable and fairness), one model can be better than the other according to one criterion and vice-versa. Since both critera are important, instead of privileging one of them we present our results in terms of Pareto fronts. For a set of trained models, Pareto front consists of models which are not dominated by any other models in

terms of both predictive performance and fairness [49]. Models which are dominated by others in terms of both criteria are obviously irrelevant and should be discarded. Pareto front can be plotted in 2D in terms of metrics for the two criteria used and visually inspected. We base our evaluation on the overall Pareto front which is a Pareto front of all trained models (union of all model familites). Models which yield more points in such Pareto front are better. Since FAIR and baseline models do not directly optimize any of the commonly used fairness metrics, we evaluate them using several different fairness metrics. Hence, we present the overall Pareto front of all trained models with respect to AOD, ASD, and AEOD as fairness metrics and ACC as the performance metric.

Construction of the Pareto front includes model selection - models are compared according to their performance and some of them are selected. Since evaluation metrics should never be reported on the data on which the selection was performed, we take care to train all models on the training set, to perform selection of the models for the Pareto front on the validation set, and to evaluate selected models on the test set. All results reported in the following section are calculated on the test set.

5. Results and Discussion

In this section we provide experimental results following the above described setup. Further on, we provide the discussion of these results and the qualitative evaluation of the behaviour of our model.

5.1. Results

In this section we present results obtained using the experimental evaluation outlined above.

Firstly, model performances obtained on the *Adult income* dataset are illustrated in Fig. 2 by three fairness metrics (AOD, ASD or AEOD) and classification performance ($ACC_{\mathbf{y}}$). The models with greater ACC score and lower (un)fairness metric (upper left corner of plots) are preferred. It can be observed that FAIR models dominates Pareto optimal solutions with respect to the all fairness metrics. In addition, FAIR-beta and LURMI models dominate the upper left corner of Pareto fronts for AOD and AEOD, whereas the FAIR-scalar and PYCO_non_diff dominate the upper left corner of Pareto front for ASD metric. Moreover, in Table 1, the Pareto optimal solutions obtained for all three fairness metrics and ($ACC_{\mathbf{y}}$) are presented. Similarly, it can be concluded that the number of FAIR models is larger compared to the other models and FAIR can therefore be considered better then other models.

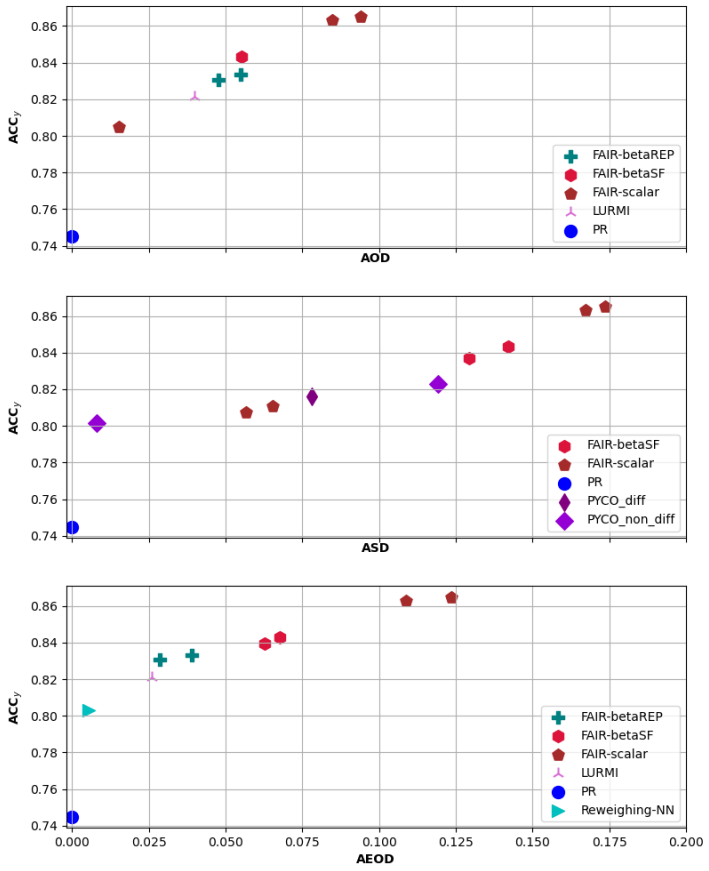

Figure 2: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *Adult income* datasets

Table 1: Pareto optimal solutions - *Adult income* dataset

model	ACC	AOD	ASD	AEOD
PR	0.7450	0.0000	0.0000	0.0000
Reweighing-NN	0.7927	0.0182	0.1517	0.0222
Reweighing-NN	0.8031	0.0200	0.1231	0.0052
Reweighing-NN	0.7963	0.0396	0.0918	0.1170
FAIR-scalar	0.8622	0.1151	0.0922	0.1544
FAIR-scalar	0.8050	0.0151	0.1105	0.0640
FAIR-scalar	0.8649	0.0939	0.1736	0.1236
FAIR-scalar	0.8630	0.0848	0.1672	0.1087
FAIR-scalar	0.7559	0.0090	0.0164	0.0161
FAIR-scalar	0.8110	0.0996	0.0654	0.1955
FAIR-scalar	0.8076	0.1466	0.0565	0.2841
FAIR-betaSF	0.8397	0.0607	0.1456	0.0627
FAIR-betaSF	0.8431	0.0551	0.1421	0.0676
FAIR-betaREP	0.8333	0.0550	0.1741	0.0390
FAIR-betaREP	0.8308	0.0475	0.1588	0.0287
LURMI	0.8212	0.0397	0.1420	0.0259
LURMI	0.7577	0.0157	0.0112	0.0286
LURMI	0.7653	0.0491	0.0860	0.0742
PYCO_diff	0.8068	0.0984	0.0596	0.1917
PYCO_diff	0.7810	0.0915	0.0233	0.1785
PYCO_diff	0.8135	0.0824	0.1000	0.1490
PYCO_diff	0.8162	0.1206	0.0780	0.2349
PYCO_diff	0.8272	0.0925	0.1408	0.1487
PYCO_non_diff	0.7802	0.5744	0.0097	0.3087
PYCO_non_diff	0.7974	0.0828	0.0626	0.1496
PYCO_non_diff	0.8016	0.2105	0.0081	0.3506
PYCO_non_diff	0.8138	0.0707	0.1084	0.1199
PYCO_non_diff	0.8231	0.0836	0.1192	0.1408

Table 2: Pareto optimal solutions - *Hospital readmission* dataset

model	ACC	AOD	ASD	AEOD
FAIR-scalar	0.8559	0.0001	0.0000	0.0002
FAIR-betaSF	0.8654	0.0002	0.0008	0.0008
FAIR-Bernoulli	0.8550	0.0000	0.0000	0.0000

Secondly, the results obtained on the *Hospital readmission* dataset are presented in Fig. 3. It can be noticed that only FAIR models exist on Pareto front and consequently all other models are dominated by them. It can be observed that in the case of all three fairness metrics FAIR-betaSF is the closest to the upper left corner and can therefore be considered better than others. The letter can be also confirmed in the Table 2 where only FAIR models exist on overall Pareto front.

Thirdly, models performances obtained on the *Hospital expenditures* dataset are illustrated in Fig. 4. It can be observed that the FAIR models dominate Pareto front in all presented metrics. In the case of ASD metric FAIR-scalar and PYCO models are the closest to the upper left corner of Pareto front, whereas in the case of AOD metric FAIR-betaSF is the best one and in the case of AEOD metrics it is the PYCO_non_diff model. Table 3 where overall Pareto front is presented, the FAIR models still dominate Pareto front.

Eventually, model performances obtained of *German credit* dataset for age and sex as sensitive attributes are presented in Figs. 6 and 5, respectively. Similarly as in previous datasets, overall Pareto fronts, for age and sex as sensitive attributes, are presented in Tables 4 and 5. It can be observed that in Fig. 5 all models are equally represented, whereas in Fig. 6 FAIR models dominate the Pareto front in all cases. Similar conclusion can be made in the case of overall Pareto fronts that are presented in Tables 4 and 5.

Additional results can be found in Appendix C.

5.2. Discussion and qualitative study

To summarize, FAIR variants often perform better than other methods, but of course the other methods can outperform them. Therefore we conclude that FAIR is roughly equal or better to the state-of-the-art methods. Most notable baselines seem to be PYCO methods. Among the FAIR variants, FAIR-beta seems to be most promising in terms of discussed accuracy and fairness metrics. A relevant question regarding FAIR models is which variant shows most promise and which one should be used in practice. The reason for introduction of probabilistic variants was the guaranteed existence of mixed Nash equilibrium, which is important since the

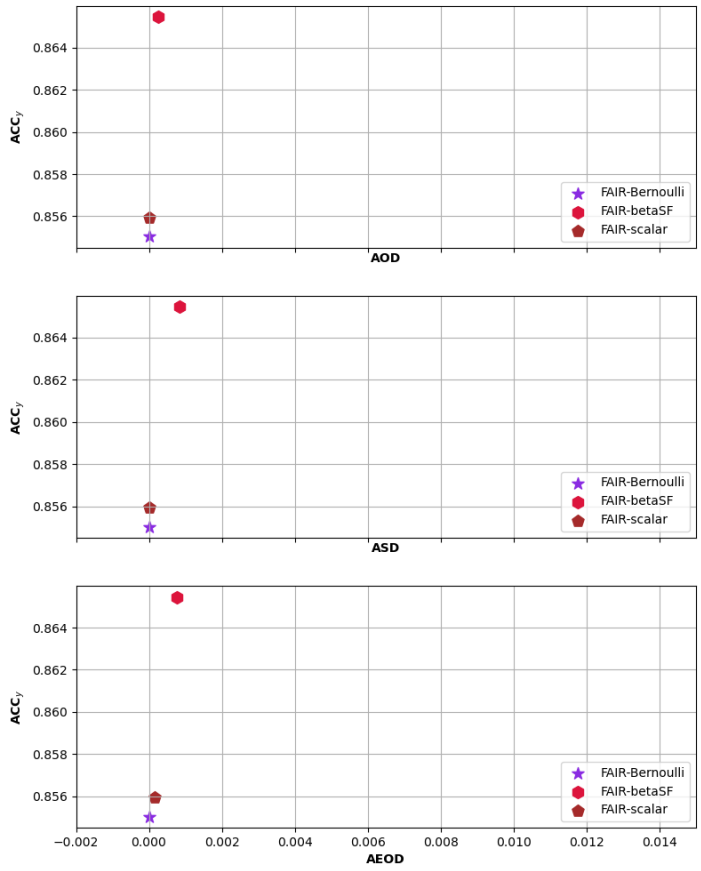

Figure 3: Classification performance and fairness of models as measured by $\text{ACC}_{\mathbf{y}}$ and AOD, ASD, or AEOD on the *Hospital readmission* dataset

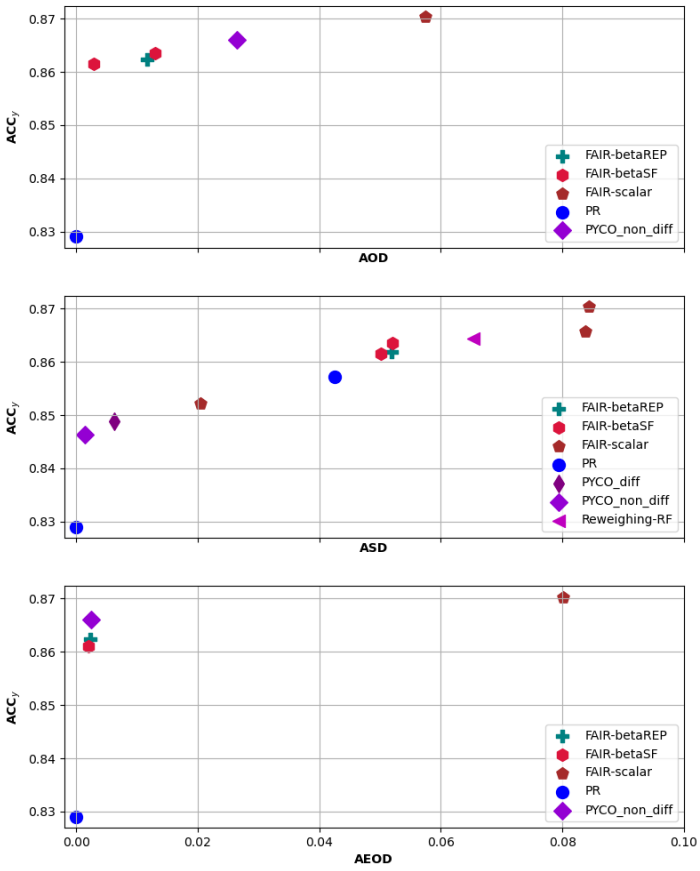

Figure 4: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *Hospital expenditures* dataset

Table 3: Pareto optimal solutions - *Hospital expenditures* dataset

model	ACC	AOD	ASD	AEOD
PR	0.8573	0.0334	0.0426	0.0589
PR	0.8291	0.0000	0.0000	0.0000
DI-NN	0.8459	0.0125	0.0324	0.0533
Reweighing-NN	0.8371	0.0107	0.0351	0.0078
Reweighing-RF	0.8644	0.0399	0.0653	0.0566
FAIR-scalar	0.8703	0.0574	0.0844	0.0801
FAIR-scalar	0.8657	0.0562	0.0837	0.0755
FAIR-scalar	0.8522	0.0276	0.0205	0.0512
FAIR-scalar	0.8484	0.0426	0.0099	0.0816
FAIR-betaSF	0.8615	0.0030	0.0502	0.0127
FAIR-betaSF	0.8636	0.0130	0.0520	0.0075
FAIR-betaSF	0.8611	0.0115	0.0545	0.0021
FAIR-Bernoulli	0.8648	0.0560	0.0866	0.0705
FAIR-betaREP	0.8623	0.0116	0.0534	0.0024
FAIR-betaREP	0.8619	0.0067	0.0519	0.0076
FAIR-betaREP	0.8581	0.0110	0.0556	0.0027
FAIR-betaREP	0.8564	0.0099	0.0575	0.0035
PYCO_diff	0.8488	0.0845	0.0063	0.1556
PYCO_non_diff	0.8319	0.0446	0.0160	0.0785
PYCO_non_diff	0.8463	0.0541	0.0014	0.0983
PYCO_non_diff	0.8661	0.0265	0.1038	0.0025
PYCO_non_diff	0.8640	0.0335	0.0839	0.0344

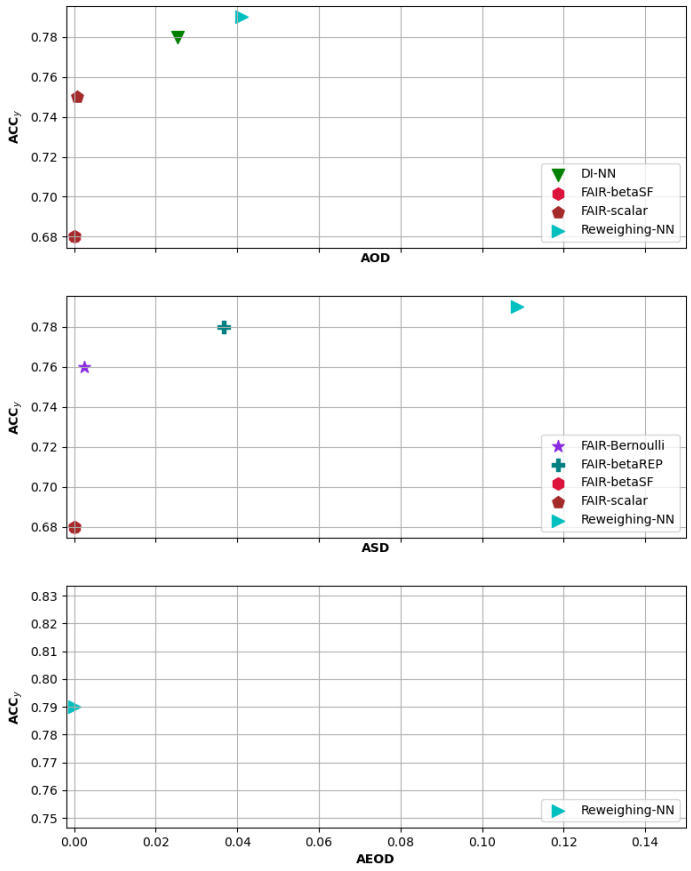

Figure 5: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *German credit* (age) dataset

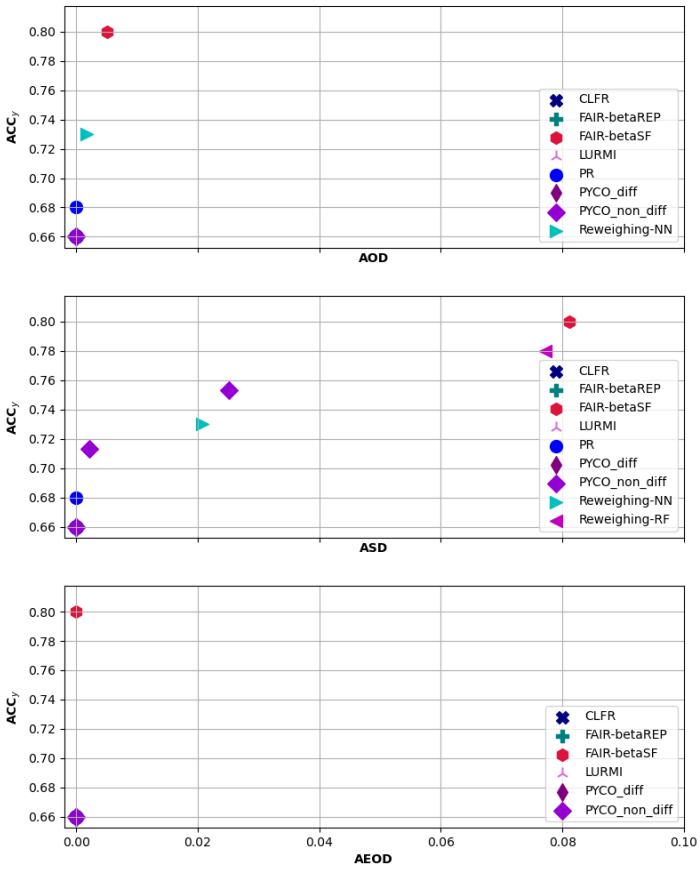

Figure 6: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *German credit* (sex) dataset

Table 4: Pareto optimal solutions - *German credit* - age dataset

model	ACC	AOD	ASD	AEOD
DI-NN	0.7800	0.0254	0.0537	0.1000
DI-NN	0.7700	0.0421	0.0427	0.1333
DI-NN	0.7700	0.0471	0.0134	0.0333
DI-RF	0.6900	0.0167	0.0110	0.0333
DI-RF	0.6900	0.0167	0.0110	0.0333
DI-RF	0.6900	0.0167	0.0110	0.0333
Reweighing-NN	0.7900	0.0410	0.1087	0.0000
Reweighing-RF	0.7000	0.0333	0.0220	0.0667
FAIR-scalar	0.6800	0.0000	0.0000	0.0000
FAIR-scalar	0.7500	0.0008	0.0867	0.1000
FAIR-scalar	0.7800	0.0852	0.1026	0.0333
FAIR-betaSF	0.6800	0.0000	0.0000	0.0000
FAIR-Bernoulli	0.7600	0.2260	0.0024	0.5667
FAIR-betaREP	0.6900	0.0167	0.0110	0.0333
FAIR-betaREP	0.7800	0.1977	0.0366	0.4000
PYCO_diff	0.7400	0.0224	0.0410	0.0278
PYCO_diff	0.7067	3.4655	0.0117	0.2444
PYCO_diff	0.7067	0.0251	0.0269	0.0211

Table 5: Pareto optimal solutions - *German credit* - sex dataset

model	ACC	AOD	ASD	AEOD
PR	0.6800	0.0000	0.0000	0.0000
DI-RF	0.7000	0.0167	0.0233	0.0333
DI-RF	0.7000	0.0178	0.0041	0.0167
Reweighing-NN	0.7300	0.0017	0.0208	0.1167
Reweighing-RF	0.7000	0.0167	0.0233	0.0333
Reweighing-RF	0.7800	0.0184	0.0771	0.0833
Reweighing-RF	0.7500	0.0289	0.0715	0.0667
FAD	0.7100	0.0154	0.0046	0.1667
FAIR-scalar	0.7600	0.0300	0.0619	0.1167
FAIR-scalar	0.7467	0.3124	0.0114	0.4643
FAIR-betaSF	0.6800	0.0000	0.0000	0.0000
FAIR-betaSF	0.8000	0.0050	0.0812	0.0000
FAIR-betaREP	0.7000	0.0101	0.0112	0.1333
FAIR-betaREP	0.6900	0.0006	0.0025	0.1333
FAIR-betaREP	0.8100	0.2228	0.2704	0.3500
CLFR	0.6600	0.0000	0.0000	0.0000
CLFR	0.6600	0.0000	0.0000	0.0000
CLFR	0.6600	0.0000	0.0000	0.0000
CLFR	0.6600	0.0000	0.0000	0.0000
LURMI	0.7000	0.0151	0.0210	0.0960
LURMI	0.6600	0.0000	0.0000	0.0000
LURMI	0.7000	0.0273	0.0094	0.0152
PYCO_diff	0.6600	0.0000	0.0000	0.0000
PYCO_non_diff	0.6600	0.0000	0.0000	0.0000
PYCO_non_diff	0.6600	0.0000	0.0000	0.0000
PYCO_non_diff	0.7133	0.0334	0.0022	0.0502
PYCO_non_diff	0.7533	0.1361	0.0251	0.2083

existence of pure Nash equilibrium is not guaranteed for $\min - \max$ problems and therefore FAIR-scalar might not converge to a mathematicaly meaningful solution. The fact that FAIR-beta achieves somehwaht better performance might be casued by such issue. On the other hand, the the advantage of FAIR-beta is not drastic and the simplicity of FAIR-scalar might make it a more appealing approach in practice if one can tollerate small drops in performance metrics.

Model behaviour of FAIR model with respect to change of hyperparameter α is shown in Fig. 7 on the *Geman credit* dataset. It can be observed that as α decreases, instances which are unfair (but potentially useful for prediction of target variable) are being discarded, so ACC metrics for both the target variable and sensitive attribute decrease. This is experimental verification of theoretical model properties presented in section 3.

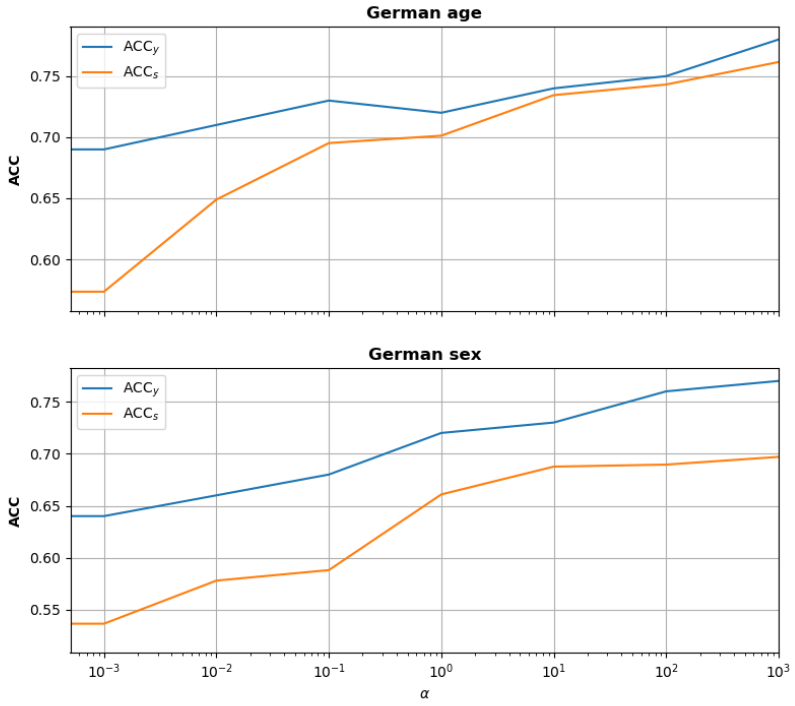

Figure 7: $\mathbf{ACC_y}$ and $\mathbf{ACC_s}$ as functions of the fariness hyperparameter α measured on the German credit - sex and Readmission datasets ($\mathbf{ACC_y}$ is preferred larger, and $\mathbf{ACC_s}$ smaller)

Furthermore, we increased the hyperparameter α in FAIR-scalar model from 0 to

Table 6: German credit dataset instances with non zero weights

Credit duration	48	36	36
Credit amount	18424	14318	15857
Investment as income percentage	1	4	2
Residence since	2	2	3
No.of credits taken	1	1	1
No. of people liable to provide maintenance for	1	1	1
Status of checking account	<200 DM	<200 DM	<0DM
Credit history	no credits taken	existing credits paid back duly till now	existing credits paid back duly till now
Purpose	other	car (new)	other
Savings	<100 DM	<100 DM	<100 DM
Employment	1<4 years	>=7 years	unemployed
Other debtors	none	none	co-applicant
Properties	Life insurance	unknown	car or other
Installment plans	bank	none	none
Housing	own	for free	own
Skill level	management	management	self-employed
Telphone	yes, under customer name	yes, under customer name	none
Foreign worker	no	no	yes
Sensitive attribute - Sex	male	female	male
Credit score -Label	Good	Good	Bad

the first value where one of the instance in training dataset has weight that tends to 1. Based on theoretical formulation of model properties this is the most fair instance in dataset. Moreover, we kept to increase parameter α until the first two instances with weights that tends to 1 in opposite sex and label categories occurred. In Table 6 the attributes of previously mentioned instances are presented.

Firstly, it could be observed that the most fair instance has good credit score mainly based on facts that he is employed as manager, does not have other debtors and credits taken, posses life insurance and house, is not a foreign worker, has small amount of money on checking account. Similar, attributes can be seen in the case of the first "fair" instance with good credit score that is female. She is not a foreign worker, employed as manager for 7 or more years, paid back duly existing credits and took credit for buying new car. She has small amount of money on checking account and does not have other debtors. Unlike this two instances, the first instance with bad credit score is unemployed man, that has other debtors, is foreign worker, does have house and car. It is obviously that unemployment and other debts has the most influential impact on labelling this instance as bad.

It can be concluded that all presented instances have reasonable explanations why they are labelled with bad or good credit score. Furthermore, it can be seen that sex does not have any kind of cause on final decision so FAIR-scalar successfully labelled them as fair.

6. Conclusions

We introduced a Fair Adversarial Instance Re-weighting (FAIR) discriminative method, which uses adversarial training to learn instance weights to ensure fairness.

We proposed four different variants of the method: a non probabilistic one and three models cast in fully probabilistic framework. In addition, we presented a possibility to introduce a baseline to reduce variance of gradient estimation for models based on score function. Theoretical analysis of FAIR model behaviour in terms of interactions of its main elements is given. We explained how changing the value of the hyperparameter controls the trade-off between model fairness and predictive performance. In experimental evaluation on five real-world tasks we demonstrated that our models are better or equal to previous state-of-the-art approaches with respect to fairness metrics and classification performance. Its additional desirable feature is its ability to provide interpretable information on the individual fairness of instances which existing adversarial approaches do not provide. Moreover, in the qualitative study, we demonstrated that FAIR model is able to find "fair" instances for small values of the hyperparameter α.

Further studies should address extending FAIR models to numerical and categorical values of sensitive attributes and adding additional loss constraints for individual fairness.

Acknowledgement

This work was supported in part by the ONR/ONR Global under Grant N62909-19-1-2008. In addition, this research is partially supported by the Ministry of Science, Education and Technological Development of the Republic of Serbia grants OI174021, TR35004 and TR41008. The authors would like to express gratitude to company Saga New Frontier Group Belgrade, for supporting this research.

References

[1] A. Kumar, A. Kaur, M. Kumar, Face detection techniques: a review, Artificial Intelligence Review 52 (2) (2019) 927–948.

[2] A. Voulodimos, N. Doulamis, A. Doulamis, E. Protopapadakis, Deep learning for computer vision: A brief review, Computational intelligence and neuroscience 2018 (2018).

[3] S. P. Singh, A. Kumar, H. Darbari, L. Singh, A. Rastogi, S. Jain, Machine translation using deep learning: An overview, in: 2017 International Conference on Computer, Communications and Electronics (Comptelix), IEEE, 2017, pp. 162–167.

[4] M. K. Domadiya, M. V. Gamit, M. K. Patel, A review on face detection and expression recognition (2019).

[5] R. P. Bunker, F. Thabtah, A machine learning framework for sport result prediction, Applied computing and informatics 15 (1) (2019) 27–33.

[6] S. Hajian, F. Bonchi, C. Castillo, Algorithmic bias: From discrimination discovery to fairness-aware data mining, in: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, 2016, pp. 2125–2126.

[7] N. Innocenti, Mining the pay gap: Compensation inequality still exists, Law Prac. 42 (2016) 56.

[8] Y. Li, Credit risk prediction based on machine learning methods, in: 2019 14th International Conference on Computer Science & Education (ICCSE), IEEE, 2019, pp. 1011–1013.

[9] K. Boyd, D. Teres, J. Rapoport, S. Lemeshow, The relationship between age and the use of dnr orders in critical care patients: Evidence for age discrimination, Archives of Internal Medicine 156 (16) (1996) 1821–1826.

[10] P. T. Kim, Data-driven discrimination at work, William & Mary Law Review 58 (2016) 857.

[11] X. Wang, H. Huang, Approaching machine learning fairness through adversarial network, arXiv preprint arXiv:1909.03013 (2019).

[12] F. Kamiran, A. Karim, X. Zhang, Decision theory for discrimination-aware classification, in: 2012 IEEE 12th International Conference on Data Mining, IEEE, 2012, pp. 924–929.

[13] F. Calmon, D. Wei, B. Vinzamuri, K. N. Ramamurthy, K. R. Varshney, Optimized pre-processing for discrimination prevention, in: Advances in Neural Information Processing Systems, 2017, pp. 3992–4001.

[14] M. B. Zafar, I. Valera, M. Gomez-Rodriguez, K. P. Gummadi, Fairness constraints: A flexible approach for fair classification., Journal of Machine Learning Research 20 (75) (2019) 1–42.

[15] T. Adel, I. Valera, Z. Ghahramani, A. Weller, One-network adversarial fairness, in: Thirty-Third AAAI Conference on Artificial Intelligence, 2019.

[16] L. E. Celis, L. Huang, V. Keswani, N. K. Vishnoi, Classification with fairness constraints: A meta-algorithm with provable guarantees, in: Proceedings of the Conference on Fairness, Accountability, and Transparency, 2019, pp. 319–328.

[17] T. Kamishima, S. Akaho, H. Asoh, J. Sakuma, Fairness-aware classifier with prejudice remover regularizer, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2012, pp. 35–50.

[18] M. Hardt, E. Price, N. Srebro, Equality of opportunity in supervised learning, in: Advances in neural information processing systems, 2016, pp. 3315–3323.

[19] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, K. Q. Weinberger, On fairness and calibration, in: Advances in Neural Information Processing Systems, 2017, pp. 5680–5689.

[20] F. Kamiran, T. Calders, Data preprocessing techniques for classification without discrimination, Knowledge and Information Systems 33 (1) (2012) 1–33.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in neural information processing systems, 2014, pp. 2672–2680.

[22] M. Nouiehed, M. Sanjabi, T. Huang, J. D. Lee, M. Razaviyayn, Solving a class of non-convex min-max games using iterative first order methods, in: Advances in Neural Information Processing Systems, 2019, pp. 14934–14942.

[23] Y.-P. Hsieh, C. Liu, V. Cevher, Finding mixed nash equilibria of generative adversarial networks, in: International Conference on Machine Learning, 2019, pp. 2810–2819.

[24] C. Wadsworth, F. Vera, C. Piech, Achieving fairness through adversarial learning: an application to recidivism prediction, arXiv preprint arXiv:1807.00199 (2018).

[25] D. Madras, E. Creager, T. Pitassi, R. Zemel, Learning adversarially fair and transferable representations, arXiv preprint arXiv:1802.06309 (2018).

[26] G. Cevora, Fair adversarial networks, arXiv preprint arXiv:2002.12144 (2020).

[27] V. Grari, S. Lamprier, M. Detyniecki, Adversarial learning for counterfactual fairness, arXiv preprint arXiv:2008.13122 (2020).

[28] S. Barocas, A. D. Selbst, Big data's disparate impact, California Law Review 104 (2016) 671.

[29] S. A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, D. Roth, A comparative study of fairness-enhancing interventions in machine learning, in: Proceedings of the Conference on Fairness, Accountability, and Transparency, ACM, 2019, pp. 329–338.

[30] S. Corbett-Davies, S. Goel, The measure and mismeasure of fairness: A critical review of fair machine learning, arXiv preprint arXiv:1808.00023 (2018).

[31] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, S. Venkatasubramanian, Certifying and removing disparate impact, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 259–268.

[32] E. Krasanakis, E. Spyromitros-Xioufis, S. Papadopoulos, Y. Kompatsiaris, Adaptive sensitive reweighting to mitigate bias in fairness-aware classification, in: Proceedings of the 2018 World Wide Web Conference, International World Wide Web Conferences Steering Committee, 2018, pp. 853–862.

[33] M. Ren, W. Zeng, B. Yang, R. Urtasun, Learning to reweight examples for robust deep learning, arXiv preprint arXiv:1803.09050 (2018).

[34] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, D. Meng, Meta-weight-net: Learning an explicit mapping for sample weighting, in: Advances in Neural Information Processing Systems, 2019, pp. 1919–1930.

[35] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, L. Fei-Fei, Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels, in: International Conference on Machine Learning, 2018, pp. 2304–2313.

[36] C. Louizos, K. Swersky, Y. Li, M. Welling, R. Zemel, The variational fair autoencoder, arXiv preprint arXiv:1511.00830 (2015).

[37] B. H. Zhang, B. Lemoine, M. Mitchell, Mitigating unwanted biases with adversarial learning, in: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, ACM, 2018, pp. 335–340.

[38] H. Zhao, A. Coston, T. Adel, G. J. Gordon, Conditional learning of fair representations, arXiv preprint arXiv:1910.07162 (2019).

[39] R. Ragonesi, R. Volpi, J. Cavazza, V. Murino, Learning unbiased representations via mutual information backpropagation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 2729–2738.

[40] A. Cotter, H. Jiang, M. R. Gupta, S. Wang, T. Narayan, S. You, K. Sridharan, Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals., J. Mach. Learn. Res. 20 (172) (2019) 1–59.

[41] A. Shah, D. Knowles, Z. Ghahramani, An empirical study of stochastic variational inference algorithms for the beta bernoulli process, in: International Conference on Machine Learning, 2015, pp. 1594–1603.

[42] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society: Series B (Methodological) 58 (1) (1996) 267–288.

[43] A. Frank, A. Asuncion, et al., Uci machine learning repository, 2010, URL http://archive.ics.uci.edu/ml 15 (2011) 22.

[44] R. Kohavi, Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid., in: Knowledge Discovery in Databases, Vol. 96, 1996, pp. 202–207.

[45] D. Dua, C. Graff, UCI machine learning repository (2017).
URL http://archive.ics.uci.edu/ml

[46] G. Stiglic, P. P. Brzan, N. Fijacko, F. Wang, B. Delibasic, A. Kalousis, Z. Obradovic, Comprehensible predictive modeling using regularized logistic regression and comorbidity based features, PloS one 10 (12) (2015).

[47] R. K. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilović, et al., Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias, IBM Journal of Research and Development 63 (4/5) (2019) 4–1.

[48] S. Bock, M. Weiß, A proof of local convergence for the adam optimizer, in: 2019 International Joint Conference on Neural Networks (IJCNN), IEEE, 2019, pp. 1–8.

[49] R. T. Marler, J. S. Arora, Survey of multi-objective optimization methods for engineering, Structural and multidisciplinary optimization 26 (6) (2004) 369–395.

[50] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.

Appendix A. FAIR-Bernoulli and FAIR-betaSF with baselines

Baseline functions are a commonly used tool to reduce the variance of the estimate of the gradient in reinforcement learning algorithms. It is shown that introducing a baseline in loss function does not introduce additional bias into the model [50]. Here we explain how these techniques can be incorporated in FAIR models. These modifications are not included in experimental evaluation since the technique is already known and our main goal is to compare basic FAIR variants against existing baselines, but we still derive algorithms with baselines as they might be of practical importance.

Already discussed adversarial loss is augmented by adding another term – the baseline loss:

$$\mathcal{L}_\alpha(\mu) = \mathrm{Var}\left[w \cdot \nabla_{\theta_g} \log P_g(w|\mathbf{x}) \cdot \big(\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x})\big) - b_\mu(\mathbf{x})\right] \tag{A.1}$$

The baseline loss includes the gradient of the adversarial loss, since its purpose is to reduce the variance of the estimate of that gradient. Keeping in mind that the variance can be represented as $\mathrm{Var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$ (where squaring of a vector v means $v^T v$), the baseline loss can be simplified due to the fact that it holds $\mathbb{E}_{P_\theta(w|\mathbf{x})}[\nabla_\theta \log P_\theta(w|\mathbf{x}) b_\mu(\mathbf{x})] = 0$ [50]:

$$\mathcal{L}_\alpha(\mu) = \mathbb{E}\left[\big(w \cdot \nabla_\theta \log P_\theta(w|\mathbf{x}) \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x})) - b_\mu(\mathbf{x})\big)^2\right] \tag{A.2}$$

where the expectation is with respect to $(\mathbf{x}, \mathbf{y}, \mathbf{s}) \sim P(\mathbf{x}, \mathbf{y}, \mathbf{s})$ and $w \sim P(w|\mathbf{x})$. Furthermore, we assumed the independence among the values involved in the expectation, and thus the expectation can be represented as:

$$\mathcal{L}_\alpha(\mu) = \mathbb{E}\left[\big(\nabla_\theta \log P_\theta(w|\mathbf{x})\big)^2\right] \cdot \mathbb{E}\left[\big(w \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x})) - b_\mu(\mathbf{x})\big)^2\right] \tag{A.3}$$

Considering that the first factor is constant with respect to b_μ it can be omitted, so that the final form of the loss $\mathcal{L}_\alpha(\mu)$ is:

$$\mathbb{E}\left[\big(w \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x})) - b_\mu(\mathbf{x})\big)^2\right] \tag{A.4}$$

Then, the gradient $\nabla_\mu \mathcal{L}_\alpha(\mu)$ is:

$$-\mathbb{E}\left[w \cdot \nabla_\mu b_\mu(\mathbf{x}) \cdot \big(\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x}) - b_\mu(\mathbf{x})\big)\right] \tag{A.5}$$

In Fig. A.8 graphical representation of FAIR-betaSF with baseline is shown. The pseudo-code is presented in Algorithm 2.

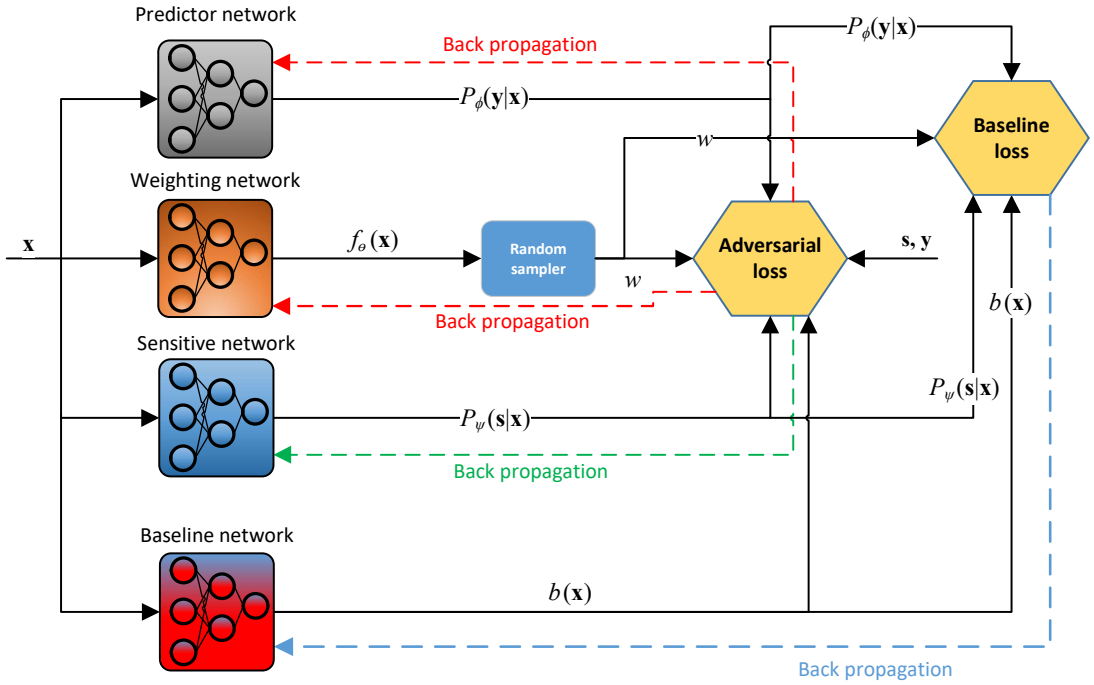

Figure A.8: Graphical representations of FAIR-betaSF model with baseline function

Appendix B. Model architecture

In all experiments with Reweighing - RF and DI - RF 500 trees were used in the random forest algorithm. Architectures, learning rates and maximum number of epochs used in models with neural networks for all datasets are presented in Tables B.7, B.8, B.9, B.10 and B.11.

Appendix C. Additional results

More detailed results of experimental evaluation are given in Figs. C.9, C.10, C.11, C.12, C.13, which present Pareto fronts with all dominated and non-dominated models.

Table B.7: Architectures of models used

Model	No. of units per layer $P_\theta(w\|\mathbf{x})$ or $P_\theta(\mathbf{z}\|\mathbf{x})$	No. of units per layer $P_\phi(\mathbf{y}\|\mathbf{x})$	No. of cells per layer $P_\psi(\mathbf{s}\|\mathbf{x})$	Activation	Early stopping epoch / learning rate
Adult					
DI - NN	-	93/62/41/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
Reweighing - NN	-	93/62/41/27/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAD	93/62/41/27/18	18/12/1	18/12/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-scalar	93/62/41/1	93/62/41/1	93/62/41/27/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-betaSF	93/62/41/1	93/62/41/27/1	93/62/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-betaREP	93/62/1	93/62/1	93/62/41/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-Bernoulli	93/62/1	93/62/41/1	93/62/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_diff	-	93/62/41/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_non_diff	-	93/62/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
LURMI	93/62/41/27/18	18/12/8/1	18/12/8/5/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
CLFR	93/62/41/27/18	18/12/8/1	18/12/8/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$

Table B.8: Architectures of models used

Model	No. of units per layer $P_\theta(w\|\mathbf{x})$ or $P_\theta(\mathbf{z}\|\mathbf{x})$	No. of units per layer $P_\phi(\mathbf{y}\|\mathbf{x})$	No. of units per layer $P_\psi(\mathbf{s}\|\mathbf{x})$	Activation	Early stopping epoch / learning rate
Readmission					
DI - NN	-	929/619/412/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
Reweighing - NN	-	929/619/412/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAD	929/619/412	412/274/182/1	412/274/182/121/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-scalar	929/619/1	929/619/412/1	929/619/412/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-betaSF	929/619/412/1	929/619/412/1	929/619/412/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-betaREP	929/619/412/274/1	929/619/412/1	929/619/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-Bernoulli	929/619/412/1	929/619/412/1	929/619/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_diff	-	929/619/412/274/182/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_non_diff	-	929/619/412/274/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
LURMI	929/619/412/274	274/182/121/1	274/182/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
CLFR	929/619/412	412/274/1	412/274/182/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$

Table B.9: Architectures of models used

Model	No. of units per layer $P_\theta(w\|\mathbf{x})$ or $P_\theta(\mathbf{z}\|\mathbf{x})$	No. of units per layer $P_\phi(\mathbf{y}\|\mathbf{x})$	No. of cells per layer $P_\psi(\mathbf{s}\|\mathbf{x})$	Activation	Early stopping epoch / learning rate
Medical expenditures					
DI - NN	-	137/91/60/40/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
Reweighing - NN	-	137/91/60/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAD	137/91/60	60/40/26/1	60/40/26/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-scalar	137/91/1	137/91/60/40/1	137/91/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-betaSF	137/91/60/1	137/91/60/1	137/91/60/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-betaREP	137/91/1	137/91/60/40/1	137/91/60/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-Bernoulli	137/91/60/1	137/91/1	137/91/60/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_diff	-	137/91/60/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_non_diff	-	137/91/60/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
LURMI	137/91/60	60/40/1	60/40/26/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
CLFR	137/91/60/40/26	26/17/1	26/17/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$

Table B.10: Architectures of models used

Model	No. of units per layer $P_\theta(w\|\mathbf{x})$ or $P_\theta(\mathbf{z}\|\mathbf{x})$	No. of units per layer $P_\phi(\mathbf{y}\|\mathbf{x})$	No. of cells per layer $P_\psi(\mathbf{s}\|\mathbf{x})$	Activation	Early stopping epoch / learning rate
German credit - sex					
DI - NN	-	56/37/24/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
Reweighing - NN	-	37/24/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAD	56/37/24/16	16/10/6/1	16/10/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-scalar	56/37/1	56/37/24/1	56/37/24/16/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-betaSF	56/37/24/1	56/37/24/1	56/37/24/16/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-betaREP	56/37/24/16/1	56/37/24/1	56/37/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-Bernoulli	56/37/24/1	56/37/24/1	56/37/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_diff	-	56/37/24/16/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_non_diff	-	56/37/24/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
LURMI	56/37/24	24/16/1	24/16/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
CLFR	56/37/24/16	16/10/6/1	16/10/6/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$

Table B.11: Architectures of models used

Model	No. of units per layer $P_\theta(w\|\mathbf{x})$ or $P_\theta(\mathbf{z}\|\mathbf{x})$	No. of units per layer $P_\phi(\mathbf{y}\|\mathbf{x})$	No. of cells per layer $P_\psi(\mathbf{s}\|\mathbf{x})$	Activation	Early stopping epoch / learning rate
German credit - age					
DI - NN	-	56/37/24/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
Reweighing - NN	-	56/37/24/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAD	56/37/24/16	16/10/6/1	16/10/6/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-scalar	56/37/24/1	56/37/24/16/1	56/37/24/16/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
FAIR-betaSF	56/37/1	56/37/24/16/1	56/37/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-betaREP	56/37/24/16/1	56/37/24/1	56/37/1	ReLU + Batch normalization + sigmoid or exp (last layer)	$15/10^{-4}$
FAIR-Bernoulli	56/37/24/16/1	56/37/1	56/37/24/16/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_diff	-	56/24/16/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
PYCO_non_diff	-	56/24/16/1	-	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
LURMI	56/37/24/16	16/10/1	16/10/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$
CLFR	56/37/24	24/16/10/1	24/16/1	ReLU + Batch normalization + sigmoid (last layer)	$15/10^{-4}$

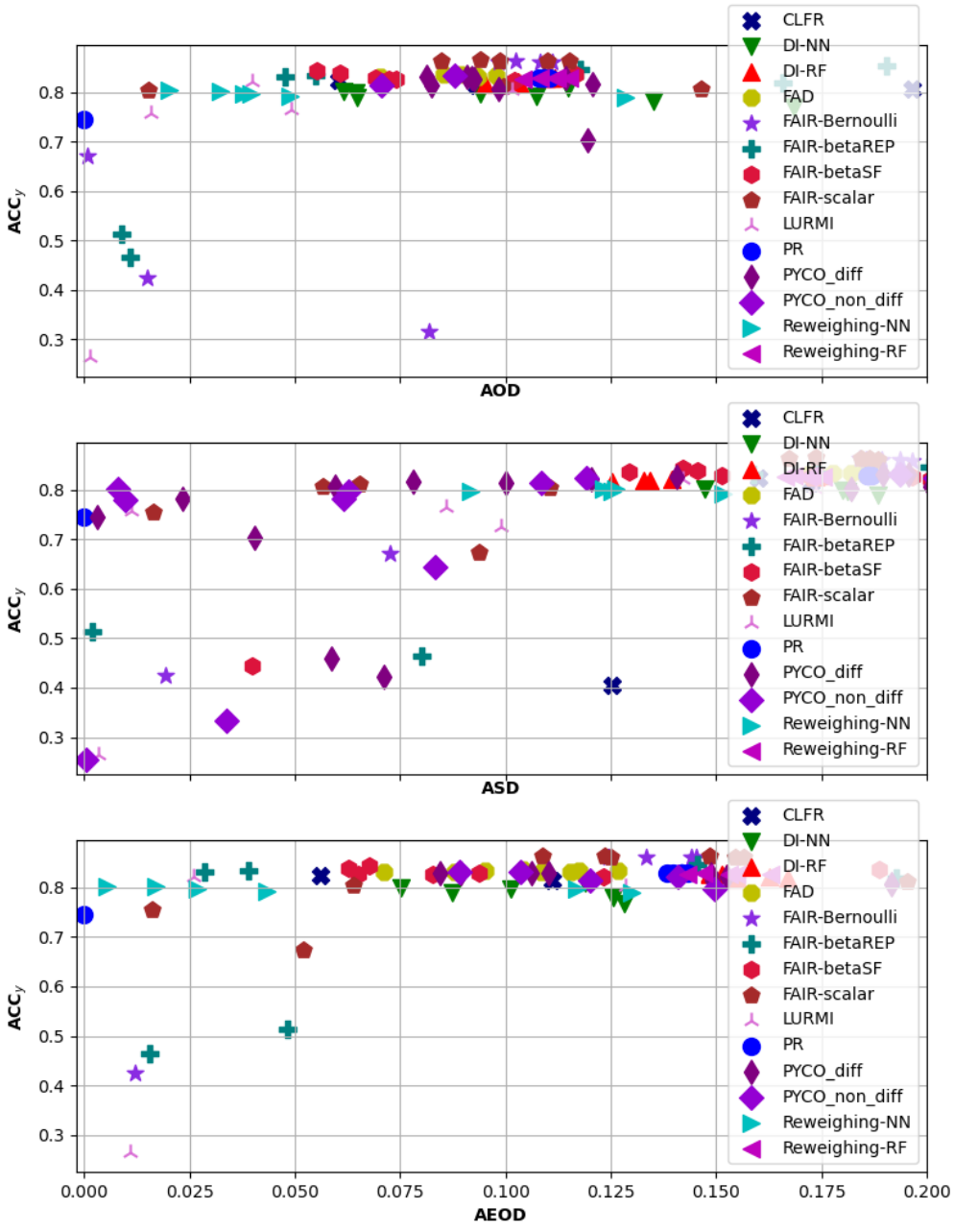

Figure C.9: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *Adult income* dataset

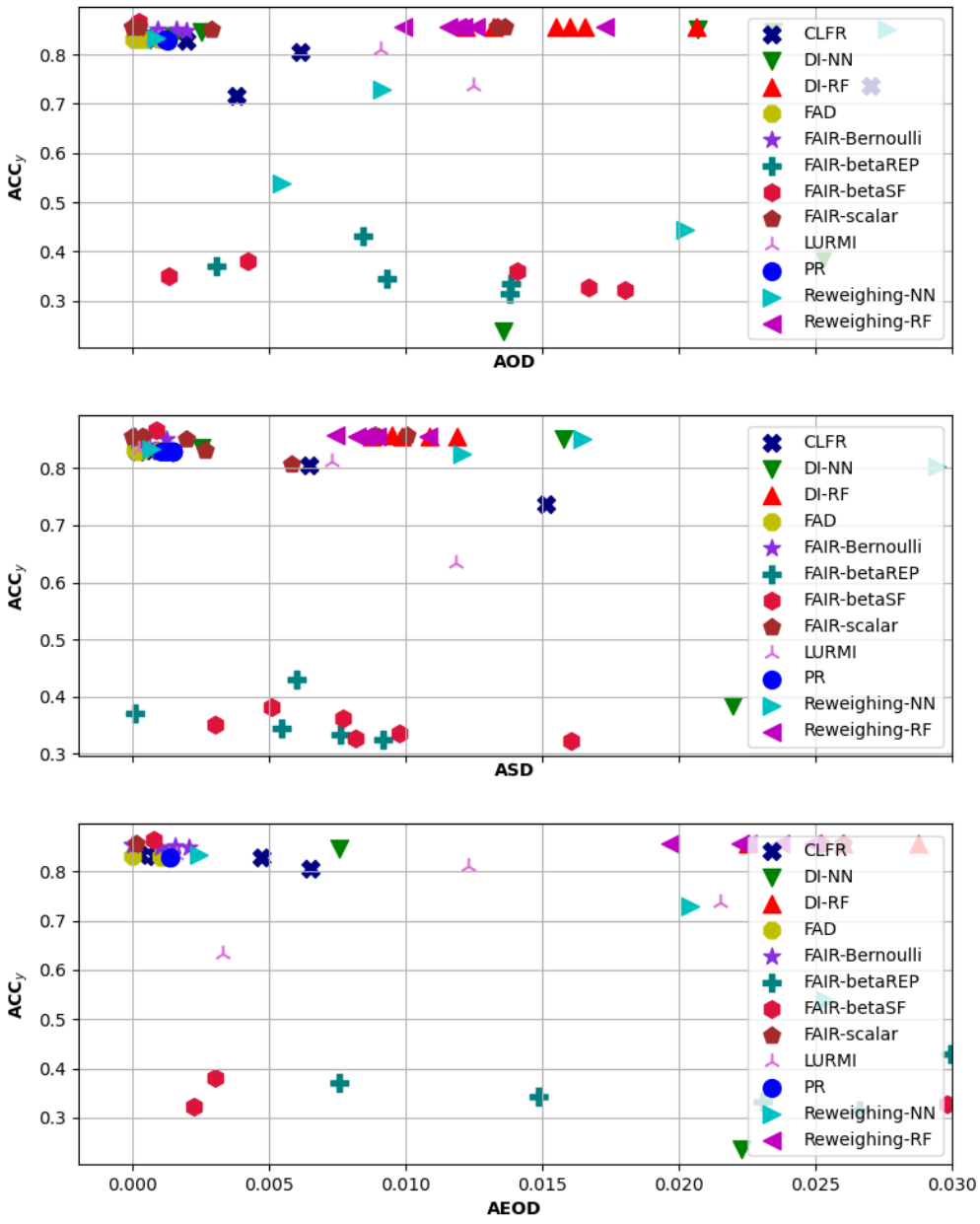

Figure C.10: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *Hospital readmission* dataset

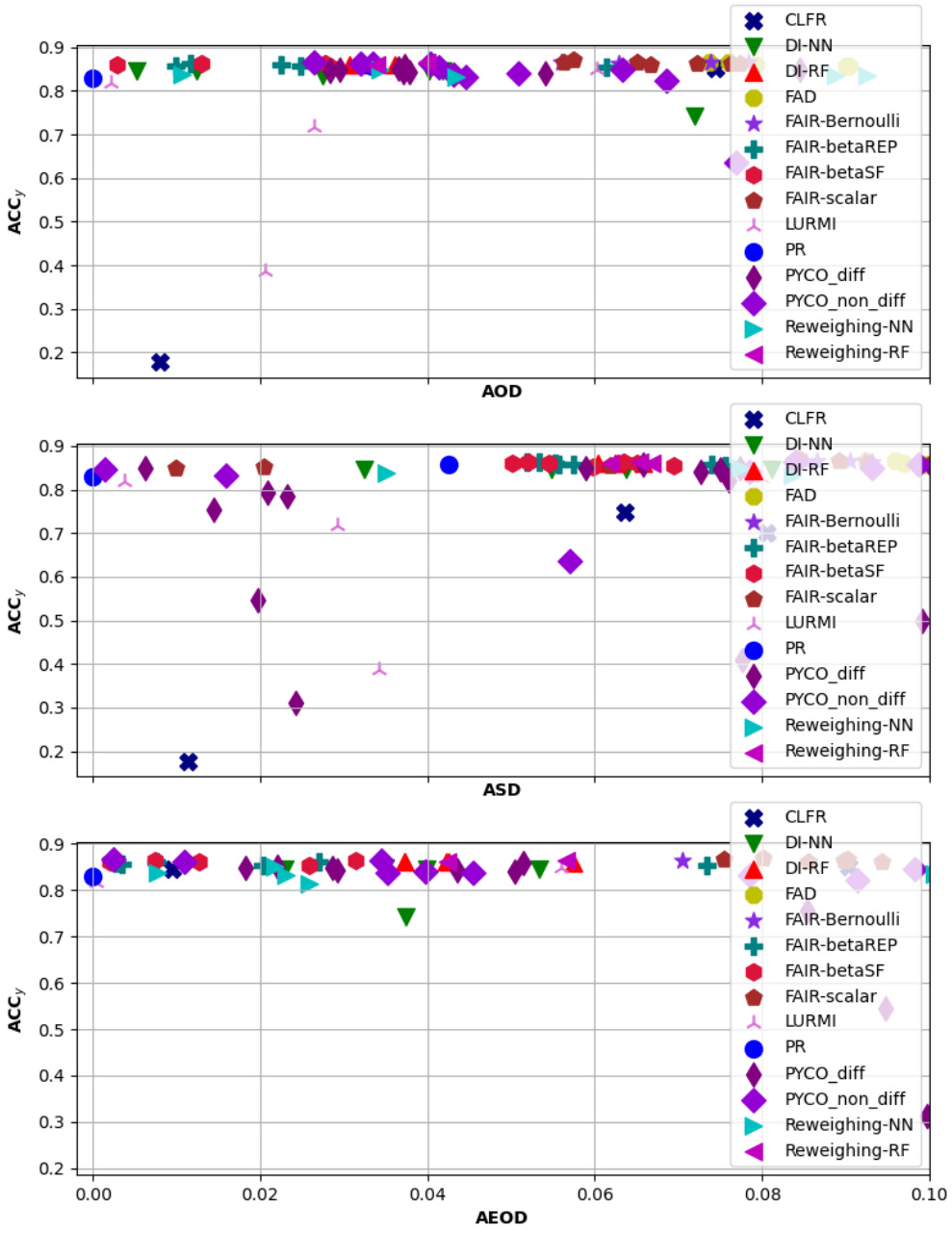

Figure C.11: Classification performance and fairness of models as measured by $\text{ACC}_\mathbf{y}$ and AOD, ASD, or AEOD on the *Hospital expenditures* dataset

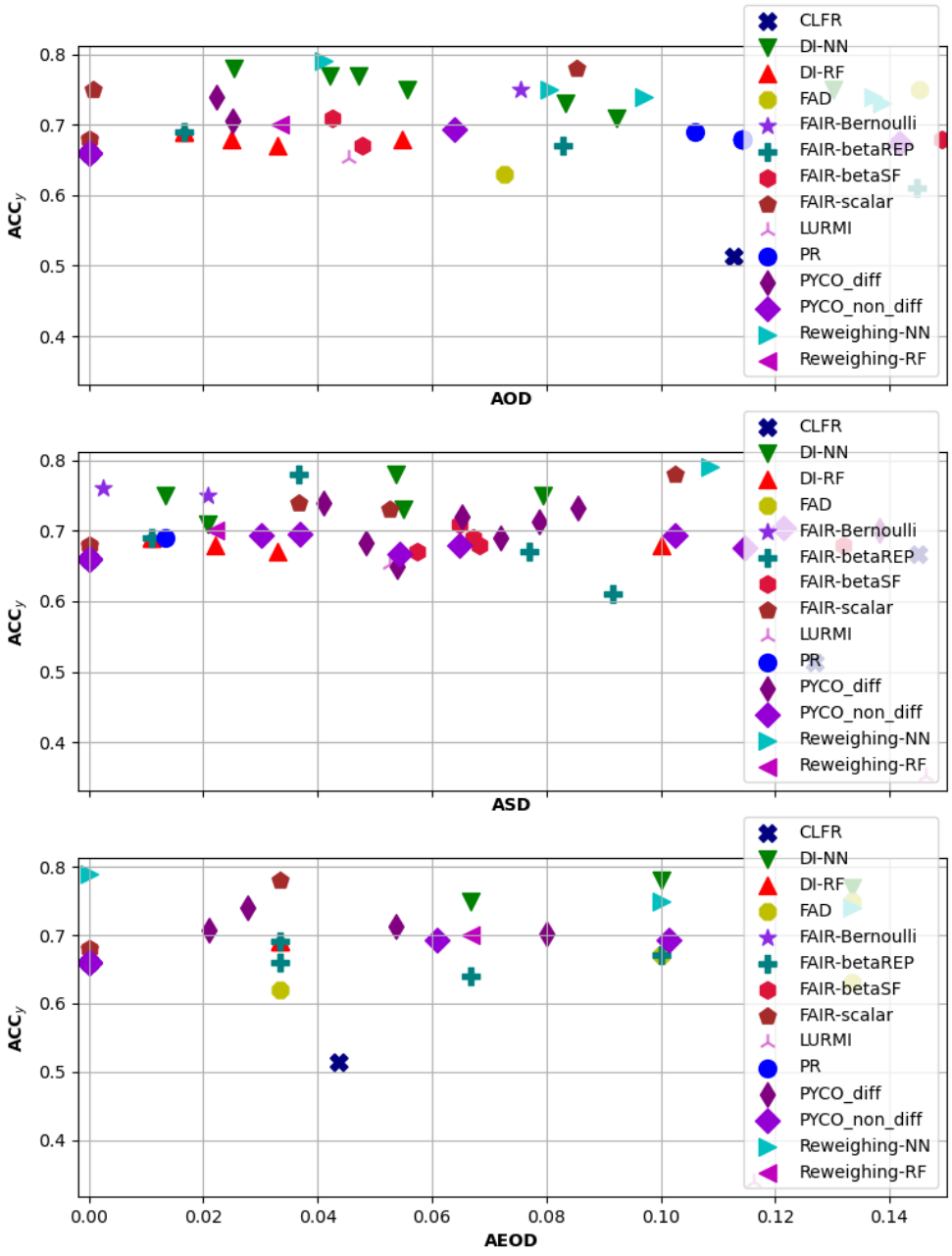

Figure C.12: Classification performance and fairness of models as measured by $ACC_{\mathbf{y}}$ and AOD, ASD, or AEOD on the *German credit* (age) dataset

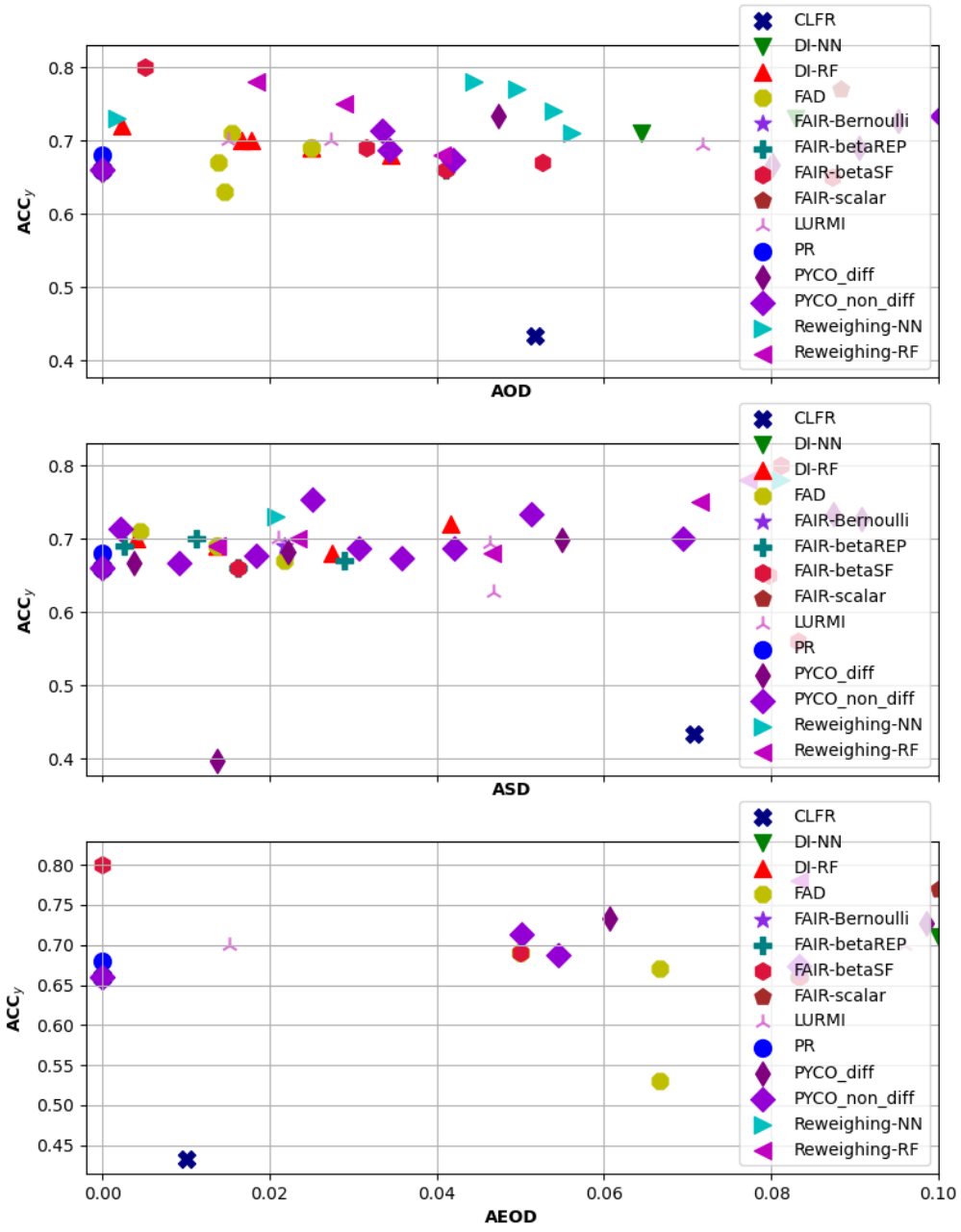

Figure C.13: Classification performance and fairness of models as measured by ACC_y and AOD, ASD, or AEOD on the *German credit* (sex) dataset

Algorithm 2 FAIR-betaSF with baseline

Input: learning rates $\gamma_\theta, \gamma_\phi, \gamma_\psi$, γ_b dataset D, hyperparameter α, number of iterations M

Output: parameters θ, ϕ, ψ, μ

Initialize θ, ϕ, ψ, μ

for i = 1 to M **do**

Sample a mini-batch $B \subseteq D$

$\alpha_{\mathbf{x}}, \beta_{\mathbf{x}} \leftarrow f_\theta(\mathbf{x})$ for each $\mathbf{x} \in B$

Sample $w_{\mathbf{x}} \sim \beta(\alpha, \beta)$ for each $\mathbf{x} \in B$

$$d_\theta \leftarrow \gamma_\theta \frac{1}{|B|} \sum_{(\mathbf{x},\mathbf{y},\mathbf{s}) \in B} \big[w_{\mathbf{x}} \nabla_\theta \log P_\theta(w_{\mathbf{x}}|\mathbf{x}) \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x}) - b_\mu(\mathbf{x})) \big]$$

$$d_\phi \leftarrow \gamma_\phi \nabla_\phi \mathcal{L}_\alpha^{\mathcal{P}}(\theta, \phi, \psi, B)$$

$$d_\psi \leftarrow -\gamma_\psi \nabla_\psi \mathcal{L}_\alpha^{\mathcal{P}}(\theta, \phi, \psi, B)$$

$$d_\mu \leftarrow -\gamma_\mu \frac{1}{|B|} \sum_{(\mathbf{x},\mathbf{y},\mathbf{s}) \in B} \big[w_{\mathbf{x}} \nabla_\mu b_\mu(\mathbf{x}) \cdot (\alpha \log P_\psi(\mathbf{s}|\mathbf{x}) - \log P_\phi(\mathbf{y}|\mathbf{x}) - b_\mu(\mathbf{x})) \big]$$

$$(\theta, \phi, \psi, \mu) \leftarrow (\theta, \phi, \psi, \mu) - (d_\theta, d_\phi, d_\psi, d_\mu)$$