Datasets often contain biases which unfairly disadvantage certain groups, and classifiers trained on such datasets can inherit these biases. In this paper, we provide a mathematical formulation of how this bias can arise. We do so by assuming the existence of underlying, unknown, and unbiased labels which are overwritten by an agent who intends to provide accurate labels but may have biases against certain groups. Despite the fact that we only observe the biased labels, we are able to show that the bias may nevertheless be corrected by re-weighting the data points without changing the labels. We show, with theoretical guarantees, that training on the re-weighted dataset corresponds to training on the unobserved but unbiased labels, thus leading to an unbiased machine learning classifier. Our procedure is fast and robust and can be used with virtually any learning algorithm. We evaluate on a number of standard machine learning fairness datasets and a variety of fairness notions, finding that our method outperforms standard approaches in achieving fair classification.

1 INTRODUCTION

Machine learning has become widely adopted in a variety of real-world applications that significantly affect people’s lives (Guimaraes and Tofiqi 2018; Guegan and Hassani 2018). Fairness in these algorithmic decision-making systems has thus become an increasingly important concern: It has been shown that without appropriate intervention during training or evaluation, models can be biased against certain groups (Angwin et al. 2016; Hardt et al. 2016). This is due to the fact that the data used to train these models often contains biases that become reinforced into the model (Bolukbasi et al. 2016). Moreover, it has been shown that simple remedies, such as ignoring the features corresponding to the protected groups, are largely ineffective due to redundant encodings in the data (Pedreshi et al. 2008). In other words, the data can be inherently biased in possibly complex ways, thus making it difficult to achieve fairness.

Research on training fair classifiers has therefore received a great deal of attention. One such approach has focused on developing post-processing steps to enforce fairness on a learned model (Doherty et al. 2012; Feldman 2015; Hardt et al. 2016). That is, one first trains a machine learning model, resulting in an unfair classifier. The outputs of the classifier are then calibrated to enforce fairness. Although this approach is likely to decrease the bias of the classifier, by decoupling the training from the fairness enforcement, this procedure may not lead to the best trade-off between fairness and accuracy (Woodworth et al. 2017). Accordingly, recent work has proposed to incorporate fairness into the training algorithm itself, framing the problem as a constrained optimization problem and subsequently applying the method of Lagrange multipliers to transform the constraints to penalties (Zafar et al. 2015; Goh et al. 2016; Cotter et al. 2018b; Agarwal et al. 2018); however such approaches may introduce undesired complexity and lead to more difficult or unstable training (Cotter et al. 2018b,c). Both of these existing methods address the problem of bias by adjusting the machine learning model. However, oftentimes it is the training data itself – i.e., the observed features and corresponding labels – which are biased, and thus it is potentially more fruitful to tackle this problem directly.

In this paper, we provide an approach to machine learning fairness that addresses the underlying data bias problem directly. We introduce a new mathematical framework for fairness in which we assume that there exists an unknown but unbiased ground truth.
label function and that the labels observed in the data are assigned by an agent who is possibly biased, but otherwise has the intention of being accurate. This assumption is natural in practice, where the observed data may be the result of manual labelling done by actors (e.g. human decision-makers) who strive to provide an accurate label while being affected by (potentially unconscious) biases; or in cases where the observed labels correspond to a process (e.g. results of a written exam) devised to be accurate and fair, but which is nevertheless affected by inherent biases.

Based on this mathematical formulation, we show how one may identify the bias of the observed label function as a closed form expression. Furthermore, our derived form for the bias suggests that its correction may be performed by assigning appropriate weights to each example in the training data. We show, with theoretical guarantees, that training the classifier under the resulting weighted objective leads to an unbiased classifier on the original un-weighted dataset. Notably, many pre-processing approaches and even constrained optimization approaches (e.g. [Agarwal et al., 2018]) optimize a loss which possibly modifies the observed labels or features, and doing so may be interpreted as training on falsified data. In contrast, our method does not modify any of the observed labels or features. Rather, we correct for the bias by changing the distribution of the sample points via re-weighting the dataset.

Our resulting method is general and can be applied to various notions of fairness, including demographic parity, equal opportunity, equalized odds, and disparate impact. Moreover, the method is practical and simple to tune: With the appropriate example weights, any off-the-shelf classification procedure can be used on the weighted dataset to learn a fair classifier. Experimentally, we show that on standard fairness benchmark datasets and under a variety of fairness notions our method can outperform previous approaches to fair classification.

2 BACKGROUND

In this section, we introduce our framework for machine learning fairness, which explicitly assumes an unknown and unbiased ground truth label function. We additionally introduce notation and definitions used in the subsequent presentation of our method.

2.1 Biased and Unbiased Labels

Consider a data domain \( \mathcal{X} \) and an associated data distribution \( \mathcal{P} \). An element \( x \in \mathcal{X} \) may be interpreted as a feature vector associated with a specific example. We let \( \mathcal{Y} := \{0, 1\} \) be the labels, considering the binary classification setting, although our method may be readily generalized.

We assume the existence of an unbiased, ground truth label function \( y_{\text{true}} : \mathcal{X} \to [0, 1] \). Although \( y_{\text{true}} \) is the assumed ground truth, in general we do not have access to it. Rather, our dataset has labels generated based on a biased label function \( y_{\text{bias}} : \mathcal{X} \to [0, 1] \). Accordingly, we assume that our data is drawn according to \( (x, y) \sim \mathcal{D} \equiv x \sim \mathcal{P}, y \sim \text{Bernoulli}(y_{\text{bias}}(x)) \), and we assume access to a finite sample \( \mathcal{D}_n := \{(x^{(i)}, y^{(i)})\}_{i=1}^n \) drawn from \( \mathcal{D} \).

In a machine learning context, our directive is to use the dataset \( \mathcal{D} \) to recover the unbiased, true label function \( y_{\text{true}} \). In general, the relationship between the desired \( y_{\text{true}} \) and the observed \( y_{\text{bias}} \) is unknown. Without additional assumptions and with access only to labels from \( y_{\text{bias}} \), it is impossible to learn a machine learning model to fit \( y_{\text{true}} \). We will attack this problem in the following sections by proposing a minimal and natural assumption on the relationship between \( y_{\text{true}} \) and \( y_{\text{bias}} \). The assumption will allow us to derive an expression for \( y_{\text{true}} \) in terms of \( y_{\text{bias}} \), and the form of this expression will immediately imply that correction of the label bias may be done by appropriately re-weighting the data.
We now discuss precise ways in which $y_{bias}$ can be biased. We describe a number of proposed notions of fairness; i.e., what it means for an arbitrary label function or machine learning model $h : \mathcal{X} \to [0,1]$ to be biased (unfair) or unbiased (fair).

We will define the notions of fairness in terms of a constraint function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. Many of the common notions of fairness may be expressed or approximated as linear constraints on $h$ (introduced previously by Cotter et al. (2018c), Goh et al. (2016)). That is, they are of the form

$$\mathbb{E}_{x \sim \mathcal{P}}[(h(x), c(x))] = 0,$$

where $(h(x), c(x)) := \sum_{y \in \mathcal{Y}} h(y|x)c(x,y)$ and we use the shorthand $h(y|x)$ to denote the probability of sampling $y$ from a Bernoulli random variable with $p = h(x)$; i.e., $h(1|x) := h(x)$ and $h(0|x) := 1 - h(x)$. Therefore, a label function $h$ is unbiased with respect to the constraint function $c$ if $\mathbb{E}_{x \sim \mathcal{P}}[(h(x), c(x))] = 0$. If $h$ is biased, the degree of bias (positive or negative) is given by $\mathbb{E}_{x \sim \mathcal{P}}[(h(x), c(x))]$.

We define the notions of fairness with respect to a protected group $\mathcal{G}$, and thus assume access to an indicator function $g(x) = 1[x \in \mathcal{G}]$. We use $Z_{\mathcal{G}} := \mathbb{E}_{x \sim \mathcal{P}}[g(x)]$ to denote the proportion of a sample drawn from $\mathcal{P}$ to be in $\mathcal{G}$. We use $P_{\mathcal{X}} = \mathbb{E}_{x \sim \mathcal{P}}[y_{true}(x)]$ to denote the proportion of $\mathcal{X}$ which is positively labelled and $P_{\mathcal{G}} = \mathbb{E}_{x \sim \mathcal{P}}[g(x) \cdot y_{true}(x)]$ to denote the proportion of $\mathcal{X}$ which is positively labelled and in $\mathcal{G}$. We now give some concrete examples of accepted notions of constraint functions:

**Demographic parity** (Dwork et al. 2012): A fair classifier $h$ should make positive predictions on $\mathcal{G}$ at the same rate as on all of $\mathcal{X}$. The constraint function may be expressed as $c(x,0) = 0, c(x,1) = g(x)/Z_{\mathcal{G}} - 1$.

**Disparate impact** (Feldman et al. 2015): This is identical to demographic parity; only that, in addition, during inference the classifier does not have access to the indicator function $g(x)$.

**Equal opportunity** (Hardt et al. 2016): A fair classifier $h$ should have equal true positive rates on $\mathcal{G}$ as on all of $\mathcal{X}$. The constraint may be expressed as $c(x,0) = 0, c(x,1) = g(x)y_{true}(x)/P_{\mathcal{G}} - y_{true}(x)/P_{\mathcal{X}}$.

**Equalized odds** (Hardt et al. 2016): A fair classifier $h$ should have equal true positive and false positive rates on $\mathcal{G}$ as on all of $\mathcal{X}$. In addition to the constraint associated with equal opportunity, this notion applies an additional constraint with $c(x,0) = 0, c(x,1) = g(x)(1 - y_{true}(x))/(Z_{\mathcal{G}} - P_{\mathcal{G}}) - (1 - y_{true}(x))/(1 - P_{\mathcal{X}})$.

In practice, there are often multiple fairness constraints $\{c_k\}_{k=1}^K$ associated with multiple protected groups $\{\mathcal{G}_k\}_{k=1}^K$. It is clear that our subsequent results will assume multiple fairness constraints and protected groups, and that the protected groups may have overlapping samples.

## 3 MODELING HOW BIAS ARISES

We now introduce our underlying mathematical framework to understand bias in the data, by providing the relationship between $y_{bias}$ and $y_{true}$ (Assumption 1 and Proposition 1). This will allow us to derive a closed form expression for $y_{true}$ in terms of $y_{bias}$ (Corollary 1). In Section 4 we will show how this expression leads to a simple algorithm that performs appropriate weighting on data with biased labels in order to train a classifier to produce the true, unbiased labels.

We begin with an assumption on the relationship between the observed $y_{bias}$ and the underlying $y_{true}$.

**Assumption 1.** Let the fairness constraints be $c_1, ..., c_K$, with respect to which $y_{true}$ is unbiased (i.e. $\mathbb{E}_{x \sim \mathcal{P}}[(y_{true}(x), c_k(x))] = 0$ for $k \in [K]$). We assume that there exist $\epsilon_1, ..., \epsilon_K \in \mathbb{R}$ such that the observed, biased label function $y_{bias}$ is the solution of the following constrained optimization problem:

$$\arg \min_{\hat{y}: \mathcal{X} \to [0,1]} \mathbb{E}_{x \sim \mathcal{P}}[D_{KL}(\hat{y}(x) || y_{true}(x))]
$$

s.t. $\mathbb{E}_{x \sim \mathcal{P}}[(\hat{y}(x), c_k(x))] = \epsilon_k$ for $k = 1, ..., K$,

where we use $D_{KL}$ to denote the KL-divergence.

In other words, we assume that $y_{bias}$ is the label function closest to $y_{true}$ while achieving some amount of bias, where proximity to $y_{true}$ is given by the KL-divergence. We note that this assumption is agnostic to the features (other than requiring that examples with equal features should have equal label functions). Thus it is also applicable to settings where the features themselves are biased and that the observed labels were generated by a process depending on the features (i.e. situations where there is bias in both the features and labels).

The KL-divergence in Assumption 1 presents a convex optimization problem, whose closed form is well-known (Friedlander and Gupta 2006, Botev and Kroese 2011). In our setting, the closed form provides us with the following simple expression for the observed $y_{bias}$. See the Appendix for a proof.

**Proposition 1.** Suppose that Assumption 1 holds. Then for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, $y_{bias}$ satisfies,

$$y_{bias}(y|x) \propto y_{true}(y|x) \cdot \exp\{ -\sum_{k=1}^{K} \lambda_k \cdot c_k(x,y)\},$$

for some $\lambda_1, ..., \lambda_K \in \mathbb{R}$.

---

We use equality constraints to simplify the analysis. Inequality constraints may be used as well, with the same results.
Given this form of $y_{\text{bias}}$ in terms of $y_{\text{true}}$, we can immediately deduce the form of $y_{\text{true}}$ in terms of $y_{\text{bias}}$: 

**Corollary 1.** Suppose that Assumption 1 holds. The unbiased label function $y_{\text{true}}$ is of the form,

$$y_{\text{true}}(y|x) \propto y_{\text{bias}}(y|x) \cdot \exp \left\{ \sum_{k=1}^{K} \lambda_k \cdot c_k(x,y) \right\},$$

for some $\lambda_1, \ldots, \lambda_K \in \mathbb{R}$.

We note that previous approaches to learning fair classifiers often formulate a constrained optimization problem similar to that appearing in Assumption 1 (i.e., maximize the accuracy or log-likelihood of a classifier subject to linear constraints) and then solve it, usually via the method of Lagrange multipliers which translates the constraints to penalties on the training loss. In our approach, rather than using the constrained optimization problem to formulate a training objective, we use it to express the generative process which modifies the true, underlying labels to the biased labels we observe. Furthermore, rather than training with respect to these biased labels, our approach aims to recover the true ones. As we will show in the following sections, this may be done by simply optimizing the training loss on a re-weighting of the dataset. In contrast, the penalties associated with Lagrangian approaches can often be cumbersome: The original, non-differentiable, fairness constraint functions must be relaxed or approximated before conversion to penalties. Even then, the derivatives of these approximations may be near-zero for large regions of the domain, causing difficulties during training.

### 4 LEARNING UNBIASED LABELS

We have derived a closed form expression for the true, unbiased label function $y_{\text{true}}$ in terms of the observed label function $y_{\text{bias}}$, coefficients $\lambda_1, \ldots, \lambda_K$, and constraint functions $c_1, \ldots, c_K$. In this section, we elaborate on how one may learn a machine learning model $h$ to fit $y_{\text{true}}$, given access to a dataset $D$ with labels sampled according to $y_{\text{bias}}$. We begin by restricting ourselves to constraints $c_1, \ldots, c_K$ associated with demographic parity, allowing us to have full knowledge of these constraint functions. The same method may be extended to general notions of fairness, and we provide the details for this in the Appendix.

With knowledge of the functions $c_1, \ldots, c_K$, it remains to determine the coefficients $\lambda_1, \ldots, \lambda_K$ (which give us a closed form expression for the dataset weights) as well as the classifier $h$. We present our method by first showing how a classifier $h$ may be learned assuming knowledge of the coefficients $\lambda_1, \ldots, \lambda_K$ (Section 4.1). We subsequently show how the coefficients themselves may be learned, thus allowing our algorithm to be used in general settings (Section 4.2).

#### 4.1 Learning $h$ Given $\lambda_1, \ldots, \lambda_K$

Although we have the closed form expression $y_{\text{true}}(y|x) \propto y_{\text{bias}}(y|x) \exp \left\{ \sum_{k=1}^{K} \lambda_k c_k \right\}$ for the true label function, in practice we do not have access to the values $y_{\text{bias}}(y|x)$ but rather only access to data points with labels sampled from $y_{\text{bias}}(y|x)$. We propose the weighting technique to train $h$ on labels based on $y_{\text{true}}$.

The weighting technique weights an example $(x,y)$ by the weight $w(x,y) = \tilde{w}(x,y)/\sum_{y' \in Y} \tilde{w}(x,y')$, where $\tilde{w}(x,y') = \exp \left\{ \sum_{k=1}^{K} \lambda_k c_k(x,y') \right\}$.

We have the following theorem, which states that weighting a loss according to $w(x,y)$ is equivalent to training with respect to the true, unbiased labels while biasing the feature distribution.

**Theorem 1.** For any loss function $\ell$, training a classifier $h$ on the weighted objective $E_{x,y \sim \tilde{P}, y \sim y_{\text{bias}}(x)}[\ell(h(x,y), y)]$ is equivalent to training the classifier on the objective $E_{x,y \sim \tilde{P}, y \sim y_{\text{true}}(x)}[\ell(h(x,y), y)]$ with respect to the underlying, true labels, for some distribution $\tilde{P}$ over $\mathcal{X}$.

Theorem 1 states that the bias in observed labels may be corrected in a simple and straightforward way by re-weighting the training examples. Theorem 1 also explicitly describes the trade-off introduced: While we gain the ability to train on unbiased labels, we implicitly modify the feature distribution $\tilde{P}$ over features $x$. In Section 5 we will resolve this issue and show that given some mild conditions (see Theorem 2), the change in feature distribution does not affect the bias of the final learned classifier. Therefore, in these cases, training with respect to weighted examples with biased labels is equivalent to training with respect to the same examples and the true labels.

#### 4.2 Determining the Coefficients $\lambda_1, \ldots, \lambda_K$

We now continue to describe how to learn the coefficients $\lambda_1, \ldots, \lambda_K$. One advantage of our approach is that, in practice, $K$ is often small. Thus, we propose to iteratively learn the coefficients so that the final classifier satisfies the desired fairness constraints either on the training data or on a validation set. We first discuss how to do this for demographic parity; we discuss extensions to other notions of fairness in the Appendix. See the full pseudocode for learning $h$ and $\lambda_1, \ldots, \lambda_K$ in Algorithm 1.

Intuitively, the idea is that if the positive prediction rate for a protected class $\mathcal{G}$ is lower than the overall positive prediction rate, then the corresponding coefficient should be increased; i.e., if we increase the

---

3See the Appendix for an alternative to the weighting technique – the *sampling* technique, based on a coin-flip.
Algorithm 1 Training a fair classifier for Demographic Parity, Disparate Impact, or Equal Opportunity (see Appendix for Equalized Odds).

**Inputs:** Learning rate $η$, number of loops $T$, training data $D_{[n]} = \{(x_{i}, y_{i})\}_{i=1}^{n}$, classification procedure $H$, constraints $c_{1}, \ldots, c_{K}$ corresponding to protected groups $G_{1}, \ldots, G_{K}$.

Initialize $λ_{1}, \ldots, λ_{K}$ to 0 and $w_{1} = w_{2} = \cdots = w_{n} = 1$.

Let $h := H(D_{[n]}; \{w_{i}\}_{i=1}^{n})$

for $t = 1, \ldots, T$ do

- Let $Δ_{k} := \mathbb{E}_{x \sim D_{[n]}}[(h(x), c_{k}(x))]$ for $k \in [K]$.
- Update $λ_{k} = λ_{k} - η \cdot Δ_{k}$ for $k \in [K]$.
- Let $w_{i} := \exp\left(\sum_{k=1}^{K} λ_{k} \cdot 1[x \in G_{k}]\right)$ for $i \in [n]$
- Let $w_{i} = w_{i}/(1 + w_{i})$ if $y_{i} = 1$, otherwise $w_{i} = 1/(1 + w_{i})$ for $i \in [n]$
- Update $h = H(D_{[n]}; \{w_{i}\}_{i=1}^{n})$

end for

Return $h$

Weights of the positively labeled examples of $G$ and decrease the weights of the negatively labeled examples of $G$, then this will encourage the classifier to increase its accuracy on the positively labeled examples in $G$, while the accuracy on the negatively labeled examples of $G$ may fall. Either of these two events will cause the positive prediction rate on $G$ to increase, and thus bring $h$ closer to the true, unbiased label function.

Accordingly, Algorithm 1 works by iteratively performing the following steps: (1) evaluate the demographic parity constraints; (2) update the coefficients by subtracting the respective constraint violation multiplied by a fixed step-size; (3) compute the weights for each sample based on these multipliers using the closed-form provided by Proposition 1 and (4) retrain the classifier given these weights. Algorithm 1 takes in a classification procedure $H$, which given a dataset $D_{[n]} := \{(x_{i}, y_{i})\}_{i=1}^{n}$ and weights $\{w_{i}\}_{i=1}^{n}$, outputs a classifier. In practice, $H$ can be any training procedure which minimizes a weighted loss function over some parametric function class (e.g. logistic regression).

4.3 Extensions to other fairness notions

The initial restriction to demographic parity was made so that the values of the constraint functions $c_{1}, \ldots, c_{K}$ on any $x \in \mathcal{X}, y \in \mathcal{Y}$ would be known. We note that Algorithm 1 works for disparate impact as well: the only change would be that the classifier does not have access to the protected attributes. However, in other notions of fairness, such as equal opportunity or equalized odds, the constraint functions depend on $y_{true}$, which is unknown.

For these cases, we propose to apply the same technique of iteratively re-weighting the loss to achieve the desired fairness notion, with the weights $w(x,y)$ on each example determined only by the protected attribute $g(x)$ and the observed label $y \in \mathcal{Y}$. This is equivalent to using Theorem 1 to derive the same procedure presented in Algorithm 1 but approximating the unknown constraint function $c(x,y)$ as a piece-wise constant function $d(g(x),y)$, where $d : \{0,1\} \times \mathcal{Y} \rightarrow \mathbb{R}$ is unknown. Although we do not have access to $d$, we may treat $d(g(x),y)$ as an additional set of parameters – one for each protected group attribute $g(x) \in \{0,1\}$ and each label $y \in \mathcal{Y}$. These additional parameters may be learned in the same way the $λ$ coefficients are learned. In some cases, their values may be wrapped into the unknown coefficients. For example, for equal opportunity, there is in fact no need for any additional parameters. On the other hand, for equalized odds, the unknown values for $λ_{1}, \ldots, λ_{K}$ and $d_{1}, \ldots, d_{K}$, are instead treated as unknown values for $λ_{1}^{op}, \ldots, λ_{K}^{op}, λ_{1}^{eq}, \ldots, λ_{K}^{eq}$; i.e., separate coefficients for positively and negatively labelled points. Due to space constraints, see the Appendix for further details on these and more general constraints. We explain how to use the procedure for Equal Opportunity below.

**Equal Opportunity:** Algorithm 1 can be directly used by replacing the demographic parity constraints with equal opportunity constraints. Recall that in equal opportunity, the goal is for the positive prediction rates on the positive examples of the protected group $G$ to match that of the overall. If the positive prediction rate for positive examples $G$ is less than that of the overall, then Algorithm 1 will up-weight the examples of $G$ which are positively labeled. This encourages the classifier to be more accurate on the positively labeled examples of $G$, which in other words means that it will encourage the classifier to increase its positive prediction rate on these examples, thus leading to a classifier satisfying equal opportunity. In this way, the same intuitions supporting the application of Algorithm 1 to demographic parity or disparate impact also support its application to equal opportunity. We note that in practice, we do not have access to the true labels function, so we approximate the constraint violation $\mathbb{E}_{x \sim D_{[n]}}[(h(x), c_{k}(x))]$ using the observed labels as $\mathbb{E}_{(x,y) \sim D_{[n]}}[h(x) \cdot c_{k}(x,y)]$.

5 THEORETICAL ANALYSIS

In this section, we provide theoretical guarantees on a learned classifier $h$ using the weighting technique. We show that with the coefficients $λ_{1}, \ldots, λ_{K}$ that satisfy Proposition 1 training on the re-weighted dataset leads to a finite-sample non-parametric rates of consistency on the estimation error provided the classifier
has sufficient flexibility.

We need to make the following regularity assumption on the data distribution, which assumes that the data is supported on a compact set in \( \mathbb{R}^D \) and \( y_{bias} \) is smooth (i.e. Lipschitz).

**Assumption 2.** \( X \) is a compact set over \( \mathbb{R}^D \) and both \( y_{bias}(x) \) and \( y_{true}(x) \) are L-Lipschitz (i.e. \( ||y_{bias}(x) - y_{bias}(x')|| \leq L \cdot |x - x'| \)).

We now give the result. The proof is technically involved and is deferred to the Appendix.

**Theorem 2 (Rates of Consistency).** Let \( 0 < \delta < 1 \). Let \( D_\nu = \{(x_i, y_i)\}_{i=1}^n \) be a sample drawn from \( D \). Suppose that Assumptions 1 and 2 hold. Let \( H \) be the set of all 2L-Lipschitz functions mapping \( X \) to \([0, 1]\). Suppose that the constraints are \( c_1, ..., c_K \) and the corresponding coefficients \( \lambda_1, ..., \lambda_K \) satisfy Proposition 1.

Let \( h^* \) be the optimal function in \( H \) under the weighted mean square error objective, where the weights satisfy Proposition 1. Then there exists \( C_0 \) depending on \( D \) such that for \( n \) sufficiently large depending on \( D \), we have with probability at least \( 1 - \delta \):

\[
||h^* - y_{true}||_2 \leq C_0 \cdot \log(2/\delta)^{1/(2+D)} \cdot n^{-1/(4+2D)},
\]

where \( ||h - h'||_2 := \mathbb{E}_{x \sim P}[((h(x) - h'(x))^2)] \).

Thus, with the appropriate values of \( \lambda_1, ..., \lambda_K \) given by Proposition 1, we see that training with the weighted dataset based on these values will guarantee that the final classifier will be close to \( y_{true} \). However, the above rate has a dependence on the dimension \( D \), which may be unattractive in high-dimensional settings. See the Appendix for an intrinsic dimension dependent result.

6 RELATED WORK

**Post-processing:** One approach to fairness is to perform a post-processing of the classifier outputs. Examples of previous work in this direction include (Doherty et al., 2012; Feldman, 2015; Hardt et al., 2016). However, this approach of post-processing the outputs to encourage fairness has limited flexibility. (Pleiss et al., 2017) showed that a deterministic solution is only compatible with a single error constraint and thus cannot be applied to fairness notions such as equalized odds. Moreover, decoupling the training and fairness constraint can lead to models with poor accuracy trade-off. In fact (Woodworth et al., 2017) showed that in certain cases, post-processing can be provably suboptimal. Other works discussing the incompatibility of fairness notions include (Chouldechova, 2017; Kleinberg et al., 2016).

**Lagrangian Approach:** There has been much recent work done on enforcing fairness by transforming the constrained optimization problem via the method of Lagrange multipliers. Some works (Zafar et al., 2015; Gol et al., 2016) apply this to the convex setting. In the non-convex case, there is work which frames the constrained optimization problem as a two-player game (Kearns et al. 2017; Agarwal et al. 2018; Cotter et al., 2018b). Related approaches include (Edwards and Storey, 2015; Corbett-Davies et al., 2017; Narasimhan, 2018). There is also recent work similar in spirit which encourages fairness by adding penalties to the objective; e.g. (Donini et al., 2018) studies this for kernel methods and (Komiyama et al., 2018) for linear models. However, the fairness constraints are often irregular and have to be relaxed in order to optimize. Notably, our method does not use the constraints directly in the model loss, and thus does not require them to be relaxed. Moreover, these approaches typically are not readily applicable to equality constraints as feasibility challenges can arise; thus, there is the added challenge of determining appropriate slack during training. Finally, the training can be difficult as (Cotter et al., 2018c) has shown that the Lagrangian may not even have a solution to converge to.

When the classification loss and the relaxed constraints have the same form (e.g. a hinge loss as in (Eban et al., 2017)), the resulting Lagrangian may be rewritten as a cost-sensitive classification, explicitly pointed out in (Agarwal et al., 2018), who show that the Lagrangian method reduces to solving an objective of the form \( \sum_{i=1}^n w_i [h(x_i) \neq y_i] \) for some non-negative weights \( w_i \). In this setting, \( y_i \) may not necessarily be the true label, which may occur for example in demographic parity when the goal is to predict more positively within a protected group and thus may be penalized for predicting correctly on negative examples. In contrast, our approach is a non-negative re-weighting of the original loss (i.e., does not modify the observed labels) and is thus simpler and more aligned with intuition.

**Pre-processing:** This approach has primarily involved massaging the data to remove bias. Examples include (Calders et al., 2009; Kamiran and Calders, 2009; Zliobaite et al., 2011; Zemel et al., 2013; Fish et al., 2015; Feldman et al., 2015; Beutel et al., 2017). Many of these approaches involve changing the labels and features of the training set, which may have legal implications since it is a form of training on falsified data (Barocas and Selbst, 2016). In contrast, our approach does not modify the training data and only reweights the importance of certain sensitive groups. The re-weighting approach is simple and intuitive, thus it is natural that strategies similar to ours have been proposed (Kamiran and Calders, 2012; Krasanakis et al., 2018). These approaches have thus far come with few theoretical guarantees. Our approach is unique for pro-
Figure 2: **Results as λ changes**: We take the optimal λ = λ∗ found by Algorithm 1 and for each value x on the x-axis, we train with data weights based on the setting λ = x · λ∗ and plot the error and violations. When x = 1, we train based on the λ found by Algorithm 1 and get the lowest fairness violation; x = 0 corresponds to training on the unweighted dataset and gives us the lowest prediction error. More charts are in the Appendix.

Providing a mathematically grounded motivation for the reasons for and specific form of re-weighting. The motivation we employ is similar to (Calmon et al., 2017), where the authors propose to modify the data distribution to satisfy some constraints while staying close in a statistical sense to the original data. Our approach is more practical since (1) we focus on a KL-divergence on the labels, which leads to a simple re-weighting strategy, and (2) we provide an automatic method for tuning the modified data distribution (through λ).

7 EXPERIMENTS

7.1 Baselines

We compare results with the following baselines:

**Unconstrained (Unc.)**: This method applies logistic regression on the dataset without any constraints.

**Post-calibration (Cal.)**: This method Hardt et al. (2016) first trains without consideration for fairness, and then determines appropriate thresholds for the protected groups such that fairness is satisfied in training. In the case of overlapping groups, we treat each intersection as their own group.

**Lagrangian approach (Lagr.)** This general method used in several works Eban et al. (2017); Goh et al. (2016); Zafar et al. (2015); Cotter et al. (2018b) proceeds by jointly training the Lagrangian in both model parameters and Lagrange multipliers and uses a hinge approximation of the constraints to make the Lagrangian differentiable in its input. We fix the model learning rate and use ADAM optimizer with learning rate 0.01 and train for 100 epochs. We select a slack for the constraints as without slack, we often converged to degenerate solutions. We chose the smallest slack in increments of 5% until the procedure returned a non-degenerate solution.

**Our method**: We use Algorithm 1 for demographic parity, disparate impact, and equal opportunity and Algorithm 2 for equalized odds (see the appendix). We fix the learning rate η = 1. and number of loops T = 100 across all of our experiments.

For all of the methods except for the Lagrangian, we train using Scikit-Learn’s Logistic Regression Pedregosa et al. (2011) with default hyperparameter settings. We test our method against the unconstrained baseline, post-processing calibration, and the Lagrangian approach with hinge relaxation of the constraints. For all of the methods, we fix the hyperparameters across all experiments.

7.2 Benchmark Fairness Experiments

Our main results are presented in Table 1. We compare our method for learning fair classifiers on a number of benchmark datasets: Bank Marketing Lichman et al. (2013), Communities and Crime Lichman et al. (2013), ProPublica’s COMPAS ProPublica (2018), German Statlog Credit Data Lichman et al. (2013), and Adult Lichman et al. (2013).

We see that our method consistently leads to more fair classifiers, often yielding a classifier with the lowest test violation out of all methods. We also include test error rates in the results. Although the primary objective of these algorithms is to yield a fair classifier, we find that our method is able to find reasonable trade-offs between fairness and accuracy. Our method often provides either better or comparative predictive error than the other fair classification methods (see Figure 2 for more insight into the trade-offs found by our algorithm).

The results in Table 1 also highlight the disadvantages of existing methods for training fair classifiers. Although the post-processing method is an improvement over an unconstrained model, it is often unable to find a classifier with lowest bias.

We also find the results of the Lagrangian approach to not consistently provide fair classifiers. As noted in previous work Cotter et al. (2018c); Goh et al. (2016), constrained optimization can be inherently unstable or requires a certain amount of slack in the objective as the constraints are typically relaxed to make gradient-based training possible and for feasibility purposes. Moreover, due to the added complexity of this method, it can overfit and have poor fairness generalization as
Table 1: **Experiment Results: Benchmark Fairness Tasks**: Each row corresponds to a dataset and fairness notion. We show the accuracy and fairness violation of training with no constraints (Unc.), with post-processing (PP), the Lagrangian approach (Lagr.) and our method. **Bolded** is the method achieving lowest fairness violation for each row. All reported numbers are evaluated on the test set.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bank</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dem. Par.</td>
<td>9.41%</td>
<td>.0349</td>
<td>9.70%</td>
<td>.0068</td>
<td>10.46%</td>
<td>.0126</td>
<td>9.63%</td>
<td>.0056</td>
</tr>
<tr>
<td>Eq. Opp.</td>
<td>9.41%</td>
<td>.1452</td>
<td>9.55%</td>
<td>.0506</td>
<td>9.86%</td>
<td>.1237</td>
<td>9.48%</td>
<td>.0431</td>
</tr>
<tr>
<td>Eq. Odds</td>
<td>9.41%</td>
<td>.1452</td>
<td>N/A</td>
<td>N/A</td>
<td>9.61%</td>
<td>.0879</td>
<td>9.50%</td>
<td>.0376</td>
</tr>
<tr>
<td>Disp. Imp.</td>
<td>9.41%</td>
<td>.0304</td>
<td>N/A</td>
<td>N/A</td>
<td>10.44%</td>
<td>.0135</td>
<td>9.89%</td>
<td>.0063</td>
</tr>
<tr>
<td><strong>COMPAS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dem. Par.</td>
<td>31.49%</td>
<td>.2045</td>
<td>32.33%</td>
<td>.0201</td>
<td>30.16%</td>
<td>.0495</td>
<td>35.44%</td>
<td>.0155</td>
</tr>
<tr>
<td>Eq. Opp.</td>
<td>31.49%</td>
<td>.2373</td>
<td>31.63%</td>
<td>.0256</td>
<td>30.86%</td>
<td>.0566</td>
<td>35.06%</td>
<td>.0663</td>
</tr>
<tr>
<td>Eq. Odds</td>
<td>31.49%</td>
<td>.2373</td>
<td>N/A</td>
<td>N/A</td>
<td>42.69%</td>
<td>.0566</td>
<td>30.65%</td>
<td>.0769</td>
</tr>
<tr>
<td>Disp. Imp.</td>
<td>31.21%</td>
<td>.1362</td>
<td>N/A</td>
<td>N/A</td>
<td>40.35%</td>
<td>.0499</td>
<td>42.64%</td>
<td>.0256</td>
</tr>
<tr>
<td><strong>Comm.</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dem. Par.</td>
<td>11.62%</td>
<td>.4211</td>
<td>32.06%</td>
<td>.0653</td>
<td>28.46%</td>
<td>.0519</td>
<td>30.06%</td>
<td>.0107</td>
</tr>
<tr>
<td>Eq. Opp.</td>
<td>11.62%</td>
<td>.5513</td>
<td>17.64%</td>
<td>.0584</td>
<td>28.45%</td>
<td>.0897</td>
<td>26.85%</td>
<td>.0833</td>
</tr>
<tr>
<td>Eq. Odds</td>
<td>11.62%</td>
<td>.5513</td>
<td>N/A</td>
<td>N/A</td>
<td>28.46%</td>
<td>.0962</td>
<td>26.65%</td>
<td>.0769</td>
</tr>
<tr>
<td>Disp. Imp.</td>
<td>14.83%</td>
<td>.3960</td>
<td>N/A</td>
<td>N/A</td>
<td>28.26%</td>
<td>.0557</td>
<td>30.26%</td>
<td>.0073</td>
</tr>
<tr>
<td><strong>German</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dem. Par.</td>
<td>24.85%</td>
<td>.0766</td>
<td>24.85%</td>
<td>.0346</td>
<td>25.45%</td>
<td>.0410</td>
<td>25.15%</td>
<td>.0137</td>
</tr>
<tr>
<td>Eq. Opp.</td>
<td>24.85%</td>
<td>.1120</td>
<td>24.54%</td>
<td>.0922</td>
<td>27.27%</td>
<td>.0757</td>
<td>25.45%</td>
<td>.0662</td>
</tr>
<tr>
<td>Eq. Odds</td>
<td>24.85%</td>
<td>.1120</td>
<td>N/A</td>
<td>N/A</td>
<td>34.24%</td>
<td>.1318</td>
<td>25.45%</td>
<td>.1099</td>
</tr>
<tr>
<td>Disp. Imp.</td>
<td>24.85%</td>
<td>.0608</td>
<td>N/A</td>
<td>N/A</td>
<td>27.57%</td>
<td>.0468</td>
<td>25.15%</td>
<td>.0156</td>
</tr>
<tr>
<td><strong>Adult</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dem. Par.</td>
<td>14.15%</td>
<td>.1173</td>
<td>16.60%</td>
<td>.0129</td>
<td>20.47%</td>
<td>.0198</td>
<td>16.51%</td>
<td>.0037</td>
</tr>
<tr>
<td>Eq. Opp.</td>
<td>14.15%</td>
<td>.1195</td>
<td>14.43%</td>
<td>.0170</td>
<td>19.67%</td>
<td>.0374</td>
<td>14.46%</td>
<td>.0092</td>
</tr>
<tr>
<td>Eq. Odds</td>
<td>14.15%</td>
<td>.1195</td>
<td>N/A</td>
<td>N/A</td>
<td>19.04%</td>
<td>.0160</td>
<td>14.58%</td>
<td>.0221</td>
</tr>
<tr>
<td>Disp. Imp.</td>
<td>14.19%</td>
<td>.1108</td>
<td>N/A</td>
<td>N/A</td>
<td>20.48%</td>
<td>.0199</td>
<td>17.37%</td>
<td>.0334</td>
</tr>
</tbody>
</table>

Table 2: **MNIST with Label Bias**

<table>
<thead>
<tr>
<th>Method</th>
<th>Test Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trained on True Labels</td>
<td>97.85%</td>
</tr>
<tr>
<td>Unconstrained</td>
<td>88.18%</td>
</tr>
<tr>
<td>Post-processing</td>
<td>89.79%</td>
</tr>
<tr>
<td>Lagrangian</td>
<td>94.05%</td>
</tr>
<tr>
<td>Our Method</td>
<td><strong>97.00%</strong></td>
</tr>
</tbody>
</table>

noted in Cotter et al. (2018a). Accordingly, we find that the Lagrangian method often yields poor trade-offs in fairness and accuracy, at times yielding classifiers with both worse accuracy and more bias.

### 7.3 MNIST with Label Bias

We investigate the behavior of our method on a variant of the well-known MNIST task. We take the MNIST dataset under the standard train/test split and then randomly select 20% of the training data points and change their label to 2, yielding a biased set of labels. On such a dataset, our method should be able to find appropriate weights so that training on the weighted dataset roughly corresponds to training on the true labels. To this end, we train a classifier with a demographic-parity-like constraint on the predictions of digit 2; i.e., we encourage a classifier to predict the digit 2 at a rate of 10%, the rate appearing in the true labels. We compare to the same baseline methods as before.

We present the results in Table 2. We report test accuracy computed with respect to the true labels. We find that our method is the only one that is able to approach the accuracy of a classifier trained with respect to the true labels. Both unconstrained training or post-processing provide little improvement, with error rates greater than 10%. Compared to the next best method (the Lagrangian), our proposed technique improves error rate by roughly half. These results give further evidence of our method’s ability to effectively train on the true labels despite only observing biased labels.

### 8 CONCLUSION

We presented a new framework to model how bias can arise in a dataset, assuming that there exists an unbiased ground truth. Our method for correcting for this bias is based on re-weighting the training examples. Given the appropriate weights, we showed with finite-sample guarantees that the learned classifier will be approximately unbiased. We gave practical procedures which approximate these weights and showed that the resulting algorithm leads to fair classifiers in a variety of settings.
Acknowledgements

We thank Maya Gupta, Andrew Cotter and Harikrishna Narasimhan for many insightful discussions and suggestions as well as Corinna Cortes for helpful comments.

References


Identifying and Correcting Label Bias in Machine Learning


