# A Fully Bayesian Tracking Algorithm for Mitigating Disparate Prediction Misclassification

May 16, 2022

**Abstract**

We develop a fully Bayesian tracking algorithm with the purpose of providing classification prediction results that are unbiased when applied uniformly to individuals with differing sensitive variable values; e.g., of different races, sexes, etc. Here, we consider bias in the form of group-level differences in false prediction rates between the different sensitive variable groups. Given that the method is fully Bayesian, it is well suited for situations where group parameters or regression coefficients are dynamic quantities. We illustrate our method, in comparison to others, on simulated datasets and two real-world datasets.

## 1 Introduction

Algorithmic scoring is employed in a variety of decision making situations including parole and bail [9, 3], loan approval [19], and credit scoring [27]. In the case of bail decisions and the well-known COMPAS algorithm, false positive rates are much higher for African-American defendants compared to Caucasian [9]. Recently a number of approaches have been introduced to improve fairness of machine learning algorithms. In [11], disparate group thresholds on logistic regression predictions are used to improve fairness post model training. In [15] the authors transform input features to achieve independence of predictions from group membership. Of particular relevance for us is the work of [29, 30], in which a penalized loss is used in training in an effort to match false positive and negative rates across groups; this particular method will be discussed in more detail below.

A number of fairness-aware forecasting methods have been introduced in the literature specifically for forecasting recidivism. In [4], the authors consider a convex surrogate loss where the step function representing the decision at the cutoff is replaced by a linear approximation (simply the score itself). Fairness can also be encouraged by post-processing forecasting scores [28]. Other work has emphasized interpretability of recidivism forecasts (for example through super-sparse integer models), which may be a more immediately achievable goal

[23, 26]. Research has shown that recidivism forecasts utilizing limited feature sets can under-perform human decision making [9], however algorithmic forecasts outperform humans when the feature set is expanded [12]. In all of these studies offline training is used and predictions are evaluated in batch on a test set. Furthermore, these methods do not usually quantify uncertainty in estimates of fairness or disparate misclassification.

In this paper we introduce a method for mitigating disparate misclassification between groups where the distributions of covariates or other relevant parameters may be changing over time and need to be tracked. Under this scenario, a fairness-aware algorithm trained offline may deviate from fair predictions over time due to the changing distribution of the data. We therefore propose a Bayesian tracking method that will estimate changing covariate and outcome probability distributions in real-time and dynamically modify decisions to mitigate disparate misclassification in predictions. The method has the additional benefit of allowing for uncertainty quantification in fairness-aware classification. To date Bayesian approaches to fairness have been limited to offline studies [8, 24] and to our knowledge this is the first to consider the Bayesian fairness tracking problem. We also note that there has been recent research on dynamic logistic regression with Bayesian variable selection [1], however such research has not addressed the question of fairness in dynamic logistic classification.

There are many possible domains where algorithmic decisions may need to be made sequentially under changes in the underlying distribution of the data. For example, issues of bias and fairness may arise in other criminal justice applications beyond recidivism, parole, and bail decisions, including traffic stops and hotspot policing based on spatial crime forecasts. These scenarios are known to present complex spatial-temporal dynamics with potential feedback [14, 5, 18]. The methodology may also apply to growth stage technology companies expanding into new geographic regions and customer segments. For example, a peer-to-peer lending company may benefit from a sequential Bayesian approach to predicting default on loans. Bias of lending decisions may change as such a company grows from an early adopter customer base into newer and larger markets. More generally, since fair prediction algorithms are of specific importance to social systems and applications, and societal changes can sometimes occur rather abruptly – in the forms of elections, new laws, rapid adoption or abandonment of fads, etc – it seems prudent to develop prediction algorithms that are able to handle such changes gracefully should they arise, while still being able to ensure fairness of these predictions.

Finally, it is known [31] that classification algorithms can lead to outcomes that display certain kinds of bias when the distribution of feature vectors varies from one group to the next, even if the algorithm does not explicitly include knowledge of the group membership when making its classifications. One could attempt to counteract this by having the algorithm explicitly take into account the group membership of the individual in question when making a prediction or classification, for the purposes of removing the implicit bias. However, this is often deemed as undesirable, and in some real-use cases, may be illegal. Instead, we develop a tracking algorithm that need not (and probably should not) explicitly take into account an individual's group membership in order to make the prediction, but instead applies the exact same method uniformly to all individuals, while nonetheless attempting to guarantee similar statistical results across groups.

The outline of the paper is as follows. In Section 2 we present the formulation of the problem that we study and the details of our Bayesian algorithm. In Section 3 we demonstrate the effectiveness of the approach on synthetic data when ground truth is known and in Section 4 we illustrate the application of the methodology to the well-known ProPublica COMPAS dataset and a dataset on traffic stops. We discuss our results and directions for future research in Section 5.

## 2 Methodology

Our algorithm shall accept as input streaming, $N$ dimensional feature data $\mathbf{x}_i$, where subscript $i$ denotes the time at which this specific data point arrives for processing; we assume that no two individual's data arrives simultaneously, so $i$ also implicitly references individuals as well. It is not strictly necessary that the data actually be generated at different, well-ordered points in time. However, as a tracking algorithm, one of the strengths of this method is that it is built to handle data that is dynamically evolving in some way, so we cast the problem in this light. Since our main concern here is providing an algorithm that is unbiased (in the sense of matched false positive or negative rates) when applied to data from individuals of different groups, we stipulate that each feature vector $\mathbf{x}_i$ is accompanied by a categorical value $z_i$ that indicates the value of a sensitive variable (sex, race, age, etc.) for the individual $i$. We will assume that feature vectors for individuals with specific sensitive variable value $z$ are drawn from a probability density $\mathcal{D}_z(\mathbf{x})$, which may be changing in time; this will be discussed in more detail below. The algorithm will then produce binary classifications $\hat{y}_i \in \{0, 1\}$ for each individual; depending on the domain in question, this classification could correspond to a belief that the individual will or will not default on a loan, commit a crime in the near future, or soon become homeless, among many other possibilities. We differentiate here between the predicted classifications $\hat{y}_i$ and the true classifications $y_i$, which are assumed to derive from some probability mass function that depends on $\mathbf{x}_i$, potentially $z_i$, and some generally unknown parameters. Note that in some domains, the true classifications may not always be available, or even in some sense exist, or may only become available after some time has passed after the predicted classification is made. For the purposes of this study, we simply assume that $y_i$ exists and is known immediately after the predicted value $\hat{y}_i$ is generated. Crucially, we will insist that the classifications our algorithm makes for the different sensitive variable groups must approximately match in terms of false prediction rates, which is one common choice for these kinds of algorithms, as mentioned above. But, we note that other bias/fairness metrics may be used as alternatives to this. See [16] for a review of fair machine learning, including a summary of the different metrics used in practice, and [6] for a discussion of the tradeoffs.

The main point of comparison for our model will be the method of Zafar et al. [29, 30]. This is largely because the Zafar method uses the same notion of bias as we have chosen here – disparity in false prediction rates between groups – and also uses (but does not specifically depend on) a logistic classifier, which we will also adopt below. Given the same inputs as described above, the Zafar method is

$$\begin{aligned}&\text{minimize: } -\sum_{i=1}^{N} \log\left[P(y_i|\theta, \mathbf{x}_i)\right] \\ &\qquad\qquad \text{subject to: } |\mathrm{Cov}(z, g_\theta(y,\mathbf{x}))| \le c_g \text{ and } |\mathrm{Cov}(z, f_\theta(y,\mathbf{x}))| \le c_f \ , \end{aligned} \quad (1)$$

where $P(y_i|\theta, \mathbf{x}_i)$ is given in (6) below for a logistic classifier and $z$ is the sensitive variable. Here, $g_\theta(y,\mathbf{x})$ and $f_\theta(y,\mathbf{x})$ are functions that serve as measures of false negatives and false positives, respectively, and are given by:

$$g_\theta(y,\mathbf{x}) = min\left(0, (2y-1)y d_\theta(\mathbf{x})\right), \quad (2)$$

$$f_\theta(y,\mathbf{x}) = min\left(0, (1-y)(2y-1) d_\theta(\mathbf{x})\right), \quad (3)$$

where $d_\theta(\mathbf{x})$ is the signed distance from the decision boundary (dictated by $\theta$) for an individual with feature vector $\mathbf{x}$, such that if $d_\theta(\mathbf{x}) \ge 0$ the person is classified as positive ($\hat{y} = 1$), and otherwise is classified as negative ($\hat{y} = 0$). The parameters $c_g$ and $c_f$ serve to limit potential covariance between sensitive variable values and false predictions, hence attempting to equalize false predictions between different groups, and are chosen such that $c = mc_u$, where $c_u$ is the value of the given covariance when using an unconstrained classifier, and $m \le 1$ is a parameter chosen by the user. Whereas the Zafar method solves the optimization problem in (1) on a fixed training data set, our goal is to develop a dynamic method for tracking and mitigating disparate false prediction rates that will be updated after each new observation, thus allowing for situations where the underlying covariate or classifier distributions are changing over time.

A second point of comparison we analyze is a simple, online version of the model detailed in [4, 17]. The model, which we will refer to as the Berk model, consists of a linear regression, $y_i = \theta^T \mathbf{x}_i$, estimated using a convex surrogate loss where the step function representing the decision at the cutoff is replaced by a linear approximation (simply the score itself):

$$MSE + \lambda\left(\sum_{\mathbf{x}_i \in S_{00}} \frac{\theta^T \mathbf{x}_i}{|S_{00}|} - \sum_{\mathbf{x}_i \in S_{10}} \frac{\theta^T \mathbf{x}_i}{|S_{10}|}\right)^2. \quad (4)$$

Here $S_{00}$ is the set of individuals of sensitive variable group 0 in the negative label class ($y_i = 0$) and $S_{10}$ is the set of individuals with sensitive variable in group 1 in the negative label group. The penalty term encourages the average scores over the negative class ($y_i = 0$) to be matched across the sensitive variable (as $\lambda$ increases). Note, then, that this method does not attempt to match results for those in the positive class, though a similar penalty to encourage matched false negative rates or precision could be easily added. Because the loss function in (4) is quadratic, there is an analytical solution [17]. The model as originally specified in [4] is not dynamic, but to make it so, we estimate the model parameters iteratively over sequential batches of data (throughout the paper we use a batch size of 500 samples). Let $b_i$ be the batch index containing data point $i$. Letting $\overline{\theta}_b$ be the parameters estimated by minimizing Equation 4 from samples in batch $b$, the dynamic update is then $\theta_b = \rho\theta_{b-1} + (1-\rho)\overline{\theta}_b$. The prediction for data point $i$ is then made using parameters $\theta_{b_i-1}$ (we do not make predictions for those points in the first batch).

The remainder of this section is broken into three parts. Our main contribution is in subsection 2.2, which presents a new method to remove disparate misclassification between groups in a dynamic Bayesian context. This method will require some way to produce classifications of individuals based on their feature vectors, and some way to estimate the feature vector distributions of individuals based on their sensitive variable value. In subsections 2.1 and 2.3, we discuss how a Bayesian classifier and Bayesian feature vector tracker may be implemented to accomplish these tasks, respectively. It is important to emphasize, though, that the specific methods we use in these two sections are not of fundamental importance to the main contribution here, and that our method for removing classification bias could be paired with other Bayesian classification and/or feature vector trackers, with only relatively small changes in the bias removal algorithm. That said, the choices made below are helpful in that the posterior distributions will all be assumed normal, allowing for some simplifications of various integrals that will appear. We note that in many real-world scenarios the features themselves, and/or the posterior distributions of parameters, may violate the normal assumption we make in deriving the algorithm. However, we find that our Bayesian logistic tracker works well in practice on the synthetic and real data examples we consider in this paper.

## 2.1 Bayesian logistic tracker

To perform the classification task, we employ a Bayesian logistic tracker. This choice is made partly for simplicity, since there are well known methods for Bayesian logistic tracking [21, 1]. Further, this provides for a more direct comparison to the Zafar algorithm described above. However, this portion of the algorithm could be accomplished by an alternative classifier if desired, so long as it can be implemented in a Bayesian way that results in a distribution over some kind of parameter space that is used to make predictions. Since this is not the main contribution of this work, which is in the method for removing bias from the classifier, we do not test any alternate classifiers here; this is an area where future work may make a contribution.

The logistic model assumes that the true classifications $y_i$ are Bernoulli random variables with probability

$$p(\mathbf{x}_i|\theta_i) = \frac{e^{\theta_i^T \mathbf{x}_i}}{1 + e^{\theta_i^T \mathbf{x}_i}} \ , \tag{5}$$

where $\theta_i$ is an $N$ dimensional vector of feature weights at time $i$. The probability of observing a specific $y_i$ for an individual with feature vector $\mathbf{x}_i$ is given by

$$P(y_i|\theta_i, \mathbf{x}_i) = p(\mathbf{x}_i|\theta_i)^{y_i} \left(1 - p(\mathbf{x}_i|\theta_i)\right)^{1-y_i} \ . \tag{6}$$

Using this equation, the classifier algorithm will attempt to recursively generate an estimate for $\theta_i$, in the form of a probability distribution, given a sequence of observations $\{\mathbf{x}_j\}$ and $\{y_j\}$ for $j \leq i$, as described below.

Noting that the model (6) is nonlinear, two prominent possibilities to perform the tracking are the Unscented Kalman filter and the Extended Kalman filter. For our purposes, we employ an Extended Kalman filter, though we make no claim of its superiority over the Unscented Kalman filter, other than its relative speed in our particular case. The resulting classifier has been presented before (see [20] for one such instance), but we briefly provide its derivation here, largely to provide an opportunity to define several key variables and concepts of our algorithm.

Let us now assume that the prior belief over $\theta_i$ before incorporating observation $y_i$ is a multivariate normal with mean $\overline{\theta}_{i|i-1}$ and covariance matrix $C_{i|i-1}$, denoted $\mathcal{N}(\theta_i|\overline{\theta}_{i|i-1}, C_{i|i-1})$. We further insist that the posterior belief over $\theta_i$ after incorporating observation $y_i$ is a multivariate normal with mean $\overline{\theta}_i$ and covariance matrix $C_i$, $\mathcal{N}(\theta_i|\overline{\theta}_i, C_i)$ . Under these assumptions, Bayes' rule, with logarithms applied to all terms, gives

$$\begin{aligned} -\frac{1}{2}\left(\theta_i - \overline{\theta}_i\right)^T C_i^{-1} \left(\theta_i - \overline{\theta}_i\right) &= y_i \ln\left[p(\mathbf{x}_i|\theta_i)\right] + \\ (1-y_i)\ln\left[1 - p(\mathbf{x}_i|\theta_i)\right] &- \frac{1}{2}\left(\theta_i - \overline{\theta}_{i|i-1}\right)^T C_{i|i-1}^{-1} \left(\theta_i - \overline{\theta}_{i|i-1}\right) + D \ , \end{aligned} \tag{7}$$

where $D$ is a constant unrelated to $\theta_i$. We now Taylor expand the logarithmic terms on the right hand side of (7) up to second order around the point $\theta_i = \overline{\theta}_{i|i-1}$ and, after some algebraic manipulation, obtain our iterative update equations by matching the linear and quadratic terms in $\theta_i$ on both sides of the equation, finding

$$C_i^{-1} = C_{i|i-1}^{-1} + \mathbf{h}_i \mathbf{h}_i^T \ , \tag{8}$$

$$\overline{\theta}_i = \overline{\theta}_{i|i-1} - C_i \mathbf{f}_i \tag{9}$$

where

$$\mathbf{f}_i = (-1)^{y_i} \mathbf{x}_i p\left((-1)^{y_i} \mathbf{x}_i | \overline{\theta}_{i|i-1}\right) \ , \tag{10}$$

$$\mathbf{h}_i = \mathbf{x}_i p\left((-1)^{1-y_i} \mathbf{x}_i \middle| \, \overline{\theta}_{i|i-1}\right) \exp\left((-1)^{y_i} \overline{\theta}_{i|i-1}^T \mathbf{x}_i / 2\right) \ . \tag{11}$$

Because of the special form of the matrix used to update $C_i^{-1}$ in (8), the equation (8) can be computed very efficiently without the need for any matrix inversion by using the rank-one update rule

$$C_i = C_{i|i-1} - \frac{C_{i|i-1}\mathbf{h}_i\mathbf{h}_i^T C_{i|i-1}}{1 + \mathbf{h}_i^T C_{i|i-1}\mathbf{h}_i} \ . \tag{12}$$

The tracking algorithm is completed by providing a model for the dynamics of $\theta_i$, allowing one to find the prior parameters $\overline{\theta}_{i+1|i}$ and $C_{i+1|i}$ for the next observation from the posterior parameters $\overline{\theta}_i$ and $C_i$ from the previous observation. For our purposes, and lacking any more informed model, we simply make the common choice that $\theta_i$ is undergoing a simple random walk $\theta_{i+1} = \theta_i + \mathcal{N}(\mathbf{0}, Q)$; this choice maintains normality of the probability distribution. Then we simply have $\overline{\theta}_{i+1|i} = \overline{\theta}_i$ and $C_{i+1|i} = C_i + Q$. The value of covariance matrix $Q$ affects how well the algorithm is able to track changes over time, with too high a value causing the tracked value to fluctuate too rapidly and too low a value causing the system to adjust too slowly to changes in the tracked variable.

## 2.2 Bias estimation and elimination

Given our streaming data and the output of our Bayesian tracker (9)-(12), one can construct predictions $\hat{y}_i$ for the classifications $y_i$ by first computing the expected probability for an individual with feature vector $\mathbf{x}_i$,

$$\overline{p}(\mathbf{x}_i|\overline{\theta}_{i|i-1}, C_{i|i-1}) = \int_{\mathbb{R}^N} p(\mathbf{x}_i|\theta_i)\mathcal{N}(\theta_i|\overline{\theta}_{i|i-1}, C_{i|i-1})d\theta_i \ , \tag{13}$$

and then thresholding this probability by a value $\tau$ such that $\hat{y}_i = 1$ if $\overline{p} > \tau$ and $\hat{y}_i = 0$ if $\overline{p} < \tau$; generally, and in the remainder of this work unless otherwise explicitly stated, $\tau = 0.5$. A simpler version of the integral above can be found by first noting that $p(\mathbf{x}_i|\theta_i)$ depends only on the argument $q_i = \theta_i^T\mathbf{x}_i$ and employing the well known property [25] that if $\theta_i \sim \mathcal{N}(\overline{\theta}_{i|i-1}, C_{i|i-1})$, then $\theta_i^T\mathbf{x}_i \sim \mathcal{N}(\overline{\theta}_{i|i-1}^T\mathbf{x}_i, \mathbf{x}_i^T C_{i|i-1}\mathbf{x}_i)$. Then we have

$$\overline{p}(\mathbf{x}_i|\overline{\theta}_{i|i-1}, C_{i|i-1}) = \int_{-\infty}^{\infty} \frac{e^{q_i}}{1+e^{q_i}}\mathcal{N}(q_i|\overline{\theta}_{i|i-1}^T\mathbf{x}_i, \mathbf{x}_i^T C_{i|i-1}\mathbf{x}_i)dq_i \ . \tag{14}$$

The above integral has several known approximations that can be used to simplify computation, and we have chosen to use the approximation from [7].

However, as previously mentioned, under many circumstances these predictions will show bias in terms of false prediction rates when comparing between those predictions made for individuals of differing sensitive variable values. For example, consider the case in which the sensitive variable value $z_i$ may only take one of two possible values, which we will simply choose to be 0 and 1 for convenience. Let us assume that the feature vectors $\mathbf{x}_i$ for those individuals with sensitive variable value 0 are well described by a probability density $\mathcal{D}_0(\mathbf{x}_i)$ and similarly $\mathcal{D}_1(\mathbf{x}_i)$ for individuals with sensitive variable value 1. Then if we were to employ posteriors $\overline{\theta}_i$ and $C_i$ to make hypothetical predictions for more individuals at time $i$, the expected instantaneous false negative rate and false positive rate for sensitive variable value $z$, $\mathrm{FNR}_i(z)$ and $\mathrm{FPR}_i(z)$ respectively, would be

$$\mathrm{FNR}_i(z) = \frac{\int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\theta}_i,C_i)<\tau}\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i}{\int_{\mathbb{R}^N} \overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i} \ , \tag{15}$$

$$\mathrm{FPR}_i(z) = \frac{\int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\theta}_i,C_i)>\tau}[1-\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)]\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i}{\int_{\mathbb{R}^N} [1-\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)]\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i} \ , \tag{16}$$

where $\mathbf{1}$ is an indicator function. Similarly, the expected instantaneous accuracy would be given by

$$\begin{aligned}\mathrm{ACC}_i(z) = \int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\theta}_i,C_i)>\tau}\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i \ + \\ \int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\theta}_i,C_i)<\tau} \left[1-\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\right] \mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i \ .\end{aligned} \tag{17}$$

It is important to note that discrepancies between the expected false prediction rates for the two sensitive variable values can arise even if the posterior distribution parameters $\overline{\theta}_i$ and $C_i$ are correctly specified; that is, even if they themselves are not biased due to flawed data being used to estimate them. This is simply due to the fact that $\mathcal{D}_0(\mathbf{x}_i)$ and $\mathcal{D}_1(\mathbf{x}_i)$ may differ [31].

If we assume that $\overline{\theta}_i$ and $C_i$ are indeed correct, but the false prediction rates generated by using (15) and (16) are unequal between the two groups, then the only way to possibly equalize the false prediction rates, which is our goal, is to change the indicator function term present in the integrals above, which represents the prediction methodology. The method we propose to equalize false positive and negative rates leverages a surrogate multivariate normal distribution with parameters $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ rather than the distribution with parameters $\overline{\theta}_i$ and $C_i$. The details of how we obtain these parameters are given below in Equations 22 and 25. The intuition is that the posterior distribution of $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ will concentrate probability density in a subset of regions where the posterior of $\overline{\theta}_i$ and $C_i$ is concentrated, but such that false positive and negative rates are more closely matched across the sensitive variable groups.

Using these newly proposed parameters, then, our false prediction rates and accuracy are

$$\hat{\mathrm{FNR}}_i(z) = \frac{\int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\mathbf{\Theta}}_i,\mathbb{C}_i)<\tau}\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i}{\int_{\mathbb{R}^N} \overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i} \ , \tag{18}$$

$$\hat{\mathrm{FPR}}_i(z) = \frac{\int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\mathbf{\Theta}}_i,\mathbb{C}_i)>\tau}[1-\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)]\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i}{\int_{\mathbb{R}^N} [1-\overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)]\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i} \ . \tag{19}$$

$$\hat{\text{ACC}}_i(z) = \int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\mathbf{\Theta}}_i,\mathbb{C}_i)>\tau} \overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i \, + \int_{\mathbb{R}^N} \mathbf{1}_{\overline{p}(\mathbf{x}_i|\overline{\mathbf{\Theta}}_i,\mathbb{C}_i)<\tau} \left[1 - \overline{p}(\mathbf{x}_i|\overline{\theta}_i, C_i)\right] \mathcal{D}_z(\mathbf{x}_i)d\mathbf{x}_i \; . \quad (20)$$

It is important to note in these equations that the distribution with parameters $\overline{\theta}_i$ and $C_i$ is still assumed to accurately indicate whether or not an individual will exhibit $y_i = 0$ or $y_i = 1$, whereas the predicted value $\hat{y}_i$ is made using our newly proposed, ideally unbiased distribution. The goal then is to generate posterior parameters $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ that reduce or eliminate differences in false prediction rates, while still retaining some level of accuracy.

To detail how we accomplish this, we begin first by defining our precise metric for measuring expected prediction bias at time $i$

$$\Delta_i = \sqrt{[\hat{\text{FPR}}_i(1) - \hat{\text{FPR}}_i(0)]^2 + [\hat{\text{FNR}}_i(1) - \hat{\text{FNR}}_i(0)]^2} \; , \quad (21)$$

where our goal will be to make $\Delta_i < \epsilon$ for some chosen small $\epsilon$ value. Suppose, then, that we possess some prior values for $\overline{\mathbf{\Theta}}_{i|i-1}$ and $\mathbb{C}_{i|i-1}$, which at the beginning of the algorithm must be initialized in some way, presumably to the same values as $\overline{\theta}_{1|0}$ and $C_{1|0}$. Given data point $\mathbf{x}_i$, $y_i$, we will then use our Bayesian classification tracker to update them, via

$$\mathbb{C}_i = \mathbb{C}_{i|i-1} - \frac{\mathbb{C}_{i|i-1}\mathbf{H}_i\mathbf{H}_i^T\mathbb{C}_{i|i-1}}{1 + \mathbf{H}_i^T\mathbb{C}_{i|i-1}\mathbf{H}_i} \; , \quad (22)$$

$$\overline{\mathbf{\Theta}}_i = \overline{\mathbf{\Theta}}_{i|i-1} - \mathbb{C}_i\mathbf{F}_i \quad (23)$$

where

$$\mathbf{F}_i = (-1)^{y_i}\mathbf{x}_i p\left((-1)^{y_i}\mathbf{x}_i|\overline{\mathbf{\Theta}}_{i|i-1}\right) \; , \quad (24)$$

$$\mathbf{H}_i = \mathbf{x}_i p\left((-1)^{1-y_i}\mathbf{x}_i\middle|\,\overline{\mathbf{\Theta}}_{i|i-1}\right) \exp\left((-1)^{y_i}\overline{\mathbf{\Theta}}_{i|i-1}^T\mathbf{x}_i/2\right) \; ; \quad (25)$$

$\overline{\theta}_i$ and $C_i$ are also updated after this observation as described previously in (8)-(9). Upon obtaining these new posterior values $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$, we then evaluate (15)-(21). Importantly, at the end of this series of calculations we will find one of two things. One possibility is that $\Delta_i < \epsilon$, in which case the posteriors $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ are accomplishing the goal of creating predictions with little or no bias and the algorithm can simply proceed to the next observation without any need to address classification bias at this time. The other possibility is that $\Delta_i \geq \epsilon$, in which case the posteriors $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ are not accomplishing their intended goal of creating unbiased classifications, and must be modified in some way in order to meet this goal. We now detail how this modification is done.

First, we again recognize that $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ are parameters describing a multivariate normal distribution. Assuming we are dealing with the case where the current such distribution of $\overline{\mathbf{\Theta}}_i$ and $\mathbb{C}_i$ causes our bias metric to exceed its threshold, it must be true that the false positive and/or false negative rates differ too substantially between the two groups. However, we hypothesize that there are some subregions where posterior probability density is concentrated that, if they were the only regions of support when calculating $\overline{p}$ – that is, if the integral in (14), properly normalized, were only over those subregions and not all of space – then the resulting bias metric would fall below its threshold. So, to construct our prior distribution for the next step of the algorithm, we seek to alter our current posterior distribution by retaining only those regions over which the bias metric would be below its threshold, and rejecting the rest of the distribution. In practice, we achieve this via Monte Carlo sampling. Specifically, we first sample from the current posterior multivariate normal distribution described by $\mathcal{N}(\theta_i|\overline{\mathbf{\Theta}}_i, \mathbb{C}_i)$, $M_\Theta$ potential predictor coefficient vectors, each denoted by a $\tilde{\mathbf{\Theta}}_{ij}$ where index $j$ runs from 1 to $M_\Theta$. Then for each of these sampled predictor coefficient vectors $j$ we calculate $\hat{\mathrm{FNR}}_{ij}(z)$ and $\hat{\mathrm{FPR}}_{ij}(z)$ using the right hand side of (19)-(18) but with the indicator function replaced with $\mathbf{1}_{p(\mathbf{x}_i|\tilde{\mathbf{\Theta}}_{ij})}$; that is, we use the sampled predictor coefficient vectors to make the hypothetical predictions. We can then calculate the bias $\Delta_{ij}$ for each sampled vector, retaining those samples whose $\Delta_{ij} < \epsilon$ and rejecting those whose $\Delta_{ij} \geq \epsilon$. After processing all $M_\Theta$ samples in this way, the remaining, non-rejected samples all represent regions of the posterior that lead to unbiased predictions.

However, we note that it is easy to construct predictor coefficient vectors that yield classifications that are completely unbiased, but that have low predictive accuracy. Specifically, the predictor coefficient vector $\tilde{\mathbf{\Theta}}_{ij} = \mathbf{0}$ will automatically classify all individuals as positive (assuming $\tau = 0.5$), which will lead to a false positive rate of 1 and a false negative rate of 0 for both groups, making the predictions unbiased via our metric. And, depending on what the initial values for the priors of the various tracking parameters are at the beginning of the algorithm, this particular predictor vector may be quite likely to be chosen in our sampling method. But (depending on the nature of the true classifications) this particular unbiased classification will generally lead to low accuracy, in comparison with the standard, biased classifier. So, we further restrict our unbiased samples to those whose predictive accuracy $\hat{\mathrm{ACC}}_{ij}$, which is calculated via the right hand side of (20) but with the indicator functions replaced with $\mathbf{1}_{p(\mathbf{x}_i|\tilde{\mathbf{\Theta}}_{ij})}$, lies above some threshold in relation to the predictive accuracy of the standard, biased classifier in (17). Specifically, we require that

$$\min_z \left[\hat{\mathrm{ACC}}_{ij}(z)/\mathrm{ACC}_i(z)\right] > \alpha \ , \tag{26}$$

where $0 < \alpha < 1$. In our various experiments (detailed below), we have found that the obtained solutions depend on the choice of $\alpha$ (for a given $\epsilon$) in a relatively straightforward way. Specifically, there appear to be roughly two transition points for the solutions, call them $\alpha_L$ and $\alpha_H$, with $\alpha_L < \alpha_H$. For values of $\alpha < \alpha_L$, solutions tend toward the trivially unbiased answer of classifying all individuals as positive (or possibly negative), with generally low accuracy. With $\alpha > \alpha_H$, there generally are no fair predictor coefficients that exhibit the required accuracy, in which case we abort the algorithm and simply state that it was unable to attain the requested fairness and accuracy combination. Finally, for $\alpha_L < \alpha < \alpha_H$, the algorithm is able to find numerous predictor coefficient vectors that fit the required bias and accuracy constraints while retaining a non-trivial classification of individuals, and importantly seems to be generally independent of the specific $\alpha$ value used. These threshold $\alpha$ values can be found via trial and error, which in our experience has been easy to do, given that we have generally observed a significant separation between the two threshold values.

Finally, after processing all $M_\Theta$ samples both for lack of bias and desired relative accuracy, the mean $\overline{\mathbf{\Theta}}_i^\epsilon$ and covariance matrix $\mathbb{C}_i^\epsilon$ for that subset meeting these two criteria is computed, and we simply use those to construct our prior for the next step in the tracker via $\overline{\mathbf{\Theta}}_{i+1|i} = \overline{\mathbf{\Theta}}_i^\epsilon$ and $\mathbb{C}_{i+1|i} = \mathbb{C}_i^\epsilon + Q$. We see, then, that our method involves tracking two (potentially) different coefficient vector distributions using (8)-(12): the "true" distribution $\mathcal{N}(\theta_i|\overline{\theta}_i, C_i)$ that is never used to make predictions $\hat{y}$ but is used to predict the current expected bias level and accuracy, and the "unbiased" distribution $\mathcal{N}(\theta_i|\overline{\mathbf{\Theta}}_i^\epsilon, \mathbb{C}_i^\epsilon)$ that is used to make predictions $\hat{y}$ and is forced to produce a bias metric $\Delta$ that is below our threshold $\epsilon$ after every datapoint $y_i$ is assimilated via the Monte Carlo sampling method described above, while retaining at least some relative measure of accuracy.

Of course, the sampling method for constructing our unbiased distribution involves evaluating the integrals in (19)-(20) for each of our $M_\Theta$ potential predictor coefficient vectors, which is easier said than done. There are at least two difficulties: the integral may be over a high dimensional space and the integrals require the knowledge of $\mathcal{D}_z(\mathbf{x}_i)$. We deal with the first problem by performing the integral via Monte Carlo methods, at this point assuming that estimates for the distributions $\mathcal{D}_z(\mathbf{x}_i)$ are known; we will detail how these may be estimated shortly. Specifically, we sample $M_x$ feature vectors $\tilde{\mathbf{x}}_k$ from each distribution $\mathcal{D}_z$, then for each sample we compute its corresponding $\overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i)$ and $p(\tilde{\mathbf{x}}_k|\tilde{\mathbf{\Theta}}_{ij})$ via (14) and (5), respectively. Then the integrals are approximated as

$$\hat{\mathrm{FNR}}_{ij}(z) = \frac{\sum_{k=1}^{M_x} \mathbf{1}_{p(\tilde{\mathbf{x}}_k|\tilde{\mathbf{\Theta}}_{ij})<\tau} \overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i)}{\sum_{k=1}^{M_x} \overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i)} \ , \tag{27}$$

$$\hat{\mathrm{FPR}}_{ij}(z) = \frac{\sum_{k=1}^{M_x} \mathbf{1}_{p(\tilde{\mathbf{x}}_k|\tilde{\mathbf{\Theta}}_{ij})>\tau} \left[1 - \overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i)\right]}{\sum_{k=1}^{M_x} \left[1 - \overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i)\right]} \ , \tag{28}$$

$$\hat{\mathrm{ACC}}_{ij}(z) = \frac{1}{M_x} \sum_{k=1}^{M_x} \mathbf{1}_{p(\tilde{\mathbf{x}}_k|\tilde{\mathbf{\Theta}}_{ij})>\tau} \overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i) \ + \frac{1}{M_x} \sum_{k=1}^{M_x} \mathbf{1}_{p(\tilde{\mathbf{x}}_k|\tilde{\mathbf{\Theta}}_{ij})<\tau} \left[1 - \overline{p}(\tilde{\mathbf{x}}_k|\overline{\theta}_i, C_i)\right] \ ; \tag{29}$$

the integral within (17) is evaluated similarly.

## 2.3 Bayesian feature tracker

The only remaining portion of our algorithm to describe is our method for estimating the distributions $\mathcal{D}_z$ for the feature vectors $\mathbf{x}$ of individuals with sensitive variable value $z$. Similarly to the classification portion of our algorithm, there exist many methods of tracking evolving feature distributions from streaming data. Since the particular method used is not the focus of our work, and in theory any Bayesian tracker that allows one to estimate distributions of feature vectors over time could be employed, we simply adopt a standard approach. Noting that the $N$ dimensional feature vectors will generally all have an entry of 1 as their final component, allowing for a constant probability offset for all individuals in the dataset if necessary (as is standard in logistic regression) the distributions $\mathcal{D}_z$ are really $N-1$ dimensional in our case. Then we assume that the first $N-1$ entries of each $\mathbf{x}$ for a given sensitive variable value $z$ are drawn from a multivariate normal distribution $\mathcal{N}(\mathbf{x}|\mu_{z,i}, \Sigma_{z,i})$. The parameters $\mu_{z,i}$ and $\Sigma_{z,i}$ are themselves unknown, but we select their prior to be a normal-inverse-Wishart distribution with hyperparameters $\mathbf{m}_{z,i|i-1}$, $\lambda_{z,i|i-1}$, $\Phi_{z,i|i-1}$, and $\nu_{z,i|i-1}$; note that it is required that $\lambda_{z,i|i-1} > 0$, $\nu_{z,i|i-1} > N-2$, and $\Phi_{z,i|i-1}$ be a positive definite $N-1 \times N-1$ matrix. This choice is the conjugate prior of the multivariate normal distribution assumed for the observations $\mathbf{x}$, so that a relatively simple Bayesian update rule for the posterior hyperparameters is known:

$$\lambda_{z,i} = \lambda_{z,i|i-1} + 1 \tag{30}$$

$$\mathbf{m}_{z,i} = \frac{\lambda_{z,i|i-1}\mathbf{m}_{z,i|i-1} + \mathbf{x}_i}{\lambda_{z,i}} \tag{31}$$

$$\Phi_{z,i} = \Phi_{z,i|i-1} + \frac{\lambda_{z,i|i-1}}{\lambda_{z,i}}(\mathbf{x}_i - \mathbf{m}_{z,i|i-1})(\mathbf{x}_i - \mathbf{m}_{z,i|i-1})^T \tag{32}$$

$$\nu_{z,i} = \nu_{z,i|i-1} + 1 \tag{33}$$

Given these posterior hyperparameters, the posterior predictive distribution $\hat{\mathcal{D}}_z$ for $\mathbf{x}$ for sensitive variable value $z$ is multivariate $t$:

$$\hat{\mathcal{D}}_z(\mathbf{x}_i) = t_{\nu_{z,i}-N+2}\left(\mathbf{x}_i|\mathbf{m}_{z,i}, \frac{\lambda_{z,i}+1}{\lambda_{z,i}(\nu_{z,i}-N+2)}\Phi_{z,i}\right) . \tag{34}$$

It is this posterior predictive distribution that is used to generate the samples $\tilde{\mathbf{x}}_k$ used in (27)-(29). For the multivariate $t$ to have a finite mean and variance, we need $\nu_{z,i} > N$, which is more restrictive than the requirement above.

We complete the specification of the feature vector distribution tracker by providing a means by which the posterior parameters' values after step $i$ are used to construct the priors for step $i+1$. We use $\mathbf{m}_{z,i|i-1} = \mathbf{m}_{z,i-1}$, $\Phi_{z,i|i-1} = \Phi_{z,i-1}$, and $\nu_{z,i|i-1} = \nu_{z,i-1}$. Since $\lambda$ effectively serves as a factor that weighs how much the prior mean contributes to the posterior mean, and we are interested in scenarios where the mean may be evolving over time, we do not want $\lambda$ to continually increase after each observation, as (30) might indicate. Hence, we simply set $\lambda_{z,i|i-1} = \beta$. If $\beta$ is very small, then the estimated mean will fluctuate rapidly with each new datapoint and potentially cause the tracker to lose accuracy, whereas if $\beta$ is very large then the mean will flucutate very little potentially causing the tracker to have a long lag time between a true change in mean and when that is detected. Generally, an intermediate value will provide a compromise between these two extremes, allowing a true change in mean to be detected relatively quickly, while not causing the mean to fluctuate by large amounts between every datapoint.

We summarize the full algorithm in Algorithm 1, which sequentially updates the posterior as new data is observed.

## 3 Results on Synthetic data

We first illustrate our method on synthetic datasets, both with static and dynamic parameters.

### 3.1 Static parameters

We use a low dimensional case of $N = 3$. The data is generated such that the static mean of $\mathcal{D}_0$ is $\mu_0 = [-1; -3]$ while for $\mathcal{D}_1$ we have $\mu_1 = [2; 3]$; both use $\Sigma = [5, 1; 1, 5]$. To generate the simulated classifications $y_i$, each individual's $z_i$ is first determined uniformly from $\{0, 1\}$, then $\mathbf{x}_i$ is generated via the appropriate $\mathcal{D}_z$, then the dot product of this feature vector with the vector $\mathbf{v} = [1, -1]$ is taken, such that $q_i = \mathbf{v}\mathbf{x}_i$. Then, if $q_i > 0$, we let $p_i = 0.7$, while if $q_i < 0$, we let $p_i = 0.3$. Finally, the true classification $y_i$ is Bernoulli with probability $p_i$. Note, then, that the true classifications are not generated via a logistic function, though the feature vectors are in fact generated via multivariate normal distributions. We generate $10,000$ such data points. Initial values for the various tracking variable priors are $\overline{\theta}_{1|0} = [0; 0; 0]$, $C_{1|0} = 0.0001 I_3$, $\mathbf{m}_{z,1|0} = [0; 0]$, $\lambda_{z,1|0} = \beta$, $\Phi_{z,1|0} = I_2$, and $\nu_{z,1|0} = N + 1$. We use $\epsilon = 0.05$, $\alpha = 0.85$, $Q = 0.00001 I_3$, and $\beta = 49$ (this value was chosen to accomplish the compromise in tracking mentioned above). Plots in Fig. 1 show the evolution of the estimated values of $\overline{\theta}$, $\overline{\boldsymbol{\Theta}}$, $\mathbf{m}_z$, and the false prediction rates $\hat{\text{FNR}}(z)$ and $\hat{\text{FPR}}(z)$. The estimated covariance values are $\Sigma_0 = [5.00, 0.98; 0.98, 5.05]$ and $\Sigma_1 = [5.11, 1.05; 1.05, 5.02]$ after all points have been tracked.

Importantly, one can also determine the actual false prediction rates after the fact, obtaining false positive rates 0.48 and 0.48 and false negative rates 0.28 and 0.28 for sensitive values $z = 0$ and $z = 1$, respectively, over the final 9000 data points (we only use these points to allow for some stabilization of the algorithm before evaluating). These are clearly within the tolerance requested, and match quite well to the estimated false prediction rates averaged over the last 9000 data points, which give 0.50 and 0.50 for false positive rates and 0.30 and 0.30 for false negative rates. These essentially unbiased false prediction rates should be compared to the false prediction rates one would obtain if no bias elimination were employed (using a value of $\epsilon = 2$), which are 0.62 and

**Algorithm 1** Bias reducing logistic regression tracking algorithm

```
**Sequential Input:** $M$ feature vectors $\mathbf{x}_i$, sensitive variable values $z_i$, and true classifications $y_i$; bias threshold $\epsilon$, accuracy threshold $\alpha$, covariance matrix $Q$, and feature tracker sensitivity $\beta$
**Initialize:** priors $\overline{\theta}_{1|0}$, $C_{1|0}$, $\overline{\mathbf{\Theta}}_{1|0}$, $\mathbb{C}_{1|0}$, $\lambda_{z,1|0}$, $\mathbf{m}_{z,1|0}$, $\Phi_{z,1|0}$, $\nu_{z,1|0}$
**for** $i = 1 : M$ **do**
  $Z \leftarrow z_i$
  Find $\lambda_{Z,i}$, $\mathbf{m}_{Z,i}$, $\Phi_{Z,i}$, $\nu_{Z,i}$ using (30)–(33)
  **for** $z \neq Z$ **do**
    Set posteriors $\lambda_{z,i}$, $\mathbf{m}_{z,i}$, $\Phi_{z,i}$, $\nu_{z,i}$ to their current prior values
  **end for**
  Find $C_i$ and $\overline{\theta}_i$ using (9)–(12)
  Find $\mathbb{C}_i$ and $\overline{\mathbf{\Theta}}_i$ using (22)–(25)
  **for all** $z$ **do**
    Generate set of $M_x$ feature vectors $\{\tilde{\mathbf{x}}\}_z \sim \hat{\mathcal{D}}_z$ from (34)
    Find $\mathrm{ACC}_i(z)$ from (17) with integrals evaluated as in (29)
  **end for**
  $\tilde{\mathbf{\Theta}}_{i0} \leftarrow \overline{\mathbf{\Theta}}_i$
  **for all** $z$ **do**
    Compute $\hat{\mathrm{FNR}}_{i0}(z)$ and $\hat{\mathrm{FPR}}_{i0}(z)$ using (27)–(28)
  **end for**
  Compute $\Delta_{i0}$ using (21)
  **if** $\Delta_{i0} < \epsilon$ **then**
    $\overline{\mathbf{\Theta}}_i^{\epsilon} \leftarrow \overline{\mathbf{\Theta}}_i$, $\mathbb{C}_i^{\epsilon} \leftarrow \mathbb{C}_i$
  **else**
    Generate set of $M_\Theta$ coefficient vectors $\{\tilde{\mathbf{\Theta}}_i\} \sim \mathcal{N}(\overline{\mathbf{\Theta}}_i, \mathbb{C}_i)$
    **for** $j = 1 : M_\Theta$ **do**
      **for all** $z$ **do**
        Compute $\hat{\mathrm{FNR}}_{ij}(z)$, $\hat{\mathrm{FPR}}_{ij}(z)$, and $\hat{\mathrm{ACC}}_{ij}(z)$ using (27)–(29)
      **end for**
      Compute $\Delta_{ij}$ using (21)
      **if** $\Delta_{ij} \geq \epsilon$ or accuracy goal (26) is violated **then**
        Remove entry $j$ from set $\{\tilde{\mathbf{\Theta}}_i\}$
      **end if**
    **end for**
    $\overline{\mathbf{\Theta}}_i^{\epsilon} \leftarrow \mathrm{mean}\left(\{\tilde{\mathbf{\Theta}}_i\}\right)$, $\mathbb{C}_i^{\epsilon} \leftarrow \mathrm{covariance}\left(\{\tilde{\mathbf{\Theta}}_i\}\right)$
  **end if**
  Set priors $\mathbf{m}_{z,i+1|i}$, $\Phi_{z,i+1|i}$, $\nu_{z,i+1|i}$ and $\overline{\theta}_{i+1|i}$ to their current posterior values
  $\overline{\mathbf{\Theta}}_{i+1|i} \leftarrow \overline{\mathbf{\Theta}}_i^{\epsilon}$, $\mathbb{C}_{i+1|i} \leftarrow \mathbb{C}_i^{\epsilon} + Q$, $C_{i+1|i} \leftarrow C_i + Q$, $\lambda_{z,i+1|i} \leftarrow \beta$
**end for**
```

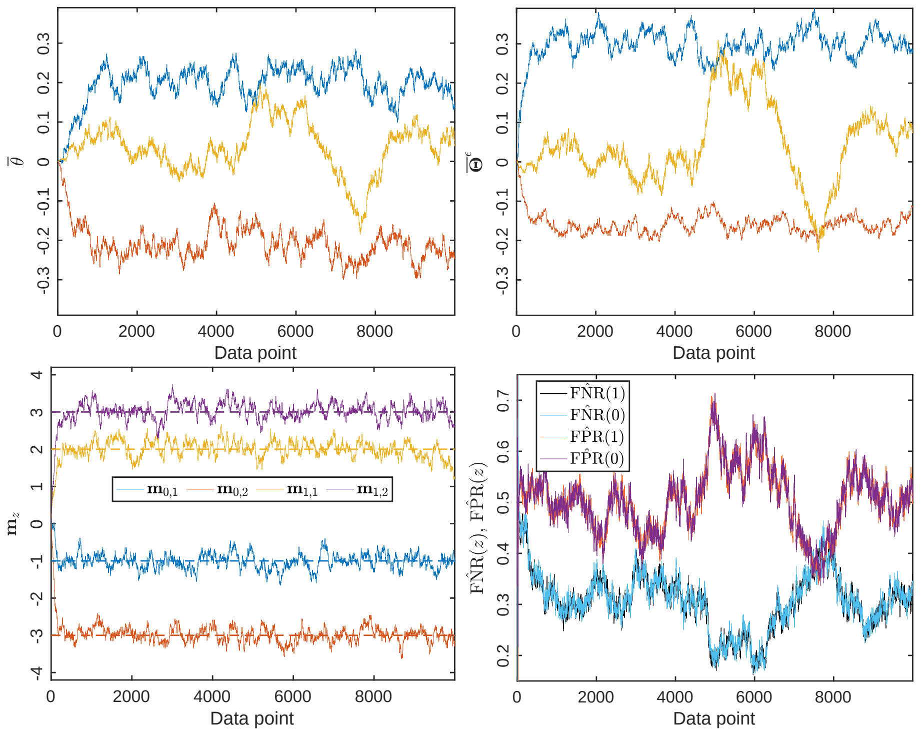

Figure 1: Plots of the tracked values of $\overline{\theta}$ (top left), $\overline{\mathbf{\Theta}}^{\epsilon}$ (top right), $\mathbf{m}_z$ (bottom left, shown with the true values $\mu_z$ as dashed lines), and the estimated unbiased false prediction rates $\hat{\mathrm{FNR}}(z)$ and $\hat{\mathrm{FPR}}(z)$ (bottom right) for static parameters described in the text.

0.23 for false positive rates and 0.11 and 0.46 for false negative rates. The tradeoff of decreased bias is also a decrease in accuracy, however. With no bias elimination, the accuracies are 0.69 and 0.67 for sensitive variable values 0 and 1, respectively, while our reduced-bias accuracies are 0.64 and 0.60.

The mean value of $\overline{\theta}$ over the last 9000 points is $[0.21, -0.21, 0.04]$, while the average value of $\overline{\boldsymbol{\Theta}}^{\epsilon}$ is $[0.30, -0.16, 0.05]$. A standard logistic regression over this set of data yields a coefficient vector $[0.21, -0.22, 0.04]$ almost identical to the mean of $\overline{\theta}$ that we obtain, showing that our logistic tracker works as anticipated. In comparison to our baseline methods, the results of our algorithm essentially match perfectly with those of the Berk method with $\lambda = 10$. We also apply the Zafar algorithm discussed previously to this dataset, with a covariance threshold multiplicative factor of $m = 0.0005$. Here, we train the algorithm over the full dataset, but only evaluate it over the last 9000 points, to put it on an equal footing with our own algorithm. The results in this case are a coefficient vector of $[0.24, -0.14, 0.04]$, leading to false positive rates of 0.49 and 0.47, and false negative rates of 0.26 and 0.29, with accuracies of 0.65 and 0.61. While certainly less biased than the standard logistic classifier, the difference in false prediction rates with the Zafar algorithm is certainly larger than in our case; consequently, the accuracy is slightly higher than in our case. This is likely due to the fact that our algorithm is directly attempting to equalize the false prediction rates, while Zafar uses a proxy measure to achieve the same goal. Of course, our algorithm is much slower than that of Zafar due to the several Monte Carlo steps involved.

## 3.2 Dynamic parameters

The scenario here is very similar to that used in the static parameter case above. The exception is that, after the first 1000 data points, the means $\mu_0$ and $\mu_1$ begin to linearly drift with each new data point, such that the two values have exactly swapped by the 10,000th data point. Specifically, we use $\mu_{0,i} = [-1; -3]$ and $\mu_{1,i} = [2; 3]$ for $i \leq 1000$ while $\mu_{0,i} = [-1; -3] + [3; 6](i - 1000)/9000$ and $\mu_{1,i} = [2; 3] - [3; 6](i - 1000)/9000$ for $i > 1000$. This highly contrived scenario allows us to focus on a situation in which, when considering the final 9000 data points all together, we expect to see little difference in false prediction rates between the two sensitive variable values even if a standard logistic regression is used. However, it is clear that there will be, in general, *instantaneous* bias in the predictions. This bias will switch between the two groups over the course of the observations, making the overall false prediction rates roughly equivalent.

We run our tracking algorithm with the same parameters and initial values as in the static case, and present results in Fig. 2. Here, our reduced bias false prediction rates as measured over the final 9000 data points are 0.52 and 0.54 for false positives and 0.22 and 0.23 for false negatives for sensitive values $z = 0$ and $z = 1$, respectively; accuracies are 0.64 and 0.62. However, as shown in Fig. 2, the estimated instantaneous false prediction rates vary significantly over the course of the tracking, albeit in such a way that the estimated bias is always within our tolerance $\epsilon$.

However, by design, when we run this scenario through our algorithm using

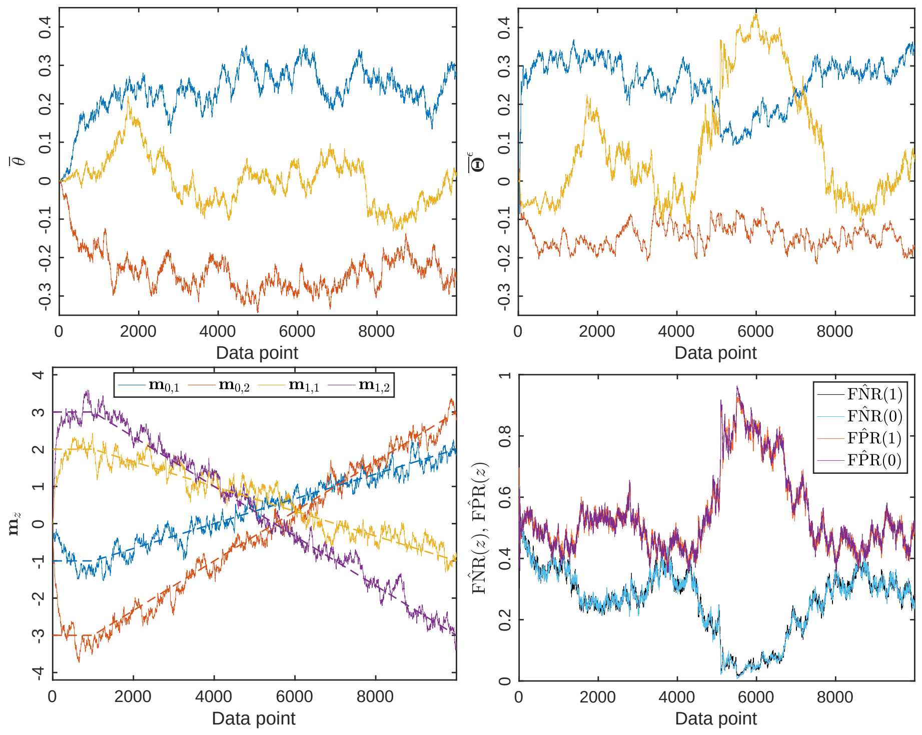

Figure 2: Plots of the tracked values of $\overline{\theta}$ (top left), $\overline{\mathbf{\Theta}}^{\epsilon}$ (top right), $\mathbf{m}_z$ (bottom left, shown with the true values $\mu_z$ as dashed lines), and the estimated unbiased false prediction rates $\hat{\mathrm{FNR}}$ and $\hat{\mathrm{FPR}}$ (bottom right) for dynamic parameters described in the text.

the large value $\epsilon = 2.0$, in which case no bias removal is actually attempted, the overall observed results still appear effectively unbiased. Over the final 9000 observations, we obtain 0.38 and 0.37 for false positive rates and 0.24 and 0.27 for false negative rates for sensitive values $z = 0$ and $z = 1$, respectively; accuracies are 0.69 and 0.69. The mean value of $\overline{\theta}$ over the last 9000 points in this case is $[0.25, -0.25, 0.00]$. All of these values are very close to those obtained for a standard logistic regression trained over this dataset, which gives coefficient vector $[0.25, -0.24, 0.00]$, and which over the last 9000 points gives false positive rates 0.36 and 0.35, false negative rates 0.23 and 0.24, and accuracies 0.71 and 0.71. In essence, because of the symmetric way that the feature distributions change over time, even a standard logistic regression will appear unbiased for this dataset when only analyzing the bulk classifications made on the interval in which the feature distributions are shifting.

The Zafar method applied to this dataset gives interesting results. Using a covariance threshold multiplicative factor of $m = 0.0005$, as used in the static case above, we obtain false positive rates of 0.06 and 0.06, false negative rates of 0.85 and 0.86, and accuracies of 0.52 and 0.51. The algorithm was able to determine some potential problem with the baseline logistic regression with regard to disparate misclassification and correct for it, but at this level was only able to resolve the problem by classifying the vast majority of individuals (89% of them) as negative. Applying a much higher coviariance threshold multiplicative factor of $m = .05$, and hence reducing the desired bias mitigation, gives false positive rates of 0.32 and 0.31, false negative rates of 0.41 and 0.42, and accuracies of 0.63 and 0.63, much more in line with the standard logistic regression, and still appearing unbiased. Of course, the dataset is designed so that even a standard logistic classifier will appear unbiased when averaged over the entire dataset, so this result is not surprising.

We run the Berk method on this dataset using parameters $\lambda = 10,\ \rho = 0.9$. The results over the final 9000 datapoints are false positive rates of 0.4 and 0.44, false negative rates of 0.29 and 0.27, and accuracies of 0.66 and 0.65; these are similar to the results of our unconstrained algorithm.

Given the design of this dataset, it is not surprising that all of the methods are able to produce results that are unbiased on average. But, to illustrate the differences between the methods on this dynamic dataset, we plot in Fig. 3 the observed false prediction rates when calculated via a symmetric moving-window average of width 2000 events. In the case of our dynamic method with $\epsilon = 0.05$ (top left panel), these moving averages show relatively small bias between the two sensitive variable values, and are similar to the estimated false prediction rates shown in Fig. 2. However, when the same moving-window average is applied to predictions made by our tracking algorithm but with $\epsilon = 2.0$ (top right panel), which mimics a static logistic regression that appears unbiased when considering the entire dataset at once as discussed above, we clearly see large differences in false prediction rates between the two protected variable groups at any given moment in time. The Zafar method with the larger value $m = 0.05$ behaves very similarly to the logistic classifier, as the two methods give very similar parameter estimates, while the results using $m = 0.0005$ (bottom left

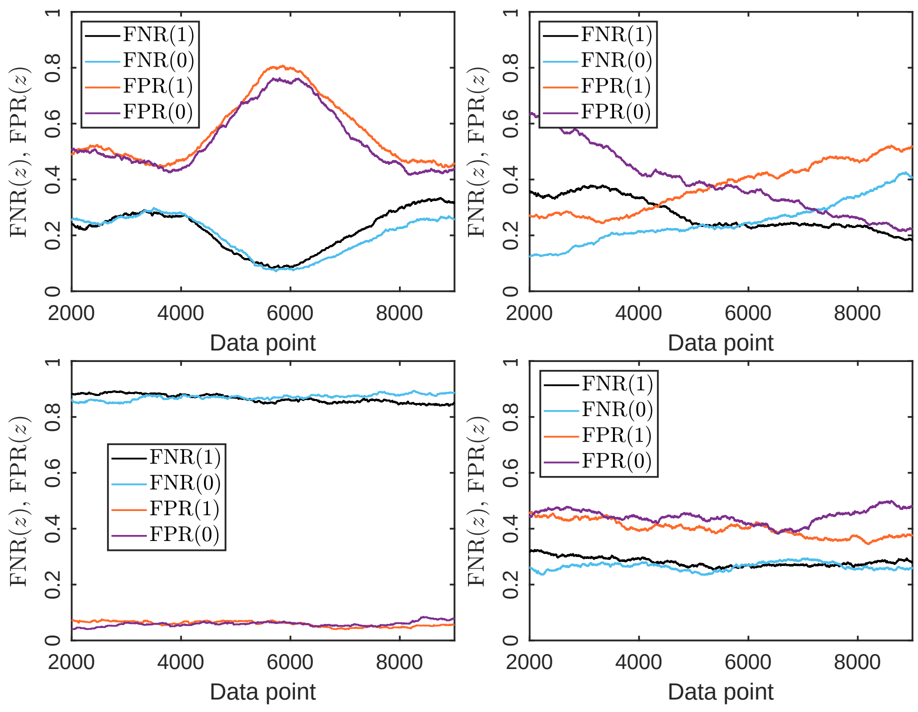

Figure 3: Plots of the observed false prediction rates computed via a moving-window average of width 2000 events for our dynamically unbiased predictions (top left panel), for effectively static logistic predictions (top right panel), for the Zafar method with $m = 0.0005$ (bottom left panel), and the Berk method with $\lambda = 10,\ \rho = 0.9$ (bottom right panel).

panel) appear effectively unbiased at all times, but only by classifying almost all individuals as negative and thereby having consistently very low accuracy. The Berk method (bottom right panel) is, as a dynamic method, able to maintain rough instantaneous fairness while also retaining good accuracy. Interestingly, the results here are qualitatively different from those of our algorithm, as the Berk method maintains similar levels of false prediction rates and accuracy throughout the data set, while our algorithm varies throughout. At this time, we can speculate that the behavior of the Berk algorithm observed here is related to the specific details of this example, as in our limited testing, other synthetic dynamic datasets do not always exhibit this same behavior with the Berk algorithm. But the specific underlying cause is not entirely clear, and highlights the need for future work on dynamic fairness algorithms, to better understand which methods might be better or worse, and under what circumstances.

## 4 Results on real-world datasets

### 4.1 ProPublica COMPAS dataset

To illustrate the ability of our algorithm to analyze real-world datasets, we applied it to the often-used ProPublica COMPAS 2-year recidivism dataset [13]. This dataset was constructed to test potential bias in recidivism forecasting between races, and includes certain features (described below) of individuals who had been arrested on suspicion of committing specific crimes, and whether or not each individual recidivized within two years of the initial event. In analyzing the dataset, we have chosen race as the sensitive variable, and only analyzed that subset of the data in which the race is listed as either "African-American" or "Caucasian". After selecting this subset, and removing a few points in the same way as described in [13], we are left with 5278 entries.

For our features, we have analyzed two scenarios. In both scenarios, we use "sex" (categorical, male=0 or female=1), "age_cat" (using two categorical variables "Less than 25" and "Greater than 45"), "priors_count" (number of prior crimes), and "c_charge_degree" (categorical, felony=0 or misdemeanor=1). In the first scenario, these are the only features considered, while in the second scenario we also directly consider "race" as the final feature (categorical, Caucasian=0 or African-American=1); this is done to match what some other have considered when analyzing this particular dataset. Importantly, we also analyze this dataset after sorting the entries by "compas_screening_date" from earliest to latest. This is done because our algorithm is specifically developed to allow for temporally evolving scenarios, so we have evaluated it as such. All initial parameters of the algorithm are the same as used in the synthetic data above, with the exception that we use $\alpha = 0.65$ here; the value of $\alpha = 0.85$ used in the synthetic experiments was too high, resulting in the algorithm failing to find predictor coefficients that were unbiased to the desired $\epsilon$ at this level of accuracy.

Table 1 lists the results of our algorithm, as well as those of the Zafar and Berk methods, for both sets of features (with and without race), and both with and without any bias constraints applied (except Berk is only analyzed in the constrained case); for the Zafar algorithm we used a covariance threshold multiplicative factor of $m = 0.000001$ and for Berk we used $\lambda = 10,\ \rho = 0.9$. The results here are computed over the second half of the dataset only; in the case of the Zafar algorithm, only the first half of the dataset is used for training. The results clearly show that, without any constraints, there is bias between false prediction rates of Caucasians vs African-Americans: false negative rates for Caucasians are higher than those for African-Americans, and the opposite is true of false positive rates. When constraints are added, all algorithms greatly reduce these differences, with ours accomplishing our goal of $\epsilon < 0.05$ in all cases, but the other algorithms generally unable to reduce the bias levels to within this same tolerance. This is not necessarily surprising, as the other algorithms only use proxy measures for bias, rather than an explicit calculation of the expected level as our own algorithm employs; further, the Berk method only attempts to equalize results for those in the negative class. Within the table, we have specifically noted the number of positive predictions given in each case. This is displayed to illustrate the fact that, in general, the constrained algorithms accomplish their goal by classifying many more individuals as negative than in the unconstrained case. The most extreme example of this is in the Zafar algorithm with race not included as a feature, in which case only 111, or 4.2%, of the 2639 predictions made are positive. This leads to an overall accuracy of only 0.48. It seems that, if not explicitly using race to make predictions, a static unbiased classifier like Zafar can only really treat this data by classifying essentially everyone in the same way (negative in this case).

| Algorithm | Features | Pred. Pos. | Overall Acc. | C. Acc. | C. FNR | C. FPR | A.A. Acc. | A.A. FNR | A.A. FPR |
|---|---|---|---|---|---|---|---|---|---|
| Ours, uncons. | w/ race | 1023 | 0.75 | 0.73 | 0.50 | 0.05 | 0.76 | 0.31 | 0.13 |
| Zafar, uncons. | w/ race | 662 | 0.61 | 0.58 | 0.83 | 0.03 | 0.63 | 0.51 | 0.15 |
| Ours, cons. | w/ race | 718 | 0.68 | 0.71 | 0.54 | 0.05 | 0.66 | 0.54 | 0.03 |
| Zafar, cons. | w/ race | 840 | 0.61 | 0.60 | 0.66 | 0.16 | 0.61 | 0.52 | 0.19 |
| Berk, cons. | w/ race | 555 | 0.58 | 0.60 | 0.71 | 0.10 | 0.56 | 0.68 | 0.08 |
| Ours, uncons. | w/o race | 1000 | 0.74 | 0.73 | 0.49 | 0.06 | 0.75 | 0.34 | 0.12 |
| Zafar, uncons. | w/o race | 607 | 0.60 | 0.60 | 0.78 | 0.06 | 0.60 | 0.59 | 0.12 |
| Ours, cons. | w/o race | 655 | 0.69 | 0.72 | 0.57 | 0.00 | 0.67 | 0.54 | 0.00 |
| Zafar, cons. | w/o race | 111 | 0.48 | 0.53 | 0.97 | 0.00 | 0.44 | 0.91 | 0.01 |
| Berk, cons. | w/o race | 361 | 0.53 | 0.56 | 0.88 | 0.03 | 0.52 | 0.75 | 0.07 |

Table 1: Results for the various algorithms on the full subset of the ProPublica dataset described in the text, under various different combinations of feature vectors (both with and without race) and constraints. Here, “C.” is “Caucasian” and “A.A.” is “African-American”.

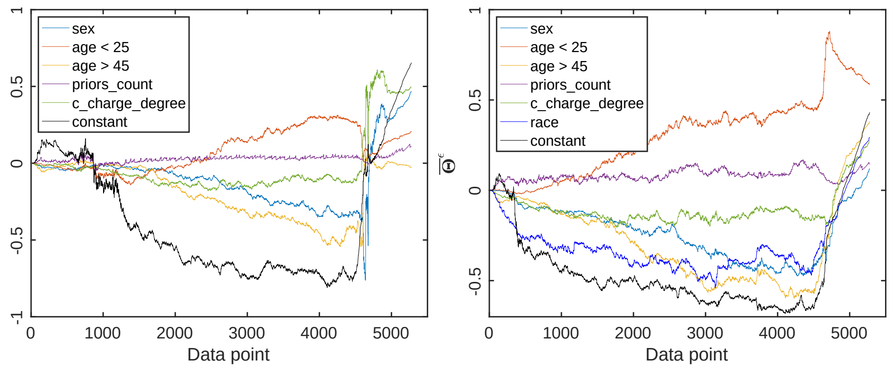

Figure 4: Plots of the tracked values of $\overline{\mathbf{\Theta}}^{\epsilon}$ for the full ProPublica COMPAS dataset both without (left panel) and with (right panel) race as a feature.

Interestingly, though, our own constrained classifier in the no-race case still makes 655 positive predictions, and ends up with an overall accuracy of 0.69, not much lower than the accuracy of 0.74 that we achieve with the same features in the unconstrained case. Somewhat amazingly, our algorithm does this with a false positive rate of 0 for both races - all 655 positive predictions were correct. Also, our algorithm displays significantly better overall accuracy than constrained Zafar and Berk across the board here. To help understand how our algorithm achieves such results, we plot in Fig. 4 the evolving estimates of the unbiased coefficients $\overline{\mathbf{\Theta}}^{\epsilon}$ that our algorithm produces in both the with-race and without-race scenarios. As can be clearly seen, there is an abrupt and very large change in estimated coefficients at around data point number 4600 in each case. Upon investigating the data directly, it was observed that every datapoint starting at number 4512 (with "compas_screening_date" of April 2, 2014) is classified as a recidivist. This data, if it is to be believed, is then a perfect example of a scenario in which a temporal trend is important to the classification task, which our algorithm handles quite naturally. The Berk algorithm also handles the data dynamically, but was unable to match the performance of our algorithm in this case. However, it is likely that these datapoints at the end of the data set are in fact erroneously classified; in fact, this precise observation has been pointed out elsewhere [2].

In light of this observation, we have performed two additional analyses on this dataset. First, we ran all of our analyses on a further subset of the COMPAS data, removing all those entries with a 'compas_screening_date" on or after April 2, 2014, as suggested by [2]. This leaves 4511 data points in this second subset. Second, we ran our algorithm and Zafar in the specific case of the constrained classifier without race (the case in which our algorithm performed suspiciously well previously) again on the full dataset, but with the dataset shuffled randomly so that any potential temporal trend (or misclassified data) was hidden. The results are shown in Table 3. In comparison to Table 1, the results in Table 3 show that the overall accuracy of our algorithm drops, while those of

Zafar increase, as one would expect. Comparing our algorithm to that of Zafar in the cases of the subset, we find that with the exception of the constrained, with-race case, results are roughly similar, though Zafar generally has slightly higher accuracy at the expense of greater bias. The constrained, with-race results on this subset are quite different between our two algorithms, however. In this case, the overall accuracy of the two algorithms are on par, but these are achieved in very different ways. Our algorithm classifies far fewer individuals as positive than Zafar in this case, ending up with quite high false negative rates and very low false positive rates, with little difference between the two races. On the other hand, the Zafar algorithm actually classifies more individuals as positive in this case than the unconstrained, with-race case, yielding a moderately high false negative rate as well as a low but notable false positive rate, and with still significant disparity between the races. Finally, we see that the Zafar results for the shuffled, full dataset are on par with those of the unshuffled full dataset, again with very few individuals classified as positive. However, our own algorithm performs very differently on the full, shuffled dataset than the unshuffled dataset, classifying a very large number of people as positive and ending up with much lower accuracy (but still similar false prediction rates). The results here are of course different than those of Zafar on the shuffled data, but neither could really be classified as better than the other; they are both quite bad.

| Algorithm | Features | Pred. Pos. | Overall Acc. | C. Acc. | C. FNR | C. FPR | A.A. Acc. | A.A. FNR | A.A. FPR |
|---|---|---|---|---|---|---|---|---|---|
| Ours, uncons. | w/ race | 408 | 0.70 | 0.72 | 0.83 | 0.05 | 0.68 | 0.58 | 0.13 |
| Zafar, uncons. | w/ race | 471 | 0.70 | 0.71 | 0.85 | 0.05 | 0.70 | 0.50 | 0.16 |
| Ours, cons. | w/ race | 197 | 0.67 | 0.72 | 0.84 | 0.05 | 0.63 | 0.83 | 0.03 |
| Zafar, cons. | w/ race | 568 | 0.67 | 0.69 | 0.73 | 0.13 | 0.66 | 0.53 | 0.19 |
| Berk, cons. | w/ race | 379 | 0.68 | 0.70 | 0.74 | 0.11 | 0.66 | 0.69 | 0.08 |
| Ours, uncons. | w/o race | 395 | 0.69 | 0.72 | 0.82 | 0.06 | 0.68 | 0.61 | 0.11 |
| Zafar, uncons. | w/o race | 416 | 0.70 | 0.71 | 0.83 | 0.06 | 0.69 | 0.57 | 0.12 |
| Ours, cons. | w/o race | 17 | 0.63 | 0.70 | 1.00 | 0.00 | 0.59 | 0.97 | 0.00 |
| Zafar, cons. | w/o race | 146 | 0.66 | 0.71 | 0.92 | 0.02 | 0.63 | 0.84 | 0.02 |
| Berk, cons. | w/o race | 235 | 0.67 | 0.71 | 0.90 | 0.03 | 0.64 | 0.76 | 0.07 |
| Ours, cons., shuffle | w/o race | 1985 | 0.49 | 0.47 | 0.25 | 0.71 | 0.50 | 0.24 | 0.78 |
| Zafar, cons., shuffle | w/o race | 13 | 0.53 | 0.61 | 1 | 0 | 0.48 | 0.99 | 0 |

Table 2: Results for the various algorithms on the restricted subset of the ProPublica dataset described in the text with all data points with "compas_screening_date" of April 2, 2014 or later removed (first ten rows), or using the full dataset but shuffled randomly (final two rows), under various different combinations of feature vectors (both with and without race) and constraints. Here, "C." is "Caucasian" and "A.A." is "African-American".

## 4.2 New Orleans data

For a second real-world example, we apply our method, along with the baseline methods, to a dataset of traffic stops in New Orleans from 2009 to 2018. The data come from the Stanford Open Policing Project [22] and contain information on the location, date, and time of the stops, race and age of the individual stopped, whether a search was conducted, and whether contraband was found. We focus on the subset of traffic stops where the race of the individual was Caucasian or African-American and where a search was conducted. After selecting this subset we are left with 73,041 traffic stops, and we evaluate the models on the first 70,000 stops. We then predict the label, defined as whether the search resulted in contraband being found ($y = 1$) or not ($y = 0$). We use as features the hour of the day (categorical with 8 bins containing 3 hours each), the district (categorical, with 7 options), the age of the individual (integer values), and race (categorical). Race is included as a feature here because the data generally appears unbiased when race is not explicitly included. We only evaluate the prediction results over the final 35,000 data points; for Zafar, we only train on the first 35,000 data points. Unlike the cases above, here we use a threshold for positive predictions of $\tau = 0.25$; this threshold leads to predictions that are not completely uniform, while the threshold of $\tau = 0.5$ generally leads to almost all negative predictions. All other parameters are the same as used for the COMPAS dataset. Given that these data specifically include time of day and location of stop, it is clear that temporal trends could play a large role, given that stops made at varying times of day and/or varying locations may be more or less likely to result in contraband being found, and there could be links between where a stop was conducted and when it was conducted. So, this dataset represents a natural venue to explore whether our method might offer advantages over methods tailored to more static data. To test this, we also ran our algorithm and Zafar on a shuffled dataset, as with the COMPAS data above.

| Algorithm | Features | Pred. Pos. | Overall Acc. | C. Acc. | C. FNR | C. FPR | A.A. Acc. | A.A. FNR | A.A. FPR |
|---|---|---|---|---|---|---|---|---|---|
| Ours, uncons. | w/ race | 12536 | 0.63 | 0.69 | 0.61 | 0.25 | 0.61 | 0.51 | 0.35 |
| Zafar, uncons. | w/ race | 2409 | 0.77 | 0.83 | 0.98 | 0.01 | 0.76 | 0.88 | 0.07 |
| Ours, cons. | w/ race | 8967 | 0.68 | 0.69 | 0.66 | 0.23 | 0.67 | 0.67 | 0.24 |
| Zafar, cons. | w/ race | 8 | 0.8 | 0.83 | 1 | 0 | 0.79 | 1 | 0 |
| Berk, cons. | w/ race | 6677 | 0.73 | 0.77 | 0.77 | 0.13 | 0.72 | 0.69 | 0.18 |
| Ours, cons., shuffle | w/ race | 9022 | 0.66 | 0.67 | 0.73 | 0.25 | 0.66 | 0.72 | 0.25 |
| Zafar, cons., shuffle | w/ race | 4708 | 0.76 | 0.78 | 0.84 | 0.11 | 0.75 | 0.78 | 0.12 |

Table 3: Results for the various algorithms on the New Orleans traffic stop data described in the text, under various constraints. Here, “C.” is “Caucasian” and “A.A.” is “African-American”.

Comparing the results, we see that our unconstrained algorithm gives by far the highest number of positive predictions, and the results are clearly biased with significantly different false prediction rates. The unconstrained Zafar algorithm seems to just predict most individuals as negative, with results still quite biased. Our constrained algorithm reduced the number of positive predictions significantly from the unconstrained version, actually yielding higher accuracy by doing so, and in a way that makes false prediction rates essentially equal between groups. The constrained Zafar algorithm is essentially all negative predictions here. Berk predicts a number of positives approaching, but less than, our constrained number, with corresponding higher accuracy, but the false prediction rates are not especially similar between the groups, so it has not reduced the bias to the level our algorithm is able to. Finally, we see that for the shuffled dataset our algorithm is still able to produce an essentially unbiased result, but with lower accuracy than the standard dataset. Interestingly, though there were more positive predictions overall for this shuffled dataset than the unshuffled version, both for our algorithm and Zafar, the false negative rate rose quite a bit for the shuffled version of the data with our algorithm, indicating that these increased positives were going to the wrong people. We also note that the behavior of Zafar on the shuffled dataset is quite different than the normal dataset, with vastly increased numbers of positive predictions. These differences in results between the two versions of the dataset give some indication that there are some temporal trends within the data that are lost upon shuffling.

## 5 Conclusions

In this work we introduced a fully Bayesian tracking algorithm for fairness-aware classification. The model sequentially tracks potential changes in the distribution of features, along with false positive and negative rates, and dynamically adjusts the model to mitigate disparate misclassification at each step. We demonstrated the effectiveness of the algorithm on synthetic and recidivism datasets, showing improved performance with regard to disparate misclassification compared to bias reducing methods that are trained in batch offline.

The present methodology has several limitations that should be noted. Here we assumed that class labels were fully observed in real time, whereas in practice some labels are unobserved and other labels may only be available after some delay. For example, in the case of traffic stop searches, the label as to whether contraband is found is immediately known and available. However, in the case of recidivism, the label may be delayed by several months or go unobserved.

The accuracy metrics considered here, namely group false positive and negative rates, may be different than those that matter to policy makers. In certain cases, precision and recall may be appropriate metrics and could be incorporated into Equation 26. In this work we focused on group-level, rather than individual-level, fairness. The method may introduce potential bias as it relates to individual fairness [10], the notion that individuals with similar features should receive similar algorithmic scores and decisions. Due to the dynamic nature of the present algorithm, an individual at an earlier time may receive a different decision than an individual with similar features at a later time. However, at each fixed time, our methodology consisted of a single model across individuals, and thus yielded similar predictions for individuals with similar features. We also note that the model performed well even when the sensitive variable was not included as a feature. We also note that the method in the present paper will likely be inefficient in high dimensional settings and that the threshold parameter $\alpha$ in Equation 26 needed to be tuned by hand. Removing these limitations will be a focus of future research.

## References

[1] Jordan Bakerman, Karl Pazdernik, Gizem Korkmaz, and Alyson G Wilson. Dynamic logistic regression and variable selection: Forecasting and contextualizing civil unrest. *International Journal of Forecasting*, 2021.

[2] Matias Barenstein. Propublica's compas data revisited. *arXiv preprint arXiv:1906.04711*, 2019.

[3] Richard Berk. An impact assessment of machine learning risk forecasts on parole board decisions and recidivism. *Journal of Experimental Criminology*, 13(2):193–216, 2017.

[4] Richard Berk, Hoda Heidari, Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, Seth Neel, and Aaron Roth. A convex framework for fair regression. *4th Workshop on Fairness, Accountability, and Transparency in Machine Learning. (FATML) 2017.*, 2017.

[5] P Jeffrey Brantingham, Matthew Valasik, and George O Mohler. Does predictive policing lead to biased arrests? results from a randomized controlled trial. *Statistics and public policy*, 5(1):1–6, 2018.

[6] Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023*, 2018.

[7] Gavin E. Crooks. Logistic approximation to the logistic-normal integral. https://threeplusone.com/pubs/on_logistic_normal.pdf, 2013.

[8] Christos Dimitrakakis, Yang Liu, David C Parkes, and Goran Radanovic. Bayesian fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 509–516, 2019.

[9] Julia Dressel and Hany Farid. The accuracy, fairness, and limits of predicting recidivism. *Science advances*, 4(1):eaao5580, 2018.

[10] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[11] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pages 3315–3323, 2016.

[12] Jongbin Jung, Sharad Goel, Jennifer Skeem, et al. The limits of human predictions of recidivism. *Science Advances*, 6(7):eaaz0652, 2020.

[13] J. Larson, S. Mattu, L Kirchner, and J. Angwin. https://github.com/propublica/compas-analysis, 2016.

[14] Kristian Lum and William Isaac. To predict and serve? *Significance*, 13(5):14–19, 2016.

[15] Kristian Lum and James Johndrow. A statistical framework for fair predictive algorithms. *arXiv preprint arXiv:1610.08077*, 2016.

[16] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635*, 2019.

[17] George Mohler and Michael D Porter. A note on the multiplicative fairness score in the nij recidivism forecasting challenge. *Crime Science*, 10(1):1–5, 2021.

[18] George Mohler, Rajeev Raje, Jeremy Carter, Matthew Valasik, and Jeffrey Brantingham. A penalized likelihood method for balancing accuracy and fairness in predictive policing. In *2018 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 2454–2459. IEEE, 2018.

[19] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617, 2020.

[20] Mahesan Niranjan. Sequential bayesian computation of logistic regression models. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, volume 2, pages 1065–1068. IEEE, 1999.

[21] William D Penny and Stephen J Roberts. Dynamic logistic regression. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, volume 3, pages 1562–1567. IEEE, 1999.

[22] Emma Pierson, Camelia Simoiu, Jan Overgoor, Sam Corbett-Davies, Daniel Jenson, Amy Shoemaker, Vignesh Ramachandran, Phoebe Barghouty, Cheryl Phillips, Ravi Shroff, et al. A large-scale analysis of racial disparities in police stops across the united states. *Nature human behaviour*, 4(7):736–745, 2020.

[23] Cynthia Rudin, Caroline Wang, and Beau Coker. The age of secrecy and unfairness in recidivism prediction. *arXiv preprint arXiv:1811.00731*, 2018.

[24] Camelia Simoiu, Sam Corbett-Davies, Sharad Goel, et al. The problem of infra-marginality in outcome tests for discrimination. *The Annals of Applied Statistics*, 11(3):1193–1216, 2017.

[25] Yung Liang Tong. *The multivariate normal distribution.* Springer Science & Business Media, 2012.

[26] Berk Ustun and Cynthia Rudin. Learning optimized risk scores. *J. Mach. Learn. Res.*, 20:150–1, 2019.

[27] Gang Wang, Jinxing Hao, Jian Ma, and Hongbing Jiang. A comparative assessment of ensemble learning for credit scoring. *Expert systems with applications*, 38(1):223–230, 2011.

[28] Dennis Wei, Karthikeyan Natesan Ramamurthy, and Flavio Calmon. Optimized score transformation for fair classification. In *International Conference on Artificial Intelligence and Statistics*, pages 1673–1683. PMLR, 2020.

[29] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th international conference on world wide web*, pages 1171–1180, 2017.

[30] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P Gummadi. Fairness constraints: A flexible approach for fair classification. *The Journal of Machine Learning Research*, 20(1):2737–2778, 2019.

[31] Lu Zhang, Yongkai Wu, and Xintao Wu. A causal framework for discovering and removing direct and indirect discrimination. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3929–3935, 2017.