Decoupled Classifiers for Group-Fair and Efficient Machine Learning

Cynthia Dwork DWORK@SEAS.HARVARD.EDU
Harvard University

Nicole Immorlica NICIMM@MICROSOFT.COM
Microsoft Research New England

Adam Tauman Kalai ADAM.KALAI@MICROSOFT.COM
Microsoft Research New England

Max Leiserson MDML@CS.UMD.EDU
University of Maryland

Editors: Sorelle A. Friedler and Christo Wilson

Abstract

When it is ethical and legal to use a sensitive attribute (such as gender or race) in machine learning systems, the question remains how to do so. We show that the naïve application of machine learning algorithms using sensitive attributes leads to an inherent tradeoff in accuracy between groups. We provide a simple and efficient *decoupling* technique, which can be added on top of any black-box machine learning algorithm, to learn different classifiers for different groups. Transfer learning is used to mitigate the problem of having too little data on any one group.

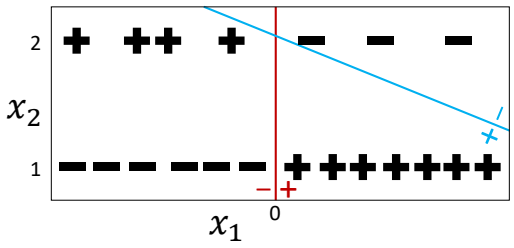

Figure 1: No linear classifiers can achieve greater than 50% accuracy on both groups.

1. Introduction

As algorithms are increasingly used to make decisions of social consequence, the social values encoded in these decision-making procedures are the subject of increasing study, with fairness being a chief concern (Pedreschi et al., 2008; Zliobaite et al., 2011; Kamishima et al., 2011; Dwork et al., 2011; Friedler et al., 2016; Angwin et al., 2016; Chouldechova, 2017; Kleinberg et al., 2016; Hardt et al., 2016; Joseph et al., 2016; Kusner et al., 2017; Berk, 2009). *Classification and regression algorithms* are one particular locus of fairness concerns. Classifiers map individuals to outcomes: applicants to accept/reject/waitlist; adults to credit scores; web users to advertisements; felons to estimated recidivism risk. Informally, the concern is whether individuals are treated "fairly," however this is defined. Still speaking informally, there are many sources of unfairness, prominent among these being training the classifier on historically biased data and a paucity of data for under-represented groups leading to poor performance on these groups, which in turn can lead to higher risk for those, such as lenders, making decisions based on classification outcomes.

Should ML systems use sensitive attributes, such as gender or race if available? The legal and ethical factors behind such a decision vary by time, country, jurisdiction, culture, and downstream application. Still speaking informally, it is known that "ignoring" these attributes does not ensure fairness, both because they may be closely correlated with other features in the data and because they provide context for understand-

ing the rest of the data, permitting a classifier to incorporate information about cultural differences between groups (Dwork et al., 2011). Using sensitive attributes may increase accuracy for all groups and may avoid biases where a classifier favors members of a minority group that meet criteria optimized for a majority group, as illustrated visually in Figure 4 of 8.

In this paper, we consider *how* to use a sensitive attribute such as gender or race to maximize fairness and accuracy, assuming that it is legal and ethical. A data scientist wishing to fit, say, a simple linear classifier, may use the raw data, upweight/oversample data from minority groups, or employ advanced approaches to fitting linear classifiers that aim to be accurate and fair. No matter what he does and what fairness criteria he uses, assuming no linear classifier is perfect, he may be faced with an inherent tradeoff between accuracy on one group and accuracy on another. As an extreme illustrative example, consider the two group setting illustrated in Figure 1, where feature x_1 perfectly predicts the binary outcome $y \in \{-1, 1\}$. For people in group 1 (where $x_2 = 1$), the majority group, $y = \text{sgn}(x_1)$, i.e., $y = 1$ when $x_1 > 0$ and -1 otherwise. However, for the minority group where $x_2 = 2$, exactly the opposite holds: $y = -\text{sgn}(x_1)$. Now, if one performed classification without the sensitive attribute x_2, the most accurate classifier predicts $y = \text{sgn}(x_1)$, so the majority group would be perfectly classified and the minority group would be classified as inaccurately as possible. However, even using the group membership attribute x_2, it is impossible to simultaneously achieve better than 50% (random) accuracy on both groups. This is due to limitations of a linear classifier $\text{sgn}(w_1x_1 + w_2x_2 + b)$, since the same w_1 is used across groups.

In this paper we define and explore *decoupled* classification systems, in which a separate[1] classifier is trained on each group. Training a classifier involves minimizing a loss function that penalizes errors; examples include mean squared loss and absolute loss. In decoupled classification systems one first obtains, for each group separately, a collection of classifiers differing in the numbers of *positive* classifications returned for the members of the given group. Let this set of outputs for group k be denoted C_k, $k = 1, \ldots, K$. The output of the decoupled training step is an element of $C_1 \times \ldots \times C_K$, that is, a single classifier for each group. The output is chosen to minimize a *joint loss function* that can penalize differences in classification statistics between groups. Thus the loss function can capture *group fairness* properties relating the treatment of different groups, *e.g.*, the false positive (respectively, false negative) rates are the same across groups; the demographics of the group of individuals receiving positive (negative) classification are the same as the demographics of the underlying population; the positive predictive value is the same across groups.[2] By pinning down a specific objective, the modeler is forced to make explicit the tradeoff between accuracy and fairness, since often both cannot simultaneously be achieved. Finally, a generalization argument relates fairness properties, captured by the joint loss on the training set, to similar fairness properties on the distribution from which the data were drawn. We broaden our results so as to enable the use of *transfer learning* to mitigate the problems of low data volume for minority groups.

The following observation provides a property essential for efficient decoupling. A *profile* is a vector specifying, for each group, a number of positively classified examples from the training set. For a given profile $(p_1, \ldots, p_K)$, the most accurate classifier also simultaneously minimizes the false positives and false negatives. *It is the choice of profile that is determined by the joint loss criterion.* We show that, as long as the joint loss function satisfies a weak form of *monotonicity*, one can use off-the-shelf classifiers to find a decoupled solution that minimizes joint loss.

The monotonicity requirement is that the joint loss is non-decreasing in error rates, for any fixed profile. This sheds some light on the thought-provoking impossibility results of Chouldechova (2017) and Kleinberg et al. (2016) on the impossibility of simultaneously achieving three specific

1. In the case of linear classifiers, training separate classifiers is equivalent to adding interaction terms between the sensitive attributes and all other attributes. More generally, the separate classifiers can equivalently be thought of as a single classifier that branches on the group attribute. The decoupling technique is a simple way to add branching to any type of classifier.

2. In contrast *individual fairness* Dwork et al. (2011) requires that *similar people are treated similarly*, which requires a task-specific, culturally-aware, similarity metric.

notions of group fairness (see Observation 1 in Section 4.1).

Finally, we present experiments on 47 datasets downloaded from `http://openml.org`. The experiments are "semi-synthetic" in the sense that the first binary feature was used as a substitute sensitive feature since we did not have access to sensitive features. We find that on many data sets our algorithm improves performance while much less often decreasing performance.

Remark. The question of whether or not to use decoupled classifiers is orthogonal to our work, which explores the mathematics of the approach, and a comprehensive treatment of the pros and cons is beyond our expertise. Most importantly, we emphasize that decoupling, together with a "poor" choice of joint loss, could be used unfairly for discriminative purposes. Furthermore, in some jurisdictions using a different classification method, or even using different weights on attributes for members of demographic groups differing in a protected attribute, is illegal for certain classification tasks, *e.g.* hiring. Even barring legal restrictions, the assumption that group membership is an input bit is an oversimplification, and in reality the information may be obscured, and the definition of the groups may be ambiguous at best. Logically pursuing the idea behind the approach it is not clear which intersectionalities to consider, or how far to subdivide. Nonetheless, we believe decoupling is valuable and applicable in certain settings and thus merits investigation.

The contributions of this work are: (a) showing how, when using sensitive attributes, the straightforward application of many machine learning algorithms will face inherent tradeoffs between accuracy across different groups, (b) introducing an efficient decoupling procedure that outputs separate classifiers for each class using transfer learning, (c) modeling fair and accurate learning as a problem of minimizing a joint loss function, and (d) presenting experimental results showing the applicability and potential benefit of our approach.

1.1. Related Work

Group fairness has a variety of definitions, including conditions of *statistical parity*, *class balance* and *calibration*. In contrast to individual fairness, these conditions constrain, in various ways, the dependence of the classifier on the sensitive attributes. The statistical parity condition requires that the assigned label of an individual is independent of sensitive attributes. The condition formalizes the legal doctrine of disparate impact imposed by the Supreme Court in Griggs v Duke Power Company. Statistical parity can be approximated by either modifying the data set or by designing classifiers subject to fairness regularizers that penalize violations of statistical parity (see Feldman et al. (2015) and references therein). Dwork et al. (2011) propose a "fair affirmative action" methodology that carefully relaxes between-group individual fairness constraints in order to achieve group fairness. Zemel et al. (2013) introduce a representational approach that attempts to "forget" group membership while maintaining enough information to classify similar individuals similarly; this approach also permits generalization to unseen data points. To our knowledge, the earliest work on trying to learn fair classifiers from historically biased data is by Pedreschi et al. (2008); see also (Zliobaite et al., 2011) and (Kamishima et al., 2011).

The class-balanced condition (called *error-rate balance* by Chouldechova (2017) or *equalized odds* by Hardt et al. (2016)), similar to statistical parity, requires that the assigned label is independent of sensitive attributes, but only *conditional on the true classification of the individual*. For binary classification tasks, a class-balanced classifier results in equal false positive and false negative rates across groups. One can also modify a given classifier to be class-balanced while minimizing loss by adding label noise (Hardt et al., 2016).

The well-calibrated condition requires that, conditional on their label, an equal fraction of individuals from each group have the same true classification. A well-calibrated classifier labels individuals from different groups with equal accuracy. Hebert-Johnson et al. (2017) extend calibration to *multi-calibration* which requires the classifier to be well calibrated on a collection of sets of individuals, eg, all those described by circuits of a given size. The class-balanced solution (Hardt et al., 2016) also fails to be well-calibrated. Chouldechova (2017) and Kleinberg et al. (2016) independently showed that, except

in cases of perfect predictions or equal base rates of true classifications across groups, there is no class-balanced and well-calibrated classifier.

A number of recent works explore causal approaches to defining and detecting (un)fairness (Nabi and Shpitser, 2017; Kusner et al., 2017; Bareinboim and Pearl, 2016; Kilbertus et al., 2017). See the beautiful primer of Pearl et al. (2016) for an introduction to the central concepts and machinery.

Finally, we mention that sensitive attributes are used in various real-world systems. As one example, Hassidim et al. (2017) describe using such features in an admissions matching system for masters students in Israel.

2. Preliminaries

Let $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2 \cup \ldots \cup \mathcal{X}_K$ be the set of possible *examples* partitioned by group. The set of possible *labels* is $\mathcal{Y}$ and the set of possible *classifications* is $\mathcal{Z}$. A *classifier* is a function $c : \mathcal{X} \to \mathcal{Z}$. We assume that there is a fixed family $\mathcal{C}$ of classifiers. For simplicity, we restrict our analysis the case of binary classification $\mathcal{Y} = \mathcal{Z} = \{0, 1\}$, but many of the results extended directly to regression or randomized classification $\mathcal{Y}, \mathcal{Z} \subseteq \mathbb{R}$.

We suppose that there is a joint distribution $\mathcal{D}$ over labeled examples $x, y \in \mathcal{X} \times \mathcal{Y}$ and we have access to n training examples $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$ drawn independently from $\mathcal{D}$. We denote by $g(x)$ the group number to which x belongs and $g_i = g(x_i)$, so $x_i \in \mathcal{X}_{g_i}$.

Finally, as is common, we consider the loss $\ell_{\mathcal{D}}(c) = \mathrm{E}_{x,y\sim\mathcal{D}}[\ell(y, c(x))]$ for an application-specific loss function $\ell : \mathcal{Y} \times \mathcal{Z} \to \mathbb{R}$ where $\ell(y, z)$ accounts for the cost of classifying as z an example whose true label is y. The group-k loss for $\mathcal{D}, c$ is defined to be $\ell_{\mathcal{D}k}(c) = \mathrm{E}_{\mathcal{D}}[\ell(y, c(x))|x \in \mathcal{X}_k]$ or 0 if $\mathcal{D}$ assigns 0 probability to $\mathcal{X}_k$. The standard approach in ML is to minimize $\ell_{\mathcal{D}}(c)$ over $c \in \mathcal{C}$. Common loss functions include the L_1 loss $\ell(y, z) = |y - z|$ and L_2 loss $\ell(y, z) = (y - z)^2$. In Section 4, we provide a methodology for incorporating a range of fairness notions into loss.

3. Decoupling and the cost of coupling

For a vector of K classifiers, $\boldsymbol{c} = (c_1, c_2, \ldots, c_K)$, the decoupled classifier $\gamma_{\boldsymbol{c}} : \mathcal{X} \to \mathcal{Z}$ is defined to be $\gamma_{\boldsymbol{c}}(x) = c_{g(x)}(x)$. The set of decoupled classifiers is denoted $\gamma(\mathcal{C}) = \{\gamma_{\boldsymbol{c}} \mid \boldsymbol{c} \in \mathcal{C}^K\}$. Some classifiers, such as decision trees of unbounded size over $\mathcal{X} = \{0, 1\}^d$, are already decoupled, i.e., $\gamma(\mathcal{C}) = \mathcal{C}$. As we shall see, however, in high dimensions common families of classifiers in use are coupled to avoid the curse of dimensionality.

The cost of coupling of a family $\mathcal{C}$ of classifiers (with respect to ℓ) is defined to be the worst-case maximum of the difference between the loss of the most accurate coupled and decoupled classifiers over distributions $\mathcal{D}$.

$$\text{cost-of-coupling}(\mathcal{C}, \ell) =$$

$$\max_{\mathcal{D}\in\Delta(\mathcal{X}\times\mathcal{Y})} \left[\min_{c\in\mathcal{C}} \ell_{\mathcal{D}}(c) - \min_{\gamma_{\boldsymbol{c}}\in\gamma(\mathcal{C})} \ell_{\mathcal{D}}(\gamma_{\boldsymbol{c}})\right].$$

Here $\Delta(S)$ denotes the set of probability distributions over set S. To circumvent measure-theoretic nuisances, we require $\mathcal{C}, \mathcal{X}, \mathcal{Y}$ to be finite sets. Note that numbers on digital computers are all represented using a fixed-precision (bounded number of bits) representation, and hence all these sets may be assumed to be of finite (but possibly exponentially large) size.

We now show that the cost of coupling is related to fairness across groups.

Lemma 1 *Suppose cost-of-coupling$(\mathcal{C}, \ell) = \not\epsilon$. Then there is a distribution $\mathcal{D}$ such that no matter which classifier $c \in \mathcal{C}$ is used, there will always be a group k and a classifier $c' \in \mathcal{C}$ whose group-k loss is at least $\not\epsilon$ smaller than that of c, i.e., $\ell_{\mathcal{D}k}(c') \leq \ell_{\mathcal{D}k}(c) - \not\epsilon$.*

Proof Let $\gamma_{\boldsymbol{c}'}$ be a decoupled classifier with minimal loss where $\boldsymbol{c}' = (c'_1, \ldots, c'_K)$. This loss is a weighted average (weighted by demography) of the average loss on each group. Hence, for any c, there must be some group k on which the loss of c'_k is $\not\epsilon$ less than that of c. ■

Hence, if the cost of coupling is positive, then the learning algorithm that selects a classifier faces an inherent tradeoff in accuracy across groups. The following theorem shows that the cost of coupling is large (a constant) for linear classifiers and

decision trees; similar arguments exist for other common classifiers. All remaining proofs are deferred to the full version.

Theorem 2 *Fix $\mathcal{X} = \{0,1\}^d$, $\mathcal{Y} = \{0,1\}$, and $K = 2$ groups (encoded by the last bit of x). Then the cost of coupling is at least 1/4 for:*

1. **Linear regression***: $\mathcal{Z} = \mathbb{R}$, $\mathcal{C} = \{w \cdot x + b \mid w \in \mathbb{R}^d, b \in \mathbb{R}\}$, and $\ell(y,z) = (y-z)^2$*
2. **Linear separators***: $\mathcal{Z} = \{0,1\}$, $\mathcal{C} = \{\mathbb{I}[w \cdot x + b \geq 0] \mid w \in \mathbb{R}^d, b \in \mathbb{R}\}$, and $\ell(y,z) = |y-z|$*
3. **Bounded-size decision trees***: For $\mathcal{Z} = \{0,1\}$, $\mathcal{C}$ being the set of binary decision trees of size $\leq 2^s$ leaves, and $\ell(y,z) = |y-z|$*

We note that it is straightforward to extend the above theorem to generalized linear models, i.e., functions $c(x) = u(w \cdot x)$ for monotonic functions $u : \mathbb{R} \to \mathbb{R}$, which includes logistic regression as one common special case. It is also possible, though more complex, to provide a lower bound on the cost of coupling of neural networks, regression forests, or other complex families of functions of bounded representation size s. In order to do so, one needs to simply show that the size-s functions are sufficiently rich in that there are two different size-s classifiers $\boldsymbol{c} = (c_1, c_2)$ such that $\gamma_{\boldsymbol{c}}$ has 0 loss (say over the uniform distribution on $\mathcal{X}$) but that every single size-s classifier has significant loss.

4. Joint loss and monotonicity

As discussed, the classifications output by an ML classifier are often evaluated by their empirical loss $\frac{1}{n}\sum_i \ell(y_i, z_i)$. To account for fairness, we generalize loss to joint classifications across groups. In particular, we consider an application-specific *joint loss* $\hat{L} : ([K] \times \mathcal{Y} \times \mathcal{Z})^* \to \mathbb{R}$ that assigns a cost to a set of classifications, where $[K] = \{1, 2, \ldots, K\}$ indicates the group number for each example. A joint loss might be, for parameter $\lambda \in [0,1]$:

$$\hat{L}\big(\langle g_i, y_i, z_i\rangle_{i=1}^n\big) =$$

$$\frac{\lambda}{n}\sum_{i=1}^{n}|y_i - z_i| + \frac{1-\lambda}{n}\sum_{k=1}^{K}\left|\sum_{i:g_i=k} z_i - \frac{1}{K}\sum_i z_i\right|.$$

The above $\hat{L}$ trades off accuracy for differences in number of positive classifications across groups. For $\lambda = 1$, this is simply L_1 loss, while for $\lambda = 0$, the best classifications would have an equal number of positives in each group. Joint loss differs from a standard ML loss function in two ways. First, joint loss is aware of the sensitive group membership. Second, it depends on the complete labelings and is not simply a sum over labels.

Even with only $K = 1$ group, this captures situations beyond what is representable by the sum $\sum \ell(y_i, z_i)$. A simple example is when one seeks exactly P positive examples:

$$\hat{L}\big(\langle g_i, y_i, z_i\rangle_{i=1}^n\big) = \begin{cases} \frac{1}{n}\sum|y_i - z_i| & \text{if } \sum z_i = P \\ 1 & \text{otherwise.} \end{cases}$$

Since $\frac{1}{n}\sum|y_i - z_i| \leq 1$, the 1 ensures that the loss minimizer will have exactly P positives, if such a classifier exists in $\mathcal{C}$ for the data.

In this section, we denote joint loss $\hat{L}$ with the hat notation indicating that it is an empirical approximation. In the next section we will define joint loss L for distributions. We denote classifications by z_i rather than the standard notation $\hat{y}_i$ which suggests predictions, because, as in the above loss, one may choose classifications $z \neq y$ even with perfect knowledge of the true labels.

For the remainder of our analysis, we henceforth consider binary labels and classifications, $\mathcal{Y} = \mathcal{Z} = \{0,1\}$. Our approach is general, however, and our experiments include regression. For a given $\langle x_i, y_i, z_i\rangle_{i=1}^n$, and for any group $k \leq K$ and all $(y,z) \in \{0,1\}^2$, recall that the groups are $g_i = g(x_i)$ and define:

$$\textit{counts}: \quad n_k = \big|\{i \mid g_i = k\}\big| \in \{1, 2, \ldots, n\}$$

$$\textit{profile}: \quad \hat{p}_k = \frac{1}{n}\sum_{i:g_i=k} z_i \in [0, n_k/n]$$

$$\textit{group losses}: \quad \hat{\ell}_k = \frac{1}{n_k}\sum_{i:g_i=k}|z_i - y_i| \in [0,1]$$

Note that the normalization is such that the standard 0-1 loss is $\sum_k \frac{n_k}{n}\hat{\ell}_k$ and the fraction of positives within any class is $\frac{n}{n_k}\hat{p}_k$.

We note many studied fairness notions, including numerical parity, demographic parity, and false-negative-rate parity can be represented in a joint loss function. For example, demographic

parity is:

$$\lambda \hat{L}_1 + (1-\lambda) \sum_k \left| \hat{p}_k \frac{n}{n_k} - \frac{1}{K} \sum_{k'} \hat{p}_{k'} \frac{n}{n_{k'}} \right|.$$

In many applications there is a different cost for false positives where $(y, z) = (0, 1)$ and false negatives where $(y, z) = (1, 0)$. The fractions of false positives and negatives are defined, below, for each group k. They can be computed based on the fraction of positive labels in each group π_k:

$$\pi_k = \frac{1}{n_k} \sum_{i:g_i=k} y_i$$

$$\mathrm{FP}_k = \frac{1}{n_k} \sum_{i:g_i=k} z_i(1-y_i) = \frac{\hat{\ell}_k + \hat{p}_k \frac{n}{n_k} - \pi_k}{2} \tag{1}$$

$$\mathrm{FN}_k = \frac{1}{n_k} \sum_{i:g_i=k} (1-z_i)y_i = \frac{\hat{\ell}_k + \pi_k - \hat{p}_k \frac{n}{n_k}}{2}, \tag{2}$$

While minimizing group loss $\hat{\ell}_k = \mathrm{FP}_k + \mathrm{FN}_k$ in general does not minimize false positives or false negatives on their own, the above implies that for a fixed profile $\hat{p}_k$, the most accurate classifier on group k simultaneously minimizes false positives and false negatives. The above can be derived by adding or subtracting the equations $\hat{\ell}_k = \mathrm{FP}_k + \mathrm{FN}_k$ (since every error is a false positive or a false negative) and $\frac{n}{n_k}\hat{p}_k = \mathrm{FP}_k + (\pi_k - \mathrm{FN}_k)$ (since every positive classification is either a false positive or true positive, and the fraction of true positives from group k are $\pi_k - \mathrm{FN}_k$). We also define the *false negative rate* $\mathrm{FNR}_k = \mathrm{FN}_k/\pi_k$. False positive rates can be defined similarly.

Equations (1) and (2) imply that, if one desires fewer false positives and false negatives (all other things being fixed), then greater accuracy is better. That is, for a fixed profile, the most accurate classifier simultaneously minimizes false positives and false negatives. This motivates the following monotonicity notion.

Definition 3 (Monotinicity) *Joint loss $\hat{L}$ is monotonic if, for any fixed $\langle g_i, y_i \rangle_{i=1}^n \in ([K] \times \mathcal{Y})^*$, $\hat{L}$ can be written as $c(\langle \hat{\ell}_k, \hat{p}_k \rangle_{k=1}^K)$ where $c : [0,1]^{2K} \to \mathbb{R}$ is a function that is nondecreasing in each $\hat{\ell}_k$ fixing all other inputs to c.*

That is, for a fixed profile, increasing $\hat{\ell}_k$ can only increase joint loss. To give further intuition behind monotonicity, we give two other equivalent definitions.

Definition 4 (Monotonicity) *Joint loss $\hat{L}$ is monotonic if, for any $\langle g_i, y_i, z_i \rangle_{i=1}^n \in ([K] \times \mathcal{Y} \times \mathcal{Z})^*$, and any i, j where $g_i = g_j$, $y_i \le y_j$ and $z_i \le z_j$: swapping z_i and z_j can only increase loss, i.e.,*

$$\hat{L}(\langle g_i, y_i, z_i \rangle_{i=1}^n) \le \hat{L}(\langle g_i, y_i, z'_i \rangle_{i=1}^n),$$

where z' is the same as z except $z'_i = z_j$ and $z'_j = z_i$.

We can see that if $y_i = y_j$ then swapping z_i and z_j does not change the loss (because the condition can be used in either order). This means that the loss is "semi-anonymous" in the sense that it only depends on the numbers of true and false positives and negatives for each group. The more interesting case is when $(y_i, y_j) = (0, 1)$ where it states that the loss when $(z_i, z_j) = (0, 1)$ is no greater than the loss when $(z_i, z_j) = (1, 0)$. Finally, monotonicity can also be defined in terms of false positives and false negatives.

Definition 5 (Monotonicity) *Joint loss $\hat{L}$ is monotonic if, for any $\langle g_i, y_i, z_i \rangle_{i=1}^n \in ([K] \times \mathcal{Y} \times \mathcal{Z})^*$, and any alternative classifications $z'_1, \ldots, z'_n$ such that, in each group k, the same profile as z but all smaller or equal false positive rates* and *all smaller or equal false negative rates, the loss of classifications z'_i is no greater than that of z_i.*

Lemma 6 *Definitions 3, 4, and 5 of Monotonicity are equivalent.*

One may be tempted to consider a simpler notion of monotonicity, such as requiring the loss with $z_i = y_i$ to be no greater than that of $z_i = 1 - y_i$, fixing everything else. However, this would rule out many natural monotonic joint losses $\hat{L}$, such as demographic parity.

4.1. Discussion: fairness metrics versus objectives

The monotonicity requirement admits a range of different fairness criteria, but not all. We do not mean to imply that monotonicity is necessary for fairness, but rather to discuss the implications of

minimizing a non-monotonic loss objective. The following example helps illustrate the boundary between monotonic and non-monotonic.

Observation 1 *Fix $K = 2$. The following joint loss is monotonic if and only if $\lambda \leq 1/2$:*

$$(1-\lambda)(\hat{\ell}_1 + \hat{\ell}_2) + \lambda|\hat{\ell}_1 - \hat{\ell}_2|.$$

The loss in the above lemma trades off accuracy for differences in loss rates between groups. What we see is that monotonic losses can account, to a limited extent, for differences across groups in fractions of errors, and related statements can be made for combinations of rates of false positive and false negative, inspired by "equal odds" definitions of fairness. However, when the weight λ on the fairness term exceeds $1/2$, then the loss is non-monotonic and one encounters situations where one group is punished with lower accuracy in the name of fairness. This may still be desirable in a context where equal odds is a primary requirement, and one would rather have random classifications (e.g., a lottery) than introduce any inequity.

What is the goal of an objective function? We argue that a good objective function is one whose optimization leads to favorable outcomes, and should not be confused with a fairness metric whose goal is *quantify* unfairness. Often, a different function is appropriate for quantifying unfairness than for optimizing it. For example, the difference in classroom performance across groups may serve as a good metric of unfairness, but it may not be a good objective on its own. The root cause of the unfairness may have begun long before the class. Now, suppose that the objective from the above observation was used by a teacher to design a semester-long curriculum with the best intention of increasing the minority group's performance to the level of the majority. If there is no curriculum that in one semester increases one group's performance to the level of another group's performance, then optimizing the above loss for $\lambda > 1/2$ leads to an undesirable outcome: the curriculum would be chosen so as to intentionally *mis*teach students the higher-performing group of students so that their loss increases to match that of the other group. This can be see by rewriting the loss as follows:

$$\begin{aligned}&(1-\lambda)(\hat{\ell}_1 + \hat{\ell}_2) + \lambda|\hat{\ell}_1 - \hat{\ell}_2| \\ &= 2\lambda \max\{\hat{\ell}_1, \hat{\ell}_2\} + (1-2\lambda)(\hat{\ell}_1 + \hat{\ell}_2).\end{aligned}$$

This rewriting illuminates why $\lambda \leq 1/2$ is necessary for monotonicity, otherwise there is a negative weight on the total loss. $\lambda = 1/2$ corresponds to maximizing the minimum performance across groups while $\lambda = 0$ means teaching to the average, and λ in between allows interpolation. However, putting too much weight on fairness leads to undesirable punishing behavior.

5. Minimizing joint loss on training data

Here, we show how to use learning algorithm to find a decoupled classifier in $\gamma(\mathcal{C})$ that is optimal on the training data. In the next section, we show how to generalize this to imperfect randomized classifiers that generalize to examples drawn from the same distribution, potentially using an arbitrary transfer learning algorithm.

Our approach to decoupling uses a learning algorithm for $\mathcal{C}$ as a black box. A *$\mathcal{C}$-learning algorithm* $A : (\mathcal{X} \times \mathcal{Y})^* \to 2^{\mathcal{C}}$ returns one or more classifiers from $\mathcal{C}$ with differing numbers of positive classifications on the training data, i.e., for any two distinct $c, c' \in A\big(\langle x_i, y_i\rangle_{i=1}^n\big)$, $\sum_i c(x_i) \neq \sum_i c'(x_i)$. In ML, it is common to simultaneously output classifiers with varying number of positive classifications, e.g., in computing ROC or precision-recall curves (Davis and Goadrich, 2006). Also note that a classifier that purely minimizes errors can be massaged into one that outputs different fractions of positive and negative examples by reweighting (or subsampling) the positive- and negative-labeled examples with different weights.

Our analysis will be based on the assumption that the classifier is in some sense optimal, but importantly note that it makes sense to apply the reduction to any off-the-shelf learner. Formally, we say A is *optimal* if for every achievable number of positives $P \in \big\{\sum_i c(x_i) \;\big|\; c \in \mathcal{C}\big\}$, it outputs exactly one classifier that classifies exactly P positives, and this classifier has minimal error among all classifiers which classify exactly P positives. Theorem 7 shows that an optimal classifier can be used to minimize any (monotonic) joint loss

Theorem 7 *For any monotonic joint loss function $\hat{L}$, any $\mathcal{C}$, and any optimal learner A for $\mathcal{C}$,*

Algorithm 1: Decouple $(A, \hat{L}, \{\langle x_i, y_i\rangle\}, \{\mathcal{X}_i\})$

Minimize training loss $\hat{L}$ using learner A

1. For $k = 1$ to K, $C_k \leftarrow A\big(\langle x_i, y_i\rangle_{i:x_i\in\mathcal{X}_k}\big)$
   ▷ Learner outputs a set of classifiers.
2. return $\gamma_{\boldsymbol{c}}$ that minimizes $\min_{\boldsymbol{c}\in C_1\times\ldots\times C_K} \hat{L}\big(\langle g_i, y_i, \gamma_{\boldsymbol{c}}(x_i)\rangle_{i=1}^n\big)$
   ▷ $\gamma_{\boldsymbol{c}}(x_i) = c_{g_i}(x))$

The simple decoupling algorithm partitions data by group and runs the learner on each group. Within each group, the learner outputs one or more classifiers of differing numbers of positives.

the Decouple *procedure from Algorithm 1 returns a classifier in $\gamma(\mathcal{C})$ of minimal joint loss $\hat{L}$. For constant K,* Decouple *runs in time linear in the time to run A and polynomial in the number of examples n and time to evaluate $\hat{L}$ and classifiers $c \in \mathcal{C}$.*

Implementation notes. Note that if the profile is fixed, as in $\hat{L}_{\boldsymbol{p}^*}$, then one can simply run the learning algorithm once for each group, targeted at p_k^* positives in each group. Otherwise, also note that to perform the slowest step which involves searching over $O(n^K)$ losses of combinations of classifiers, one can pre-compute the error rates and profiles of each classifier. In the "big data" regime of very large n, the $O(n^K)$ evaluations of a simple numeric function of profile and losses will not be the rate limiting step.

6. Generalization and transfer learning

We now turn to the more general randomized classifier model in which $\mathcal{Z} = [0, 1]$ but still with $\mathcal{Y} = \{0, 1\}$, and we also consider generalization loss as opposed to simply training loss. We will define loss in terms of the underlying joint distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ from which training examples are drawn independently. We define the *true* error, *true profile*, and true probability:

$$\nu_k = \Pr[x \in \mathcal{X}_k] = \mathrm{E}[n_k/n]$$
$$p_k = \mathrm{E}\big[z\mathbb{I}[x \in \mathcal{X}_k]\big] = \mathrm{E}[\hat{p}_k]$$
$$\ell_k = \mathrm{E}\big[|y - z| \mid x \in \mathcal{X}_k\big] = \mathrm{E}[\hat{\ell}_k | n_k > 0]$$

Joint loss L is defined on the joint distribution μ on $g, y, z \in [K] \times \mathcal{Y} \times \mathcal{Z}$ induced by $\mathcal{D}$ and a classifier $c : \mathcal{X} \to \mathcal{Z}$. A distributional joint loss L is related to empirical joint loss $\hat{L}$ in that $L = \lim_{n\to\infty} \mathrm{E}[\hat{L}]$, i.e., the limit of the empirical joint loss as the number of training data grows without bound (if it exists).

Fixing the marginal distribution over $[K] \times \mathcal{Y}$, joint loss $L : [0, 1]^{2K} \to \mathbb{R}$ can be viewed as a function of $\ell_1, p_1, \ldots, \ell_K, p_K$ (in addition to group probabilities $\Pr[g(x) = k]$ which are independent of the classification). In addition to requiring monotonicity, namely L being nondecreasing in ℓ_k fixing all other parameters, we will assume that L is continuous with a bound on the rate of change of the form:

$$|L(\ell_1, p_1, \ldots, \ell_K, p_K) - L(\ell'_1, p'_1, \ldots, \ell'_K, p'_K)| \leq$$
$$R\sum_k \big(\nu_k|\ell_k - \ell'_k| + |p_k - p'_k|\big), \qquad (3)$$

for parameter $R \geq 0$ and all $\ell_k, \ell'_k, p_k, p'_k \in [0, 1]$. Note that the ν_k in the above bound is necessary for our analysis because a loss that depends on ℓ_k without ν_k may require exponentially large quantities of data to estimate and optimize over if ν_k is exponentially small. Of course, alternatively ν_k could be removed from this assumption by imposing a lower bound on all ν_k.

Many losses, such as L_1 and $L_{NP\lambda}$ above, can be shown to satisfy this continuity requirement for $R = 1$ and $R = 2$, respectively. We also note that the reduction we present can be modified to address certain discontinuous loss functions. For instance, for a given target allocation (i.e., a fixed fraction of positive classifications for each group), one simply finds the classifier of minimal empirical error for each group which achieves the desired fraction of positives as closely as possible.

A transfer learning algorithm for $\mathcal{C}$ is $A : (\mathcal{X} \times \{0, 1\})^* \times (\mathcal{X} \times \{0, 1\})^* \to 2^{\mathcal{C}}$, where A takes *in-group examples* $\langle x_i, y_i\rangle_{i=1}^n$ and *out-group examples* $\langle x'_i, y'_i\rangle_{i=1}^{n'}$, both from $\mathcal{X} \times \{0, 1\}$. This is also called *supervised domain adaptation*. The distribution of out-group examples is different from (but related to) the distribution of in-group samples. The motivation for using the out-group examples is that if one is trying to learn a classifier on a small dataset, accuracy may be increased using related data.

Algorithm 2: G.D. $(T, \hat{L}, \{\langle x_i, y_i\rangle\}, \{\mathcal{X}_i\})$

1. For $k = 1$ to K,
   - $n_k \leftarrow |\{i \le n \mid x_i \in \mathcal{X}_k\}|$
   - $C_k \leftarrow T\big(\langle x_i, y_i\rangle_{i:x_i\in\mathcal{X}_k}, \langle x_i, y_i\rangle_{i:x_i\notin\mathcal{X}_k}\big)$
     ▷ Run transfer learner, output is a set
2. For all $c \in C_k$,
   - $\hat{p}_k[c] \leftarrow \frac{1}{n}\sum_{i:x_i\in\mathcal{X}_k} c(x_i)$
     ▷ Estimate profile
   - $\hat{\ell}_k[c] \leftarrow \frac{1}{n_k}\sum_{i:x_i\in\mathcal{X}_k} |y_i - c(x_i)|$
     ▷ Estimate error rates
3. **return** $\gamma_{\boldsymbol{c}}$ for $\boldsymbol{c} \in \arg\min_{C_1\times\ldots\times C_K} \hat{L}\left(\langle\hat{\ell}_i[c_i], \hat{p}_i[c_i]\rangle_{i=1}^K\right)$

The general decoupling algorithm uses a transfer learning algorithm T.

In the next section, we describe and analyze a simple transfer learning algorithm that down-weights samples from the out-group. For that algorithm, we show:

Theorem 8 *Suppose that, for any two groups $j, k \le K$ and any classifiers $c, c' \in \mathcal{C}$,*

$$|(\ell_j(c) - \ell_j(c')) - (\ell_k(c) - \ell_k(c'))| \le \Delta \qquad (4)$$

For algorithm 2 with the transfer learning algorithm described in Section 6.1, with probability $\ge 1-\delta$ over the n iid training data, the algorithm outputs $\hat{c}$ with $L(\hat{c})$ at most

$$\min_{c\in\mathcal{C}} L(c) + 5RK\tau + R\sum_k \min\left(\tau\sqrt{\frac{1}{\nu_k - \tau}}, \Delta\right),$$

where $\tau = \sqrt{\frac{2}{n}\log(8|\mathcal{C}|(n+K)/\delta)}$. For constant K, the run-time of the algorithm is polynomial in n and the runtime of the optimizer over $\mathcal{C}$.

The assumption in (4) states that the performance difference between classifiers is similar across different groups and is weaker than an assumption of similar classifier performance across groups. Note that it would follow from a simpler but stronger requirement that $|\ell_j(c) - \ell_k(c)| \le \Delta/2$ by the triangle inequality.

Parameter settings (see Lemma 10) and tighter bounds can be found in the next section. However, we can still see qualitatively that, as n grows, the bound decreases roughly like $O(n^{-1/2})$ as expected. We also note that for groups with large ν_k, as we will see in the next section, the transfer learning algorithm places weight 0 on (and hences ignores) the out-group data. For small[3] ν_k, the algorithm will place significant weight on the out-group data.

6.1. A transfer learning algorithm T

In this section, we describe analyze a simple transfer learning algorithm that down-weights[4] out-group examples by parameter $\theta \in [0, 1]$. To choose θ, we can either use cross-validation on an independent held-out set, or θ can be chosen to minimize a bound as we now describe. The cross-validation, which we do in our experiments, is appropriate when one does not have bounds at hand on the size of set of classifiers or the difference between groups, as we shall assume, or when one simply has a black-box learner that does not perfectly optimize over $\mathcal{C}$. We now proceed to derive a bound on the error that will yield a parameter choice θ.

Consider k to be fixed. For convenience, we write $n_{-k} = n - n_k$ as the number of samples from other groups. Define $\hat{\ell}_{-k}$ and ℓ_{-k} analogously to $\hat{\ell}_k$ and ℓ_k for out-of-group data $x_i \notin \mathcal{X}_k$.

Instead of outputting a set of classifiers, one for each different number of positives within group k, it will be simpler to think of the group-k profile $\hat{p}_k = P$ as being specified in advance, and we hence focus our attention on the subset of classifiers,

$$\mathcal{C}_{kP} = \left\{c \in \mathcal{C} \;\middle|\; \frac{1}{n}\sum_{i:x_i\in\mathcal{X}_k} c(x_i) = P\right\},$$

which depends on the training data. The bounds in this section will be uninteresting, of course, when $\mathcal{C}_{kP}$ is empty (e.g., in the unlikely event

3. For very small $\nu_k < \tau$, the term $\nu_k - \tau$ is negative (making the left side of the above min imaginary), in which case we define the min to be the real term on the right.
4. If the learning algorithm doesn't support weighting, subsampling can be used instead.

that $x_1 = x_2 = \ldots = x_n$, the only realizable $\hat{p}_k$ of interest are 0 and 1). The general algorithm will simply run the subroutine described in this section $n_k+1 \leq n+1$ times, once for each possible value of $\hat{p}_k$.[5] Of course, $|\mathcal{C}_{kP}| \leq |\mathcal{C}|$.

As before, we will assume that the underlying learner is optimal, meaning that given a weighted set of examples $(w_1, x_1, y_1), \ldots, (w_n, x_n, y_n)$ with total weight $W = \sum w_i$, it returns a classifier $c \in \mathcal{C}_{kP}$ that has minimal weighted error $\sum \frac{w_i}{W}|y_i - c(x_i)|$ among all classifiers in $\mathcal{C}_{kP}$.

In Appendix A, we derive a closed-form solution for θ, the (approximately) optimal down-weighting of out-group data for our transfer learning algorithm. This solution depends on a bound on the difference in classifier ranking across different groups. For small Δ, the difference in error rates of each pair of classifiers is approximately the same for in-group and out-group data. In this case, we expect generalization to work well and hence $\theta \approx 1$. For large Δ, out-group data doesn't provide much guidance for the optimal in-group classifier, and we expect $\theta \approx 0$.

For a fixed k and $\theta \in [0, 1]$, let $\hat{c}$ be a classifier that minimizes the empirical loss when out-of-group samples are down-weighted by θ, i.e.,

$$\hat{c} \in \operatorname*{arg\,min}_{c \in \mathcal{C}_{kP}} n_k \hat{\ell}_k(c) + \theta n_{-k} \hat{\ell}_{-k}(c),$$

and c^* be an optimal classifier that minimizes the true loss, i.e.,

$$c^* \in \operatorname*{arg\,min}_{c \in \mathcal{C}_{kP}} \ell_k(c).$$

We would like to choose θ such that $\ell_k(\hat{c})$ is close to $\ell_k(c^*)$. In order to derive a closed-form solution for θ in terms of Δ, we use concentration bounds to bound the expected error rates of $\hat{c}$ and c^* in terms of Δ and θ, and then choose θ to minimize this expression.

We find that, as long as $n_k < \frac{2}{\Delta^2} \log \frac{2|\mathcal{C}|}{\delta}$ the optimal choice of θ will be strictly in between 0 and 1.

7. Experiment

For this experiment, we used data that is "semi-synthetic" in that the 47 datasets are "real" (downloaded from openml.org) but an arbitrary binary attribute was used to represent a sensitive attribute, so $K = 2$. The base classifier was chosen to be least-squares linear regression for its simplicity (no parameters), speed, and reproducibility.

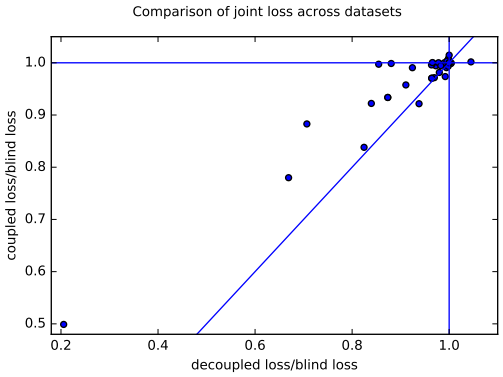

Figure 2: Comparing the joint loss of our decoupled algorithm with the coupled and blind baselines. Each point is a dataset. A ratio less than 1 means that the loss was smaller for the decoupled or coupled algorithm than the blind baseline, i.e, that using the sensitive feature resulted in decreased error. Points above the diagonal represent datasets in which the decoupled algorithm outperformed the coupled one.

In particular, each dataset was a univariate regression problem with balanced loss for squared error, i.e., $\hat{L}_B = \frac{1}{2}(\hat{\ell}_1 + \hat{\ell}_2)$ where $\hat{\ell}_k = \sum_{i:g_i=k}(y_i - z_i)^2/n_k$. To gather the datasets, we first selected the problems with twenty or fewer dimensions. Classification problems were converted to regression problems by assigning $y = 1$ to the most common class and $y = 0$ to all other classes. Regression problems were normalized so that $y \in [0, 1]$. Categorical attributes were similarly converted to binary features by assigning 1 to the most frequent category and 0 to others.

The sensitive attribute was chosen to be the first binary feature such that there were at least 100 examples in both groups (both 0 and 1 values). Further, large datasets were truncated so that there were at most 10,000 examples in each group. If there was no appropriate sensitive at-

5. In practice, classification learning algorithms generally learn a single real-valued score and consider different score thresholds.

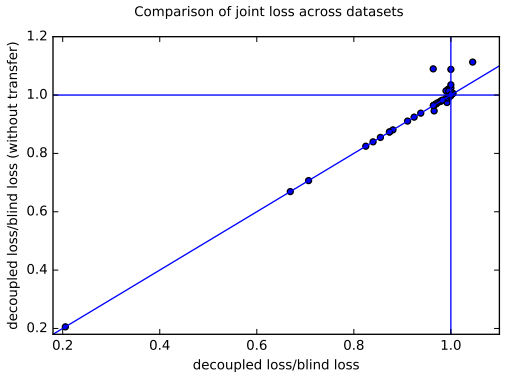

Figure 3: Comparing the joint loss of our decoupled algorithm with the decoupled algorithm with and without transfer learning. Each point is a dataset. A ratio less than 1 means that the loss was smaller for the decoupled algorithm than the blind baseline. Points above the diagonal represent datasets in which transfer learning improved performance compared to decoupling without transfer learning.

tribute, then the dataset was discarded. We also discarded a small number of "trivial" datasets in which the data could be perfectly classified (less than 0.001 error) with linear regression. The openml id's and detailed error rates of the 45 remaining datasets are listed in the appendix.

All experiments were done with five-fold cross-validation to provide an unbiased estimate of generalization error on each dataset. Algorithm 2 was implemented, where we further used five-fold cross validation (within each of the outer folds) to choose the best down-weighting parameter $\theta \in \{0, 2^{-10}, 2^{-9}, \ldots, 1\}$ for each group. Hence, least-squares regression was run $5 * 5 * 11 = 275$ times on each dataset to implement our algorithm.

The baselines were considered: the *blind* baseline is least-squares linear regression that has no access to the sensitive attribute, the *coupled* baseline is least-squares linear regression that can take into account the sensitive attribute.

Figure 2 compares the loss of the coupled baseline (x-axis) and our decoupled algorithm (y-axis) to that of the blind baseline. In particular, the log ratio of the squared errors is plotted, as this quantity is immune to scaling of the y values. Each point is a dataset. Points to the left of 1 ($x < 1$) represent datasets where the coupled classifier outperformed the blind one. Points below the horizontal line $y < 1$ represent points in which the decoupled algorithm outperformed the indiscriminate baseline. Finally, points above the diagonal line $x = y$ represent datasets where the decoupled classifier outperformed the coupled classifier.

Figure 3 compares transfer learning to decoupling without any transfer learning (i.e., just learning on the in-group data or setting $\theta = 0$). As one can see, on a number of datasets, transfer learning significantly improves performance. In fact, without transfer learning the coupled classifiers significantly outperform decoupled classifiers on a number datasets.

8. Image retrieval experiment

In this section, we describe an anecdotal example that illustrates the type of effect the theory predicts, where a classifier biases towards minority data that which is typical of the majority group. We hypothesized that standard image classifiers for two groups of images would display bias towards the majority group, and that a decoupled classifier could reduce this bias. More specifically, consider the case where we have a set $X = \mathcal{X}_1 \cup \mathcal{X}_2$ of images, and want to learn a binary classifier $c : X \to \{0, 1\}$. We hypothesized that a coupled classifier would display a specific form of bias we call *majority feature bias*, such that images in the minority group would rank higher if they had features of images in the majority group.

We tested this hypothesis by training classifiers to label images as "suit" or "no suit". We constructed an image dataset by downloading the "suit, suit of clothes" synset as a set of positives, and "male person" and "female person" synsets as the negatives, from ImageNet Deng et al. (2009). We manually removed images in the negatives that included suits or were otherwise outliers, and manually classified suits as "male" or "female", removing suit images that were neither. We used the pre-trained BVLC CaffeNet model – which is similar to the AlexNet mode from Krizhevsky et al. (2012) – to generate features for the images and clean the dataset. We

Figure 4: Differences between image classifications of "suit" using standard linear classifiers and decoupled classifiers (trained using standard neural network image features). The females selected by the linear classifier are wearing a tuxedo and blazer more typical of the majority male group.

used the last fully connected of layer ("fc7") of the CaffeNet model as features, and removed images where the most likely label according to the CaffeNet model was "envelope" (indicating that the image was missing), or "suit, suit of clothes" or "bow tie, bow-tie, bowtie" from the negatives. The dataset included 506 suit images (462 male, 44 female) and 1295 no suit images (633 male, 662 female).

We then trained a coupled and decoupled standard linear support vector classifier (SVC) on this dataset, and provide anecdotal evidence that the decoupled classifier displays less majority feature bias than the coupled classifier. We trained the coupled SVC on all images, and then ranked images according to the predicted class. We trained decoupled SVCs, with one SVC trained on the male positives and all negatives, and the other on female positives and all negatives. Both classifiers agreed on eight of the top ten "females" predicted as "suit", and Fig. 4 shows the four images (two per classifier) that differed. One of the images found by the coupled classifier is a woman in a tuxedo (typically worn by men), which may be an indication of majority feature bias; adding a binary gender attribute to the coupled classifier did not change the top ten predictions for "female suit." We further note that we also tested both the coupled and decoupled classifier on out-of-sample predictions using 5-fold cross-validation, and that both were highly accurate (both had 94.5% accuracy, with the coupled classifier predicting one additional true positive).

We emphasize that we present this experiment to provide an anecdotal example of the potential advantages of a decoupled classifier, and we do not make any claims on generalizability or effect size on this or other real world datasets because of the small sample size and the several manual decisions we made.

9. Conclusions

In this paper, we give a simple technical approach for a practitioner using ML to incorporate sensitive attributes. Our approach avoids unnecessary accuracy tradeoffs between groups and can accommodate an application-specific objective, generalizing the standard ML notion of loss. For a certain family of "weakly monotonic" fairness objectives, we give a black-box *reduction* that can use any off-the-shelf classifier to efficiently optimize the objective. In contrast to much prior work on ML which first requires complete fairness, this work requires the application designer to pin down a specific loss function that trades off accuracy for fairness.

Experiments demonstrate that decoupling can reduce the loss on some datasets for some potentially sensitive features.

References

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: Theres software used across the country to predict future criminals. and its biased against blacks. *ProPublica, May*, 23, 2016.

Elias Bareinboim and Judea Pearl. Causal inference and the data-fusion problem. *Proceedings of the National Academy of Sciences*, 113(27): 7345–7352, 2016.

Richard Berk. The role of race in forecasts of violent crime. *Race and social problems*, 1(4): 231, 2009.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *arXiv*, 2017.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. *ITCS*, 2011.

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

Sorelle A Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. On the (im)possibility of fairness. *arXiv preprint arXiv:1609.07236*, 2016.

Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. *NIPS*, 2016.

Avinatan Hassidim, Assaf Romm, and Ran I. Shorrer. Redesigning the israeli psychology master's match. *American Economic Review*, 107(5):205–09, May 2017. doi: 10.1257/aer.p20171048. URL http://www.aeaweb.org/articles?id=10.1257/aer.p20171048.

U. Hebert-Johnson, M. Kim, O. Reingold, and G. Rothblum. Calibration for the (computationally-identifiable) masses. 2017. arXiv:1711.08513v1.

Matthew Joseph, Michael Kearns, Jamie H Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. In *Advances in Neural Information Processing Systems*, pages 325–333, 2016.

Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*, ICDMW '11, pages 643–650, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4409-0. doi: 10.1109/ICDMW.2011.83. URL http://dx.doi.org/10.1109/ICDMW.2011.83.

Niki Kilbertus, Mateo Rojas-Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. *arXiv preprint arXiv:1706.02744*, 2017.

Jon Kleinberg, Sendil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. *arXiv*, 2016.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

M. J. Kusner, J. R. Loftus, C. Russell, and R. Silva. Counterfactual Fairness. *ArXiv e-prints*, March 2017.

Matt J Kusner, Joshua R Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *arXiv preprint arXiv:1703.06856*, 2017.

Razieh Nabi and Ilya Shpitser. Fair inference on outcomes. *arXiv preprint arXiv:1705.10378*, 2017.

Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: a primer*. John Wiley & Sons, 2016.

Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 560–568, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1401959. URL http://doi.acm.org/10.1145/1401890.1401959.

R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. *Proc. of Intl. Conf. on Machine Learning*, 2013.

Indre Zliobaite, Faisal Kamiran, and Toon Calders. Handling conditional discrimination. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*, ICDM '11, pages 992–1001, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4408-3. doi: 10.1109/ICDM.2011.72. URL http://dx.doi.org/10.1109/ICDM.2011.72.

Appendix A. Transfer Learning Bounds

We derive a closed-form solution for θ, the (approximately) optimal down-weighting of out-group data for our transfer learning algorithm. This solution depends on a bound Δ (defined in Theorem 8) on the difference in classifier ranking across different groups. For small Δ, the difference in error rates of each pair of classifiers is approximately the same for in-group and out-group data. In this case, we expect generalization to work well and hence $\theta \approx 1$. For large Δ, out-group data doesn't provide much guidance for the optimal in-group classifier, and we expect $\theta \approx 0$.

Finally, for a fixed k and $\theta \in [0,1]$, let $\hat{c}$ be a classifier that minimizes the empirical loss when out-of-group samples are down-weighted by θ, i.e.,

$$\hat{c} \in \arg\min_{c \in \mathcal{C}_{kP}} n_k \hat{\ell}_k(c) + \theta n_{-k} \hat{\ell}_{-k}(c),$$

and c^* be an optimal classifier that minimizes the true loss, i.e.,

$$c^* \in \arg\min_{c \in \mathcal{C}_{kP}} \ell_k(c).$$

We would like to choose θ such that $\ell_k(\hat{c})$ is close to $\ell_k(c^*)$. In order to derive a closed-form solution for θ in terms of Δ, we use concentration bounds to bound the expected error rates of $\hat{c}$ and c^* in terms of Δ and θ, and then choose θ to minimize this expression.

Lemma 9 *Fix any $k \leq K, P, n_k, n_{-k} \geq 0$ and $\Delta, \theta \geq 0$. Let $\langle x_i, y_i \rangle_{i=1}^n$ be $n = n_k + n_{-k}$ training examples drawn from $\mathcal{D}$ conditioned on exactly n_k belonging to group k. Let $\hat{c} \in \arg\min_{c \in \mathcal{C}_{kP}} n_k \hat{\ell}_k(c) + \theta n_{-k} \hat{\ell}_{-k}(c)$ be any minimizer of empirical error when the non-group-k examples have been down-weighted by θ. Then,*

$$\Pr\left[\ell_k(\hat{c}) \leq \min_{c \in \mathcal{C}_{kP}} \ell_k(c) + f(\theta, n_k, n_{-k}, \Delta, \delta)\right] \geq 1 - \delta,$$

where the probability is taken over the $n = n_k + n_{-k}$ training iid samples, and $f(\theta, n_k, n_{-k}, \Delta, \delta)$ is defined as:

$$\frac{1}{n_k + \theta n_{-k}} \left(\sqrt{2(n_k + \theta^2 n_{-k}) \log \frac{2|\mathcal{C}|}{\delta}} + \theta n_{-k} \Delta \right). \tag{5}$$

Unfortunately, the minimum value of f is a complicated algebraic quantity that is easy to compute but not easy to directly interpret. Instead, we can see that:

Lemma 10 *For f from Equation (5), $g(n_k, n_{-k}, \Delta, \delta) = \min_{\theta \in [0,1]} f(\theta, n_k, n_{-k}, \Delta, \delta)$ is at most*

$$\min\left(\sqrt{\frac{2}{n_k} \log \frac{2|\mathcal{C}|}{\delta}}, \sqrt{\frac{2}{n} \log \frac{2|\mathcal{C}|}{\delta}} + \frac{n_{-k}}{n} \Delta \right), \tag{6}$$

with equality if and only if $n_k \geq \frac{2}{\Delta^2} \log \frac{2|\mathcal{C}|}{\delta}$ in which case the minimum occurs at $\theta = 0$ where $g(n_k, n_{-k}, \Delta) = \sqrt{\frac{2}{n_k} \log \frac{2|\mathcal{C}|}{\delta}}$. Otherwise the minimum occurs at,

$$\theta^* = \sqrt{\frac{\beta^2}{4} + \frac{n_{-k}}{n_k}(1 - \beta)} - \frac{\beta}{2} \in (0, 1),$$

for $\beta = \Delta^2 \frac{2}{n_k} \log(2|\mathcal{C}|/\delta)$.

Appendix B. Dataset ids

For reproducibility, the id's and feature names for the 47 open ml datasets were as follows: (21, 'buying'), (23, 'Wifes_education'), (26, 'parents'), (31, 'checking_status'), (50, 'top-left-square'), (151, 'day'), (155, 's1'), (183, 'Sex'), (184, 'white_king_row'), (292, 'Y'), (333, 'class'), (334, 'class'), (335, 'class'), (351, 'Y'), (354, 'Y'), (375, 'speaker'), (469, 'DMFT.Begin'), (475, 'Time_of_survey'), (679, 'sleep_state'), (720, 'Sex'), (741, 'sleep_state'), (825, 'RAD'), (826, 'Occasion'), (872, 'RAD'), (881, 'x3'), (915, 'SMOKSTAT'), (923, 'isns'), (934, 'family_structure'), (959, 'parents'), (983, 'Wifes_education'), (991, 'buying'), (1014,

'DMFT.Begin'), (1169, 'Airline'), (1216, 'click'), (1217, 'click'), (1218, 'click'), (1235, 'elevel'), (1236, 'size'), (1237, 'size'), (1470, 'V2'), (1481, 'V3'), (1483, 'V1'), (1498, 'V5'), (1557, 'V1'), (1568, 'V1'), (4135, 'RESOURCE'), (4552, 'V1')