FAIRGBM: GRADIENT BOOSTING WITH FAIRNESS CONSTRAINTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular data is prevalent in many high stakes domains, such as financial services or public policy. Gradient boosted decision trees (GBDT) are popular in these settings due to performance guarantees and low cost. However, in consequential decision-making fairness is a foremost concern. Despite GBDT's popularity, existing in-processing Fair ML methods are either inapplicable to GBDT, or incur in significant train time overhead, or are inadequate for problems with high class imbalance – a typical issue in these domains. We present FairGBM, a dual ascent learning framework for training GBDT under fairness constraints, with little to no impact on predictive performance when compared to unconstrained GBDT. Since observational fairness metrics are non-differentiable, we have to employ a "proxy-Lagrangian" formulation using smooth convex error rate proxies to enable gradient-based optimization. Our implementation[1] shows an order of magnitude speedup in training time when compared with related work, a pivotal aspect to foster the widespread adoption of FairGBM by real-world practitioners.

1 INTRODUCTION

The use of Machine Learning (ML) algorithms to inform consequential decision-making has become ubiquitous in a multitude of high-stakes mission critical applications, from financial services to criminal justice or healthcare (Bartlett et al., 2019; Brennan et al., 2009; Tomar & Agarwal, 2013). At the same time, this widespread adoption of ML was followed by reports surfacing the risk of bias and discriminatory decision-making affecting people based on ethnicity, gender, age, and other sensitive attributes (Angwin et al., 2016; Bolukbasi et al., 2016; Buolamwini & Gebru, 2018). This awareness led to the rise of Fair ML, a research area focused on discussing, measuring and mitigating the risk of bias and unfairness in ML systems. Despite the rapid pace of research in Fair ML (Hardt et al., 2016; Zafar et al., 2017; Agarwal et al., 2018; Narasimhan et al., 2019; Celis et al., 2021) and the release of several open-source software packages (Saleiro et al., 2018; Bellamy et al., 2018; Agarwal et al., 2018; Cotter et al., 2019b), there is still no clear winning method that "just works" regardless of data format and bias conditions.

Fair ML methods are usually divided into three families: pre-processing, in-processing and post-processing. Pre-processing methods aim to learn an *unbiased* representation of the training data but may not guarantee fairness in the end classifier (Zemel et al., 2013; Edwards & Storkey, 2016); while post-processing methods inevitably require test-time access to sensitive attributes and may not always generalize (Hardt et al., 2016; Woodworth et al., 2017). Most in-processing Fair ML methods rely on fairness constraints to impede the model from disproportionately hurting protected groups (Zafar et al., 2017; Agarwal et al., 2018; Cotter et al., 2019b). Using constrained optimization, we can optimize for the predictive performance of *fair* models.

In principle, in-processing methods have the potential to introduce fairness with no training-time overhead and a minimal predictive performance cost – an ideal outcome for most mission critical applications, such as financial fraud detection or medical diagnosis. Sacrificing a few percentage points of predictive performance in a mission critical application may result in catastrophic outcomes, from safety hazards to substantial monetary losses. Therefore, the use of Fair ML in mission critical systems is particularly challenging, as fairness must be achieved with minimal performance drops.

[1] *Anonymized link:* `https://anonymous.4open.science/r/fairgbm-DF26`

Tabular data is a common data format in a variety of mission critical ML applications (e.g., financial services). While deep learning is the dominant paradigm for unstructured data, gradient boosted decision trees (GBDT) algorithms are pervasive in tabular data due their state-of-the art performance and the availability of fast, scalable, ready-to-use implementations (e.g., LightGBM (Ke et al., 2017), XGBoost (Chen & Guestrin, 2016)). Unfortunately, Fair ML research still lacks suitable fairness-constrained frameworks for GBDT, making it challenging to satisfy stringent fairness requirements.

As a case in point, Google's TensorFlow Constrained Optimization (TFCO) (Cotter et al., 2019b), a well-known in-processing bias mitigation technique, is only compatible with neural network models. Conversely, Microsoft's ready-to-use fairlearn EG framework (Agarwal et al., 2018) supports GBDT models, but carries a substantial training overhead, and can only output binary scores instead of a continuous scoring function, making it inapplicable to a variety of use cases. Particularly, the production of binary scores is incompatible with deployment settings with a fixed budget for positive predictions (*e.g.*, resource constraint problems (Ackermann et al., 2018)) or settings targeting a specific point in the ROC curve (*e.g.*, fixed false positive rate), such as in fraud detection.

To address this gap in Fair ML, we present FairGBM, a framework for fairness constrained optimization tailored for GBDT. Our method incorporates the classical method of Lagrange multipliers within gradient-boosting, requiring only the gradient of the constraint w.r.t. (with relation to) the model's output $\hat{Y}$. Lagrange duality enables us to perform this optimization process efficiently as a two-player game: one player minimizes the loss w.r.t. $\hat{Y}$, while the other player maximizes the loss w.r.t. the Lagrange multipliers. As fairness metrics are non-differentiable, we employ differentiable proxy constraints. Our method is inspired by the theoretical ground-work of Cotter et al. (2019b), which introduce a new "proxy-Lagrangian" formulation, and prove that a stochastic equilibrium solution does exist even when employing proxy constraints. Contrary to related work, our approach does *not* require training extra models, nor keeping the training iterates in memory.

We apply our method to a large-scale account opening fraud case study, as well as to the benchmark ACSIncome dataset (Ding et al., 2021). Moreover, we enable fairness constraint fulfillment at a specific ROC point, finding fair models that fulfill business restrictions on the number of allowed false positives. This feature is a must for problems with high class imbalance, as the prevailing approach of using a decision threshold of 0.5 is only optimal when maximizing accuracy. When compared with state-of-the-art in-processing fairness interventions, our method consistently achieves improved predictive performance for the same value of fairness.

In summary, this work's main contributions are:

- A novel constrained optimization framework for gradient-boosting, dubbed FairGBM.
- Differentiable proxy functions for popular fairness metrics based on the cross-entropy loss.
- An open-source high-performance implementation of our algorithm.
- Validation on a large public dataset and a real-world case-study.

2 FairGBM Framework

We propose a fairness-aware variant of the gradient-boosting training framework, dubbed FairGBM. Our method minimizes predictive loss while enforcing group-wise parity on one or more error rates. We focus on the GBDT algorithm, which uses regression trees as the base weak learners (Breiman, 1984). Moreover, the current widespread use of GBDT is arguably due to two highly scalable variants of this algorithm: XGBoost (Chen & Guestrin, 2016) and LightGBM (Ke et al., 2017). In this work we provide a fairness-aware implementation of LightGBM. Our work is, however, generalizable to any gradient-boosting algorithm, and to any set of differentiable constraints (not limited to fairness constraints). We refer the reader to Appendix G for notation disambiguation.

2.1 Optimization under Fairness Constraints

Constrained optimization (CO) approaches aim to find the set of parameters $\theta \in \Theta$ that minimize the standard predictive loss L of a model f_θ given a set of m fairness constraints c_i, $i \in \{1, ..., m\}$:

$$\theta^* = \arg\min_{\theta \in \Theta} L(\theta) \underset{i \in \{1,...,m\}}{\text{s. t.}} c_i(\theta) \leq 0. \tag{1}$$

This problem is often re-formulated using the Lagrangian function,

$$\mathcal{L}(\theta, \lambda) = L(\theta) + \sum_{i=1}^{m} \lambda_i c_i(\theta), \tag{2}$$

where $\lambda \in \mathbb{R}_+^m$ is the vector of Lagrange multipliers. The problem stated in Equation 1 is then, under reasonable conditions, equivalent to:

$$\theta^* = \arg\min_{\theta \in \Theta} \max_{\lambda \in \mathbb{R}_+^m} \mathcal{L}(\theta, \lambda), \tag{3}$$

which can be viewed as a zero-sum two-player game, where one player (the model player) minimizes the Lagrangian w.r.t. the model parameters θ, while the other player (the constraint player) maximizes it w.r.t. the Lagrange multipliers λ (Neumann, 1928). A pure equilibrium to this game will not exist in general for a given CO problem. Sufficient conditions for there to be one are, for example, that the original problem is a convex optimization problem satisfying an appropriate constraint qualification condition (Boyd et al., 2004). Consequently, two main issues arise when using classic CO methods with fairness metrics: the loss functions of state-of-the-art ML algorithms are non-convex (as is the case of neural networks), and standard fairness metrics are non-convex and non-differentiable.

2.2 Differentiable Proxies for Fairness Metrics

As a fundamentally subjective concept, there is no "one-size-fits-all" definition of fairness. Nonetheless, popular fairness notions can be defined as equalizing rate metrics across sensitive attributes (Saleiro et al., 2018; Barocas et al., 2019). For example, *equality of opportunity* (Hardt et al., 2016) is defined as equalizing expected *recall* across members of specific protected groups (*e.g.*, different genders or ethnicities). We will focus on fairness metrics for classification tasks, as the discontinuities between class membership pose a distinct challenge (non-differentiability of the step-wise function). Extension to the regression setting is fairly straightforward, as these discontinuities are no longer present and common fairness metrics are already differentiable (Agarwal et al., 2019).

In the general case, a model f is deemed fair w.r.t. the sensitive attribute S and some rate metric L if the expected value of L is independent of the value of $s \in \mathcal{S}$:

$$\mathbb{E}\left[L(f)\right] = \mathbb{E}\left[L(f)|S = s\right], \ \forall s \in \mathcal{S}. \tag{4}$$

Equation 4 can be naturally viewed as an equality constraint in the model's training process after replacing expectations under the data distribution by their sample averages. However, as discussed in Section 2.1, common fairness notions (*i.e.*, common choices of L) are non-convex and non-differentiable. Therefore, in order to find a solution to this CO problem, we must use some proxy metric $\tilde{L}$ that is indeed differentiable (or at least sub-differentiable) (Cotter et al., 2019b).

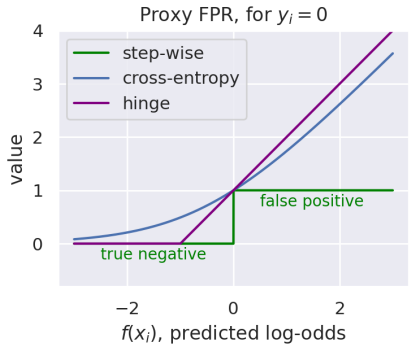


Figure 1: Convex proxies for instance-wise FPR metric, for a data sample (x_i, y_i) with negative label.

Figure 1 shows examples of convex and sub-differentiable surrogates for the False Positive Rate (FPR). Equalizing FPR among sensitive attributes is also known as *predictive equality* (Corbett-Davies

Name	**Proxy metric**, $\tilde{l}$	**Proxy derivative**, $\frac{\partial \tilde{l}}{\partial f(x)}$	**Fairness metric**
False positive	$\mathbb{I}[y=0] \cdot \log(1+e^{f(x)})$	$\mathbb{I}[y=0] \cdot \sigma(f(x))$	predictive equality
False negative	$\mathbb{I}[y=1] \cdot \log(1+e^{-f(x)})$	$\mathbb{I}[y=1] \cdot [\sigma(f(x))-1]$	equal opportunity
Predicted pos.	$\log(1+e^{f(x)})$	$\sigma(f(x))$	demographic parity
Predicted neg.	$\log(1+e^{-f(x)})$	$\sigma(f(x))-1$	demographic parity

Table 1: Instance-wise metrics used to compose common error rates and corresponding cross-entropy-based proxy metrics. σ is the sigmoid function, $f(x)$ is the predicted log-odds of instance x, and $y \in \{0,1\}$ the binary label.

et al., 2017). As any function of the confusion matrix, the FPR takes in predictions binarized using a step-wise function. As no useful gradient signal can be extracted from the step-wise function, we instead use a cross-entropy-based proxy metric that upper-bounds the step-wise function. Ideally, for some fairness constraint c, we can guarantee its fulfillment by solving the CO problem using a proxy upper-bound $\tilde{c}$, such that $c(\theta) \leq \tilde{c}(\theta) \leq 0$. Note that, while Cotter et al. (2019b) use a hinge-based proxy, which has a discontinuous derivative, we opt for a cross-entropy-based proxy, which has a continuous derivative, leading to a smoother optimization process. Table 1 shows instance-wise rate metrics commonly used to compose fairness metrics and the proposed proxy counterparts.

In practice, the fairness constraint in Equation 4 is implemented using the set of $m = |\mathcal{S}|$ inequalities in Equation 5, *i.e.*, we have, for every $b \in \mathcal{S}$:

$$\tilde{c}_b(f) = \max_{a \in \mathcal{S}} \tilde{L}_{(S=a)}(f) - \tilde{L}_{(S=b)}(f) \leq \epsilon, \tag{5}$$

where $\epsilon \in \mathbb{R}_+ \cup \{0\}$ is the allowed constraint violation, and

$$\tilde{L}_{(S=s)}(f) = \frac{1}{|D_{(S=s)}|} \sum_{(x,y) \in D_{(S=s)}} \tilde{l}(y, f(x)), \tag{6}$$

is the proxy loss measured over the dataset $D_{(S=s)} \subseteq D$ of samples with sensitive attribute $S = s$. Original (non-proxy) counterpart functions, c_b and $L_{(S=s)}$, are obtained by substituting the proxy instance-wise metric $\tilde{l}$ with its original (potentially non-differentiable) counterpart l.

2.3 Fairness-Aware GBDT

If our objective function and constraints were convex, we could find the pure Nash equilibrium of the zero sum, two-player game corresponding to the saddle point of the Lagrangian, $\mathcal{L}$. This equilibrium could be found by iterative and interleaved steps of gradient descent over our model function, f, and ascent over the Lagrange multipliers, λ. Importantly, this setting is relevant for GBDT models but not for Neural Networks, as the first have a convex objective and the latter do not. See Appendix B for a discussion on the limitations of our method.

However, as discussed in Section 2.2, fairness constraints are not differentiable, and we must employ differentiable proxies to use gradient-based optimization. Instead of using the Lagrangian $\mathcal{L}$, we instead use a proxy-Lagrangian $\tilde{\mathcal{L}}$,

$$\tilde{\mathcal{L}}(f, \lambda) = L(f) + \sum_{i=1}^{m} \lambda_i \tilde{c}_i(f), \tag{7}$$

where L is a predictive loss function, and $\tilde{c}_i$ is a proxy inequality constraints given by Equation 5. On the other hand, simply using $\tilde{\mathcal{L}}$ for both descent and ascent optimization steps would now be enforcing our proxy-constraints and not necessarily the original ones. Thus, following Cotter et al. (2019b), we adopt a non-zero sum two-player game formulation where the descent step for the model player uses the proxy-Lagrangian $\tilde{\mathcal{L}}$ and the ascent step for the λ-player uses the Lagrangian $\mathcal{L}$ with the original constraints. The FairGBM training process (Algorithm 1) is as follows:

Descent step. The FairGBM descent step consists in minimizing the loss $\tilde{\mathcal{L}}$ over the function space $\mathcal{H}$ (Equation 7). That is, fitting a regression tree on the pseudo-residuals $r_{t,i} = -g_{t,i}$, where g is the

Algorithm 1 *FairGBM* training pseudocode

Input: $T \in \mathbb{N}$, number of boosting rounds
$\mathcal{L}, \tilde{\mathcal{L}} : \mathcal{F} \times \mathbb{R}_+^m \to \mathbb{R}$, Lagrangian and proxy-Lagrangian
$\eta_f, \eta_\lambda \in \mathbb{R}_+$, learning rates

1: Let $h_0 = \arg\min_{\gamma \in \mathbb{R}} \tilde{\mathcal{L}}(\gamma, 0)$ ▷ Initial constant "guess"
2: Initialize $f \leftarrow h_0$
3: Initialize $\lambda \leftarrow 0$
4: **for** $t \in \{1, \dots, T\}$ **do**
5: Let $g_i = \frac{\partial \tilde{\mathcal{L}}(f(x_i), \lambda)}{\partial f(x_i)}$ ▷ Gradient of proxy-Lagrangian w.r.t. model
6: Let $\Delta = \frac{\partial \mathcal{L}(f(x_i), \lambda)}{\partial \lambda}$ ▷ Gradient of Lagrangian w.r.t. multipliers
7: Let $h_t = \arg\min_{h_t \in \mathcal{H}} \sum_{i=1}^N \left(-g_i - h_t(x_i)\right)^2$ ▷ Fit base learner
8: Update $f \leftarrow f + \eta_f h_t$ ▷ Gradient descent
9: Update $\lambda \leftarrow (\lambda + \eta_\lambda \Delta)_+$ ▷ Projected gradient ascent
10: **return** $h_0, \dots, h_T$

gradient of the proxy-Lagrangian, $g_{t,i} = \frac{\partial \tilde{\mathcal{L}}(f, \lambda)}{\partial f(x_i)}$,

$$g_{t,i} = \begin{cases} \frac{\partial L}{\partial f(x_i)} + (m-1) \frac{\partial \tilde{L}_{(S=s_i)}}{\partial f(x_i)} \sum_{k \in [m] \setminus \{j\}} \lambda_k & \text{if } s_i = j \\ \frac{\partial L}{\partial f(x_i)} - \lambda_k \frac{\partial \tilde{L}_{(S=s_i)}}{\partial f(x_i)} & \text{if } s_i = k \neq j \end{cases} \tag{8}$$

where $f(x_i) = f_{t-1}(x_i)$, and $j = \arg\max_{s \in \mathcal{S}} \tilde{L}_{(S=s)}(f)$ is the group with maximal proxy loss.

Ascent step. The FairGBM ascent step consists in maximizing the (original) Lagrangian $\mathcal{L}$ over the multipliers $\lambda \in \Lambda$ (Equation 2). Thus, each multiplier is updated by a simple gradient ascent step:

$$\begin{aligned} \lambda_{t,i} &= \lambda_{t-1,i} + \eta_\lambda \frac{\partial \mathcal{L}}{\partial \lambda_i} \\ &= \lambda_{t-1,i} + \eta_\lambda c_i(f) \end{aligned} \tag{9}$$

where $i \in \{1, \dots, m\}$, m is the total number of inequality constraints, and $\eta_\lambda \in \mathbb{R}_+$ is the Lagrange multipliers' learning rate.

2.3.1 Randomized classifier

The aforementioned FairGBM training process (Algorithm 1) converges to an approximately feasible and approximately optimal solution with known bounds to the original CO problem, dubbed a "coarse-correlated equilibrium" (Cotter et al., 2019b). This solution corresponds to a mixed strategy for the model player, defined as a distribution over all f_t iterates, $t \in [1, T]$. That is, for each input x, we first randomly sample $t \in [1, T]$, and then use f_t to make the prediction for x, where $f_t = \sum_{m=0}^t \eta_m h_m$.

In practice, using solely the last iterate f_T will result in a deterministic classifier that often achieves similar metrics as the randomized classifier (Narasimhan et al., 2019), although it does not benefit from the same theoretical guarantees (Appendix D goes into further detail on this comparison). There are also several methods in the literature for reducing a randomized classifier to an approximate deterministic one (Cotter et al., 2019a).

In the general case, using this randomized classifier implies training T separate models (as performed by the EG method (Agarwal et al., 2018)), severely increasing training time. When using an iterative training process (such as gradient descent), it only implies training a single model, but maintaining all T iterates (as performed by the TFCO method (Cotter et al., 2019b)), severely increasing memory consumption. Crucially, when using gradient boosting, each iterate contains all previous iterates. Therefore, a GBDT randomized classifier can be fully defined by maintaining solely the last iterate, carrying no extra memory consumption nor significant extra training time when compared with the vanilla GBDT classifier.

To summarize, FairGBM is the result of employing the proxy Lagrangian CO method with cross-entropy-based proxies of fairness constraints, resulting in a randomized classifier with known optimality and feasibility bounds.

3 EXPERIMENTS

We implemented FairGBM[1] as a fork from the open-source Microsoft LightGBM implementation. The LightGBM algorithm (Ke et al., 2017) is a widely popular high performance GBDT implementation in C++, with a high-level Python interface for ease-of-use. This algorithm in particular builds on top of the standard GBDT framework by introducing *gradient-based one-side sampling* (GOSS) and *exclusive feature bundling*, both aimed at decreasing training and inference time. Although the FairGBM framework (Algorithm 1) could be applied to any gradient boosting algorithm, we choose to implement it on top of LightGBM due to its excellent scalability.

We validate our method on a public benchmark dataset from the fairness literature and on a real-world financial services case-study. The first scenario aims to be an easily reproducible comparison under common literature objectives and constraints, while the second scenario poses a distinct set of challenges that are seldom discussed in the fairness literature, from highly imbalanced data to tight budget constraints on the number of positive predictions.

We compare FairGBM with a set of constrained optimization baselines from the Fair ML literature. Fairlearn *EG* (Agarwal et al., 2018) is a state-of-the-art method based on the reduction of CO to a cost-sensitive learning problem (discussed in Section 2.1). It produces a randomized binary classifier, composed of several base classifiers. Fairlearn *GS* (Agarwal et al., 2018) is a similar method that instead uses a grid search over the constraint multipliers λ, and outputs a single (deterministic) classifier that achieves the best fairness-performance trade-off. *RS Reweighing* is a variation of GS that instead casts the choice of multipliers λ as another model hyperparameter, which will be selected via Random Search (RS). Both EG, GS and RS are trained using LightGBM as the base algorithm, and both are implemented in the popular open-source *fairlearn* package (Bird et al., 2020). Finally, we also show results for the standard unconstrained LightGBM algorithm.

It is well-known that the choice of hyperparameters affects both performance and fairness of trained models (Cruz et al., 2021). To control for the variability of results when selecting different hyperparameters, we randomly sample 100 hyperparameter configurations of each algorithm. We follow the reproducibility check-list put forth by Dodge et al. (2019) (further details in Appendix H). In the case of EG and GS, both algorithms already fit n base estimators as part of a single training procedure. Hence, we run 10 trials of EG and GS, each with a budget of 10 iterations, for a total budget of 100 models trained (leading to an equal budget for all algorithms). To calculate the statistical mean and variance of each algorithm, we perform bootstrapping over the trained models (Efron & Tibshirani, 1994). Each bootstrap trial consists of $k = 20$ random draws from the pool of $n = 100$ trained models (randomly select 20% of the trained models). The best performing model (the maximizer of $[\alpha \cdot \textit{performance} + (1 - \alpha) \cdot \textit{fairness}]$) of each trial is then separately selected. This process was repeated for 1000 trials to obtain both variance (Tables 2–4) and confidence intervals (Figures 4–6).

3.1 ACSINCOME DATASET

The ACSIncome dataset (Ding et al., 2021) is a recently introduced 1.6M rows dataset based on the American Community Survey (ACS) public use microdata sample from 2018. It consists of high-quality data made available to the public by the US Census Bureau. It is essentially a modern-day version of the popular 1994 UCI-Adult dataset (Dua & Graff, 2017) (50K rows), which has been widely used in ML research papers over the years. The dataset is split 60%/20%/20% in train/validation/test data, with random sampling stratified by US state (authors' directions).

Each row contains a set of demographic features, such as age, marital status, education, occupation, working hours, place of birth, race, and gender. The target label corresponds to whether US working adults' yearly income is above \$50K. In this experiment, the goal is to maximize global accuracy, while equalizing group false negative rate (FNR), also known as equality of opportunity (Hardt et al., 2016), over different binary gender groups.

Algorithm	Trade-off $\alpha = 0.75$				Run-time	
	Validation		Test			
	Fair. (%)	Perf. (%)	Fair. (%)	Perf. (%)	Total (h)	Relative
ACSIncome-Adult dataset						
FairGBM	99.5 ± 0.83	81.7 ± 0.06	99.3 ± 0.89	81.7 ± 0.08	9.9	x2.1
LightGBM	75.0 ± 3.41	81.1 ± 0.87	74.6 ± 3.57	81.1 ± 0.88	4.6	*baseline*
GS	66.4 ± 1.53	81.8 ± 0.14	65.8 ± 1.39	81.8 ± 0.14	43.8	x9.6
RS	41.4 ± 26.6	77.5 ± 3.04	41.5 ± 26.5	77.5 ± 3.06	37.1	x8.1
EG	94.4 ± 0.33	81.6 ± 0.15	93.8 ± 0.13	81.6 ± 0.17	99.4	x21.7
AOF dataset						
FairGBM	89.3 ± 4.62	65.9 ± 1.33	87.5 ± 3.36	65.9 ± 1.64	3.5	x2.4
LightGBM	58.0 ± 9.39	61.7 ± 2.68	66.6 ± 14.9	61.1 ± 2.86	1.4	*baseline*
GS	98.5 ± 1.00	23.6 ± 3.45	98.4 ± 1.67	23.7 ± 3.64	21.4	x14.7
RS	84.0 ± 19.3	36.9 ± 8.43	84.6 ± 20.9	37.4 ± 8.89	10.3	x7.1

Table 2: Results for mean and standard deviation of all models on each dataset. For each algorithm, we select the top model based on the weighted sum $[\alpha \cdot \textit{performance} + (1 - \alpha) \cdot \textit{fairness}]$ measured in validation and report results on both validation and test data. See related Tables 3 and 4.

3.1.1 Results

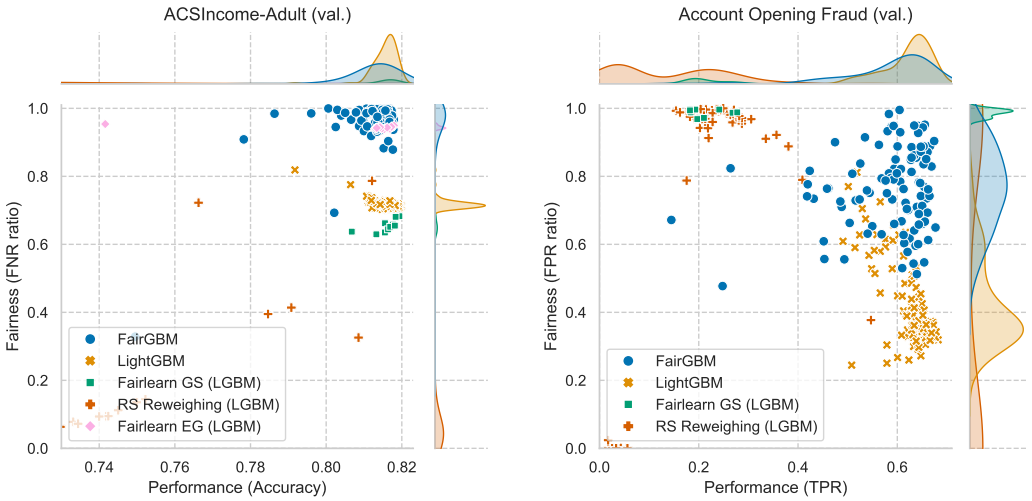
Figure 2a shows a scatter plot of the fairness-accuracy results in the validation set for trained models on the ACSIncome dataset, as well as the performance and fairness distribution for each algorithm. Note that the x axis spans a small accuracy range, as all models consistently achieve high performance on this task. Moreover, the standard fairness-blind LightGBM algorithm already achieves above 0.7 equal opportunity on this task, while naturally achieving the highest performance among all algorithms. These results further motivate our focus on a real-world setting, as it poses a much more challenging environment (compare Figures 2a and 2b).

Tables 2, 3, and 4 show results for different fairness-performance trade-offs (different choices of α for model selection). All CO algorithms were trained with an FNR parity constraint on the ACSIncome dataset, and an FPR parity constraint on the AOF dataset. Among all tested algorithms, FairGBM has the lowest average constraint violation for all three values of α on the ACSIncome dataset ($p < 0.01$ for all pair-wise comparisons), while achieving better or similar values on the performance metric. The EG algorithm follows, also achieving high performance and high fairness, although significantly behind FairGBM on the latter ($p < 0.01$). At the same time, the GS and RS Reweighing algorithms achieve a surprisingly low fairness, signalling that their randomized counterpart (the EG algorithm) seems better fitted for this setting. Importantly, the total time taken to train all FairGBM models is under a tenth of the time taken to train all EG models. As expected, fairness for the LightGBM algorithm is considerably lower than that of FairGBM or EG.

3.2 Real-world Account Opening Fraud Dataset

Each real-world environment has a set of requirements, constraints, and goals, which vary wildly depending on the context. Nonetheless, the requirements we discuss in this section (*e.g.*, alert rate, scalability) are arguably commonplace in production ML systems, and could be generalized to a wide variety of settings. Specifically, we consider a real-world setting of fraud detection in bank account opening requests for a large bank. These requests are made via an online application form. Fraudsters will attempt to open a new bank account using a synthetic identity, or to impersonate a real person using stolen credentials (or a mixture of both).

The Account Opening Fraud (AOF) dataset spans an 8-month data stream with over 500K instances. Data is split using 6 months for training, 1 month for validation, and 1 month for test, such that we train on the oldest data and test on the most recent. As fraudsters are in the minority relative to legitimate applications, our data is highly imbalanced, with only 1% fraud prevalence. Further details on this dataset are provided in Appendix F.



(a) ACSIncome results.

(b) AOF results.

Figure 2: Scatter plot showing fairness and performance of 100 trained models of each algorithm in the comparison, evaluated on validation data. EG and GS show only 10 markers, as each run already trains 10 models itself. Margins show a density plot of both metrics. Due to only providing binary (0/1) model outputs instead of real-valued scores, EG cannot be used in the AOF case-study (score threshold at 5% FPR). Note the different x axis ranges (AOF has a wider performance spread).

3.2.1 Evaluation Setup

Fraud detection is a punitive ML setting: a positive prediction leads to a negative outcome for the individual (application denied). For an ML system to be deemed fair, the likelihood of a legitimate individual to be wrongly denied access to her/his bank account (a false positive) should be independent of them belonging to a specific protected group. This is defined as equal group-wise FPR, also known as predictive equality (Corbett-Davies et al., 2017). In our setting, we target FPR parity among individuals of different age-groups (preventing ageism).

The performance goal in this setting is to maximize fraud recall at 5% false positive rate (FPR). This entails a budget on the number of false positives we can commit. See Appendix C for details on how we operate FairGBM at arbitrary ROC points. Requirements on the ROC operating point are commonplace in real-world settings, but are seldom considered in fairness benchmarks and fairness-aware algorithms. It is common practice to implicitly use a decision threshold of 0.50, which is optimal when maximizing accuracy for calibrated classifiers, but is arguably inapplicable to imbalanced data. As a case in point, we could trivially achieve 99% accuracy by predicting all inputs as negative, or 100% recall by predicting all inputs as positive. Consequently, the EG algorithm is not compatible with this real-world setting, as it outputs a *binary* randomized classifier, providing no confidence score for its predictions (using the implicit 0.50 decision threshold). As this would not make for a fair comparison, we omit this method for the AOF dataset. Moreover, the EG method also requires keeping tens of models in memory (the package default is $n = 50$), straining the scalability of the production environment.

3.2.2 Results

Figure 2b shows a scatter plot of all trained models on the AOF dataset, together with metric distributions for each algorithm on the margin. Note that, while most models achieve high performance on the ACSIncome dataset, on AOF we see a significantly wider range of performance values (compare x axis scale between the two plots). Moreover, the unconstrained LightGBM algorithm in this setting shows significant average *unfairness*.

Although FairGBM achieves a wider spread over both metrics, it significantly extends the Pareto frontier of attainable trade-offs. Indeed, FairGBM dominates LightGBM on both fairness *and*

performance for the $\alpha = 0.5$ and $\alpha = 0.75$ trade-offs, while achieving superior fairness with statistically insignificant performance differences on the $\alpha = 0.95$ trade-off (AOF test results). Although the GS method consistently achieves very high fairness, surpassing FairGBM on fairness for all three values of α, it does so at a dramatic performance cost.

4 Related Work

Prior work on algorithmic fairness can be broadly divided into three categories: pre-processing, in-processing, and post-processing; depending on whether it acts on the training data, the training process, or the model's predictions, respectively.

Pre-processing methods aim to modify the input data such that any model trained on it would no longer exhibit biases. This is typically achieved either by (1) creating a new representation U of the features X that does not exhibit correlations with the protected attribute S (Zemel et al., 2013; Edwards & Storkey, 2016), or (2) by altering the label distribution Y according to some heuristic (Fish et al., 2016; Kamiran & Calders, 2009) (*e.g.*, equalizing prevalence across sub-groups of the population). Although compatible with any downstream task, by acting on the beginning of the ML pipeline these methods may not be able to guarantee fairness on the end model. Moreover, recent empirical comparisons have shown that pre-processing methods often lag behind in-processing and post-processing methods (Ding et al., 2021).

In-processing methods alter the learning process itself in order to train models that make fairer predictions. There are a wide variety of approaches under this class of methods: training under fairness constraints (Zafar et al., 2017; Agarwal et al., 2018; Cotter et al., 2019b), using a loss function that penalizes unfairness (Fish et al., 2016; Iosifidis & Ntoutsi, 2019; Ravichandran et al., 2020), or training with an adversary that tries to predict protected-group membership (Grari et al., 2019). The main shortcoming of in-processing methods lies in their selective compatibility with particular algorithms or families of algorithms. As a case in point, there is currently no constrained optimization method tailored for the GBDT algorithm, besides the one present in this work. However, the state-of-the-art results for numerous tabular data tasks are currently held by boosting-based models (Shwartz-Ziv & Armon, 2021). AdaFair (Iosifidis & Ntoutsi, 2019) is a bias mitigation method for the AdaBoost algorithm (Freund & Schapire, 1996), a similar method to GBDT. However, we did not consider it as a direct baseline in this work as it is only compatible with the equal odds fairness metric (Hardt et al., 2016), and not with equal opportunity or predictive equality (used in our experiments). Moreover, this method lacks any theoretical guarantees, employing a series of heuristics used to change the weights of samples from underprivileged groups.

Post-processing methods alter the model's predictions to fulfill some statistical measure of fairness. In practice, this is done by (1) shifting the decision boundary for specific sub-groups (Hardt et al., 2016; Fish et al., 2016), or by (2) randomly classifying a portion of individuals of the underprivileged group (Kamiran et al., 2012; Pleiss et al., 2017). Methods based on shifting the decision-boundary have the clear advantage of achieving 100% fairness in the data where they are calibrated (training or validation data), while also being compatible with any score-based classifier. However, post-processing methods can be highly sub-optimal (Woodworth et al., 2017), as they act on the model *after* it was learned. Moreover, they can lead to higher performance degradation when compared to in-processing methods (Ding et al., 2021).

5 Conclusion

We presented FairGBM, a dual ascent learning framework under fairness constraints specifically tailored for gradient boosting. To enable gradient-based optimization we propose differentiable proxies for popular fairness metrics that are able to attain state-of-the-art fairness-performance trade-offs on tabular data. When compared with general-purpose constrained optimization methods, FairGBM is more consistent across datasets, and achieves higher fairness for the same level of performance. Crucially, FairGBM does not require significant extra computational resources, while related CO algorithms considerably increase training time (EG, GS) or memory consumption (TFCO). Finally, we enable fairness constraint fulfillment at a specified ROC point or with a fixed budget for positive predictions, a common requirement in real-world high-stakes settings.

REFERENCES

Klaus Ackermann, Joe Walsh, Adolfo De Unánue, Hareem Naveed, Andrea Navarrete Rivera, Sun-Joo Lee, Jason Bennett, Michael Defoe, Crystal Cody, Lauren Haynes, et al. Deploying machine learning models for public policy: A framework. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 15–22, 2018.

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudik, John Langford, and Hanna Wallach. A reductions approach to fair classification. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 60–69. PMLR, 10–15 Jul 2018. URL http://proceedings.mlr.press/v80/agarwal18a.html.

Alekh Agarwal, Miroslav Dudik, and Zhiwei Steven Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 120–129. PMLR, 09–15 Jun 2019.

Julia Angwin, Jeff Larson, Lauren Kirchner, and Surya Mattu. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing, May 2016.

Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. http://www.fairmlbook.org.

Robert Bartlett, Adair Morse, Richard Stanton, and Nancy Wallace. Consumer-Lending Discrimination in the FinTech Era. Technical report, National Bureau of Economic Research, 2019.

Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. AI Fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias, October 2018. URL https://arxiv.org/abs/1810.01943.

Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in AI. Technical Report MSR-TR-2020-32, Microsoft, May 2020. URL https://www.microsoft.com/en-us/research/publication/fairlearn-a-toolkit-for-assessing-and-improving-fairness-in-ai/.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *NIPS*, pp. 4349–4357. Curran Associates, Inc., 2016.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

Leo Breiman. *Classification And Regression Trees*. Wadsworth, 1984.

Tim Brennan, William Dieterich, and Beate Ehret. Evaluating the predictive validity of the compas risk and needs assessment system. *Criminal Justice and Behavior*, 36(1):21–40, 2009.

Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *FAT* '18*, volume 81, pp. 77–91. PMLR, 23–24 Feb 2018.

L Elisa Celis, Lingxiao Huang, Vijay Keswani, and Nisheeth K Vishnoi. Fair classification with noisy protected attributes: A framework with provable guarantees. In *International Conference on Machine Learning*, pp. 1349–1361. PMLR, 2021.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016. URL http://arxiv.org/abs/1603.02754.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic Decision Making and the Cost of Fairness. In *Proc. of the 23rd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining - KDD '17*, pp. 797–806, New York, New York, USA, jan 2017. ACM Press. ISBN 9781450348874. URL http://arxiv.org/abs/1701.08230http://dx.doi.org/10.1145/3097983.309809http://dl.acm.org/citation.cfm?doid=3097983.3098095.

Andrew Cotter, Maya Gupta, and Harikrishna Narasimhan. On making stochastic classifiers deterministic. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019a. URL https://proceedings.neurips.cc/paper/2019/file/5fc34ed307aac159a30d81181c99847e-Paper.pdf.

Andrew Cotter, Heinrich Jiang, Maya Gupta, Serena Wang, Taman Narayan, Seungil You, and Karthik Sridharan. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *Journal of Machine Learning Research*, 20(172):1–59, 2019b. URL http://jmlr.org/papers/v20/18-616.html.

André F. Cruz, Pedro Saleiro, Catarina Belém, Carlos Soares, and Pedro Bizarro. Promoting fairness through hyperparameter optimization. In *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 1036–1041. IEEE, 2021. doi: 10.1109/ICDM51629.2021.00119. URL https://doi.org/10.1109/ICDM51629.2021.00119.

Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2185–2194, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1224. URL https://aclanthology.org/D19-1224.

Dheeru Dua and Casey Graff. UCI Machine Learning Repository, 2017. URL http://archive.ics.uci.edu/ml.

Harrison Edwards and Amos J. Storkey. Censoring representations with an adversary. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.05897.

Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

Benjamin Fish, Jeremy Kun, and Ádám Dániel Lelkes. A confidence-based approach for balancing fairness and accuracy. *CoRR*, abs/1601.05764, 2016. URL http://arxiv.org/abs/1601.05764.

Yoav Freund and Robert E Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, pp. 148–156, 1996.

Vincent Grari, Boris Ruf, Sylvain Lamprier, and Marcin Detyniecki. Fair adversarial gradient tree boosting. In *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 1060–1065, 2019. doi: 10.1109/ICDM.2019.00124.

Moritz Hardt, Eric Price, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/9d2682367c3935defcb1f9e247a97c0d-Paper.pdf.

Vasileios Iosifidis and Eirini Ntoutsi. Adafair: Cumulative fairness adaptive boosting. *CoRR*, abs/1909.08982, 2019. URL http://arxiv.org/abs/1909.08982.

Faisal Kamiran and Toon Calders. Classifying without discriminating. In *2009 2nd International Conference on Computer, Control and Communication*, pp. 1–6, 2009. doi: 10.1109/IC4.2009.4909197.

Faisal Kamiran, Asim Karim, and Xiangliang Zhang. Decision theory for discrimination-aware classification. In *2012 IEEE 12th International Conference on Data Mining*, pp. 924–929, 2012. doi: 10.1109/ICDM.2012.45.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

Harikrishna Narasimhan, Andrew Cotter, and Maya Gupta. Optimizing generalized rate metrics with three players. *Advances in Neural Information Processing Systems*, 32, 2019.

J v Neumann. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320, 1928. doi: 10.1007/BF01448847.

Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/b8b9c74ac526fffbeb2d39ab038d1cd7-Paper.pdf.

Srinivasan Ravichandran, Drona Khurana, Bharath Venkatesh, and Narayanan Unny Edakunni. Fairxgboost: Fairness-aware classification in xgboost. *CoRR*, abs/2009.01442, 2020. URL https://arxiv.org/abs/2009.01442.

Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T Rodolfa, and Rayid Ghani. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577*, 2018.

Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *arXiv preprint arXiv:2106.03253*, 2021.

Divya Tomar and Sonali Agarwal. A survey on data mining approaches for healthcare. *International Journal of Bio-Science and Bio-Technology*, 5(5):241–266, 2013.

Xin Tong. A plug-in approach to neyman-pearson classification. *Journal of Machine Learning Research*, 14(56):3011–3040, 2013. URL http://jmlr.org/papers/v14/tong13a.html.

Alexander Vargo, Fan Zhang, Mikhail Yurochkin, and Yuekai Sun. Individually fair gradient boosting. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=JBAa9we1AL.

Blake Woodworth, Suriya Gunasekar, Mesrob I. Ohannessian, and Nathan Srebro. Learning non-discriminatory predictors. In Satyen Kale and Ohad Shamir (eds.), *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pp. 1920–1953. PMLR, 07–10 Jul 2017. URL https://proceedings.mlr.press/v65/woodworth17a.html.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P. Gummadi. Fairness Constraints: Mechanisms for Fair Classification. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 962–970. PMLR, 20–22 Apr 2017. URL https://proceedings.mlr.press/v54/zafar17a.html.

Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 325–333, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/zemel13.html.

A BACKGROUND ON GRADIENT BOOSTED DECISION TREES

A gradient boosting algorithm estimates a mapping $f : \mathbf{X} \mapsto \mathbf{y}$ that minimizes a loss function,

$$L(f) = \frac{1}{N} \sum_{i=1}^{N} l(y_i, f(x_i)), \tag{10}$$

where f is constrained to be a sum of base (weak) learners $h_t \in \mathcal{H}$. In the case of GBDT, these can be shallow decision trees with fixed depth or fixed number of nodes:

$$f = \sum_{t=0}^{T} \eta_t h_t, \tag{11}$$

where η_t is a step size parameter. Typically, h_0 would be a constant function that minimizes L and $\eta_0 = 1$. Gradient boosting can then be understood as performing gradient descent on the space of functions f. Each subsequent step, h_t, being essentially a projection onto $\mathcal{H}$ of the negative gradient of the loss L w.r.t. f. In other words, the base learner whose predictions are as close as possible, in the l_2 sense, to the negative gradient[2]:

$$h_t = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^{N} \left(-g_{t,i} - h(x_i)\right)^2, \tag{12}$$

where $g_{t,i}$ are the gradients evaluated at the current iterate $f_{t-1} = \sum_{m=0}^{t-1} \eta_m h_m$:

$$g_{t,i} = \left[\frac{\partial l(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i)=f_{t-1}(x_i)}. \tag{13}$$

Note that Equation 12 is equivalent to:

$$h_t = \arg\min_{h_t \in \mathcal{H}} \sum_{i=1}^{N} \left[g_{t,i} h_t(x_i) + \frac{1}{2} h_t^2(x_i)\right]. \tag{14}$$

XGBoost (Chen & Guestrin, 2016) and LightGBM (Ke et al., 2017) replace the approximation above with a local quadratic one thus implementing the following second order step:

$$h_t = \arg\min_{h_t \in \mathcal{H}} \sum_{i=1}^{N} \left[g_{t,i} h_t(x_i) + \frac{1}{2} H_{i,t} h_t^2(x_i)\right] + \Omega(h_t), \tag{15}$$

where $H_{i,t}$ is the hessian of l w.r.t. f computed at the current iterate and Ω is a regularization term penalizing complex base learners.

B DISCUSSION ON THE LIMITATIONS OF FAIRGBM

In this section, we discuss expected limitations of the proposed approach.

When compared with vanilla LightGBM, FairGBM requires approximately double the training time (see Table 2). This compute overhead of FairGBM can be attributed to (1) the computation of the proxy gradients relative to the constraint loss, and (2) the addition of an extra optimization step per iteration – the ascent step.

From an application standpoint, FairGBM is specifically tailored for gradient boosting methods. This contrasts with other bias mitigation methods in the literature such as EG (Agarwal et al., 2018), which, despite having considerably slower runtime, are applicable to a wider range of classifier types (all binary classifiers in the case of EG).

[2] In practice, a heuristic is used that builds the decision tree by greedily choosing a sequence of splitting variables and splitting values that most decrease the value of the function to minimize

Despite discussing several choices for differentiable proxies for the Type 1 and Type 2 errors in Section 2.2, our experiments only concern one of these proxies. As future work, we would like to perform a more thorough study on the impact of different choices of differentiable proxies for the step-wise function (*e.g.*, sigmoid, hinge, or squared loss).

Moreover, FairGBM as it is devised in this work is limited to group-wise fairness metrics, and incompatible with metrics from the individual fairness literature. In fact, individual fairness algorithms tailored for GBM have been developed in related work Vargo et al. (2021). The BuDRO method proposed by Vargo *et al.* is based on optimal transport optimization with Wasserstein distances, while our proposal fully relies on gradient-based optimization with the dual ascent method. A possible extension of our method to a wider range of constraints that enables individual fairness is a topic we aim to explore in future work.

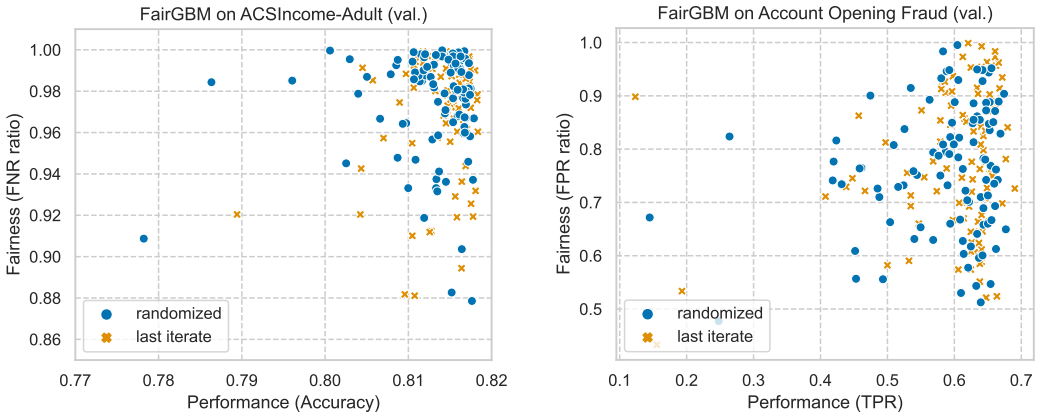
C OPERATING FAIRGBM AT A SPECIFIC ROC POINT

Without fairness constraints, meeting a specific ROC operating point can be achieved by appropriately thresholding the output of a classifier that learns to approximate the class probabilities: $p(y = 1|x)$ (Tong, 2013). That is the case of a classifier that is trained to minimize a proper scoring rule such as binary cross-entropy.

However, when optimizing for the Lagrangian (Equation 7), we are no longer optimizing for a classifier that approximates the true class probabilities. This is a key point that is often overlooked in the constrained optimization literature. Namely, both $\mathcal{L}$ and $\tilde{\mathcal{L}}$ have an implicit threshold when evaluating constraint fulfillment: the 0.5 decision threshold, or 0.0 when considering the log-odds (see Figure 1). In practice, FairGBM will be optimized to fulfill the constraints at this pre-defined threshold, but constraint fulfillment may not (and likely will not) generalize to all thresholds. Indeed, we could use any decision threshold during training, but it is impossible to know which threshold would meet our ROC requirements beforehand.

We propose to solve this by introducing our ROC requirement as another in-training constraint. In the AOF setting, this is achieved by introducing an additional constraint of global $FPR \leq 0.05$. In practice, instead of shifting the threshold to meet our target ROC operating point, we are shifting the score distribution such that the 0.5 decision threshold corresponds to our target ROC point, enabling constraint fulfillment at any attainable ROC point.

D RANDOMIZED CLASSIFIER *vs* LAST ITERATE



(a) ACSIncome results.

(b) AOF results.

Figure 3: Comparison between using the FairGBM randomized classifier (blue circles) or the predictions of the last FairGBM iterate (orange crosses).

Figure 3 shows a comparison between using the randomized classifier predictions or solely the last iterate predictions on the ACSIncome dataset. As mentioned in Section 2, the first approach benefits from theoretical convergence guarantees, while the latter benefits from being a deterministic classifier (which may be a requirement in some real-world settings). In practice, using the last iterate version of FairGBM (which always uses all trees of the GBM ensemble) achieves similar results to the randomized classifier version (which randomly picks the number of trees it will use for each prediction). The same trend is clear on the AOF dataset, and concurs with related work on randomized classifiers by Cotter et al. (2019b).

E FAIRNESS-PERFORMANCE TRADE-OFF

To better understand the trade-offs between fairness and performance, we present two groups of plots. The first group of plots in Figures 4a and 4b have each method evaluated over the possible values of α (x-axis). The metric of evaluation is the fairness-performance composed metric, weighted by α (y-axis). Higher values of this metric represent better results.

For this plot, with a step of 0.01 in α, we bootstraped 20 random models of each method and selected the one that optimized the α metric. This process was repeated 1000 times, in order to obtain the average value and the confidence interval estimations.

We observe that for these two datasets, the FairGBM method represents a better trade-off in the composed metric, being only beated in a pure performance scenario ($\alpha = 1$) by LightGBM and a pure fairness scenario ($\alpha = 0$) by Fairlearn and RS Reweighting, this latter only being observed in the AOF dataset. This shows that FairGBM is a sensible choice for scenarios where both Fairness and Performance play an important role in model selection. Additionally, this method also has low variance in the observed metrics. This means that for these datasets, it is possible to obtain a model that is close to the best attainable performance and fairness, with 20 randomly sampled models of FairGBM, whereas other methods have a higher chance of obtaining either underperforming or biased models.

In figures 5 and 6, instead of using an α weighted metric, we separate performance and fairness. In these plots, we observe that FairGBM presents the best results all-around. For the ACS Adult dataset, FairGBM obtains models that reach the best values of performance with a nearly perfect fairness result, independently of the selected α. Other algorithms have mainly lower fairness results. The EG method obtains comparable results. Regarding the AOF dataset, FairGBM presents a similar performance value to LightGBM with overall better fairness results. Other algorithms fail to generate good predictions, with recall values rounding 20% to 40%.

Algorithm	**Trade-off** $\alpha = 0.50$			
	Validation		**Test**	
	Fair. (%)	**Perf. (%)**	**Fair. (%)**	**Perf. (%)**
ACSIncome-Adult dataset				
FairGBM	99.6 ± 0.73	81.6 ± 0.08	99.4 ± 0.74	81.7 ± 0.09
LightGBM	75.6 ± 3.28	80.8 ± 0.80	75.5 ± 3.32	80.8 ± 0.80
GS	66.4 ± 1.49	81.7 ± 0.19	65.9 ± 1.36	81.8 ± 0.20
RS	42.3 ± 25.8	77.0 ± 3.70	42.4 ± 25.8	77.0 ± 3.73
EG	94.4 ± 0.32	81.6 ± 0.16	93.8 ± 0.11	81.6 ± 0.18
AOF dataset				
FairGBM	92.2 ± 4.19	64.8 ± 2.54	90.1 ± 4.51	64.7 ± 2.59
LightGBM	70.2 ± 7.92	55.8 ± 3.97	78.4 ± 9.89	55.8 ± 3.36
GS	98.9 ± 0.82	23.4 ± 3.71	98.7 ± 1.37	23.4 ± 3.87
RS	95.3 ± 3.55	31.2 ± 4.31	96.1 ± 3.47	31.3 ± 4.53

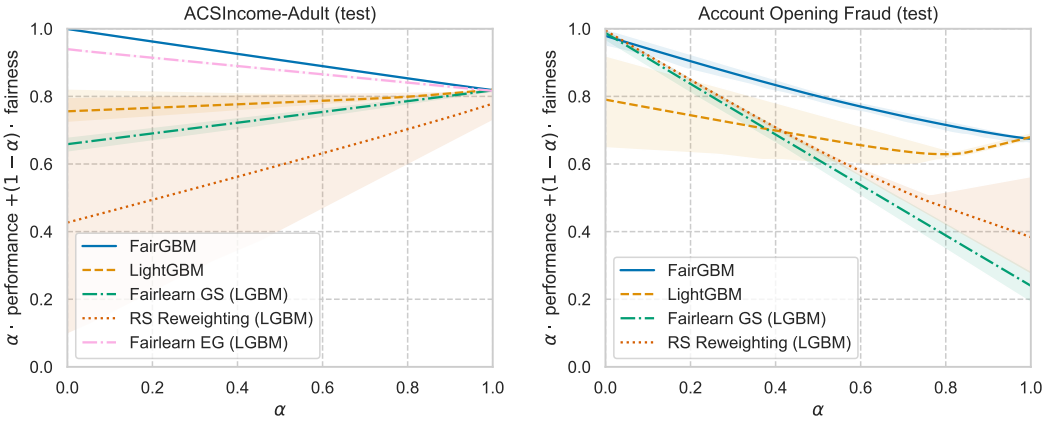
Table 3: Results for mean and standard deviation for all models on each dataset, with target trade-off $\alpha = 0.50$. Model selection metric was $[\alpha \cdot \textit{performance} + (1 - \alpha) \cdot \textit{fairness}]$. The best model is selected on validation data, and results are reported on both validation and test data.

Algorithm	Trade-off $\alpha = 0.95$			
	Validation		Test	
	Fair. (%)	Perf. (%)	Fair. (%)	Perf. (%)
ACSIncome-Adult dataset				
FairGBM	98.7 ± 1.07	81.8 ± 0.06	98.5 ± 1.22	81.8 ± 0.06
LightGBM	71.2 ± 0.30	81.9 ± 0.04	70.8 ± 0.34	82.0 ± 0.05
GS	66.3 ± 1.54	81.8 ± 0.14	65.8 ± 1.40	81.8 ± 0.14
RS	38.3 ± 25.6	77.8 ± 2.96	38.4 ± 25.6	77.8 ± 2.97
EG	94.4 ± 0.36	81.6 ± 0.15	93.9 ± 0.13	81.6 ± 0.16
AOF dataset				
FairGBM	80.0 ± 9.79	66.6 ± 0.88	80.2 ± 9.13	66.6 ± 1.06
LightGBM	33.7 ± 1.70	67.6 ± 0.47	36.0 ± 1.29	67.3 ± 1.01
GS	98.5 ± 0.99	23.6 ± 3.44	98.3 ± 1.74	23.8 ± 3.51
RS	81.5 ± 21.2	37.6 ± 9.15	82.2 ± 23.2	38.2 ± 9.64

Table 4: Results for mean and standard deviation for all models on each dataset, with target trade-off $\alpha = 0.95$. Model selection metric was $[\alpha \cdot \textit{performance} + (1 - \alpha) \cdot \textit{fairness}]$. The best model is selected on validation data, and results are reported on both validation and test data.

Algorithm	Run-time	
	Total (h)	Relative
ACSIncome-Adult dataset		
FairGBM	9.9	x2.1
LightGBM	4.6	*baseline*
GS	43.8	x9.6
RS	37.1	x8.1
EG	99.4	x21.7
AOF dataset		
FairGBM	3.5	x2.4
LightGBM	1.4	*baseline*
GS	21.4	x14.7
RS	10.3	x7.1

Table 5: Total runtimes (in hours) for training all models in the experiments section, per algorithm type and per dataset. Also shown is the relative increase versus the fastest algorithm (standard LightGBM).



(a) ACSIncome results.

(b) AOF results.

Figure 4: Plot of the attainable fairness-performance trade-offs for each algorithm, as a function of $\alpha \in [0.0, 1.0]$. Lines show the mean value, and shades show 95% confidence intervals. FairGBM achieves the best trade-off for $\alpha \in [0.00, 0.99]$ on the ACSIncome dataset (left plot), and for $\alpha \in [0.05, 0.98]$ on the AOF dataset (right plot).

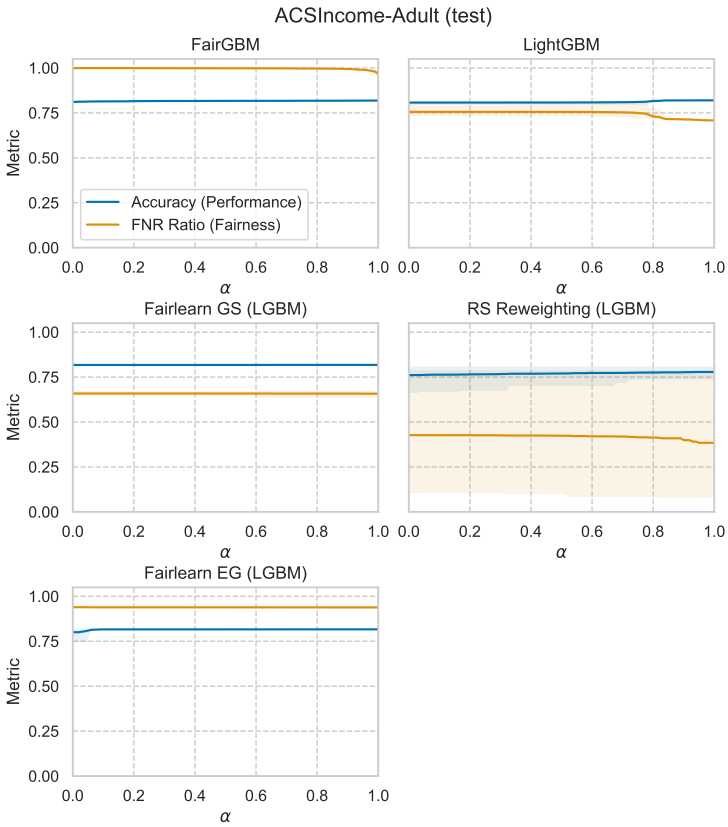


Figure 5: Plots for each separate algorithm of average performance and fairness obtained as a function of $\alpha \in [0.0, 1.0]$, on the ACSIncome test set. These plots further detail the results from Figure 4a by splitting the α-weighted metric into its two components, performance and fairness.

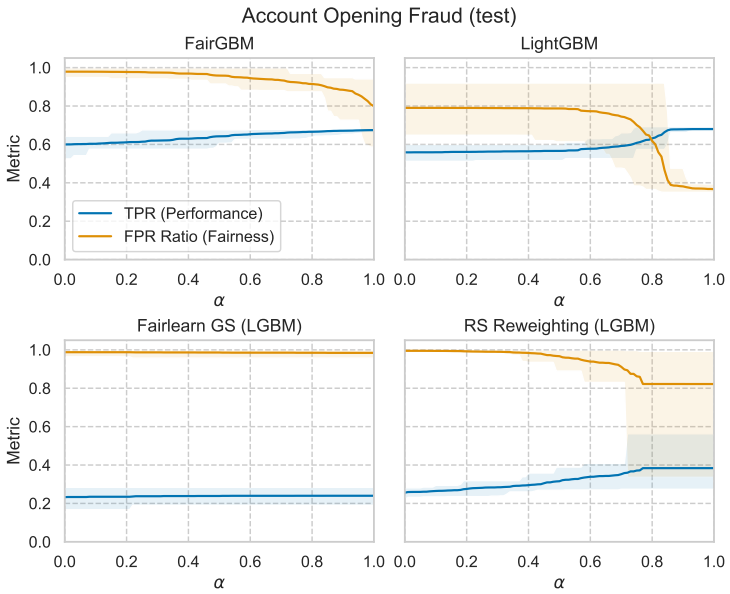


Figure 6: Plots for each separate algorithm of average performance and fairness obtained as a function of $\alpha \in [0.0, 1.0]$, on the AOF test set. These plots further detail the results from Figure 4b by splitting the α-weighted metric into its two components, performance and fairness.

F Account Opening Fraud Dataset

The Account Opening Fraud (AOF) dataset is an in-house dataset that comprises 8 months of data from a real-world fraud detection task. Specifically, on the detection of fraudulent online bank account opening applications in a large European consumer bank. In this setting, fraudsters attempt to either impersonate someone via identity theft, or create a fictional individual in order to be approved for a new bank account. After being granted access to a new bank account, the fraudster quickly maxes out the line of credit, or uses the account to receive illicit payments. All costs are sustained by the bank.

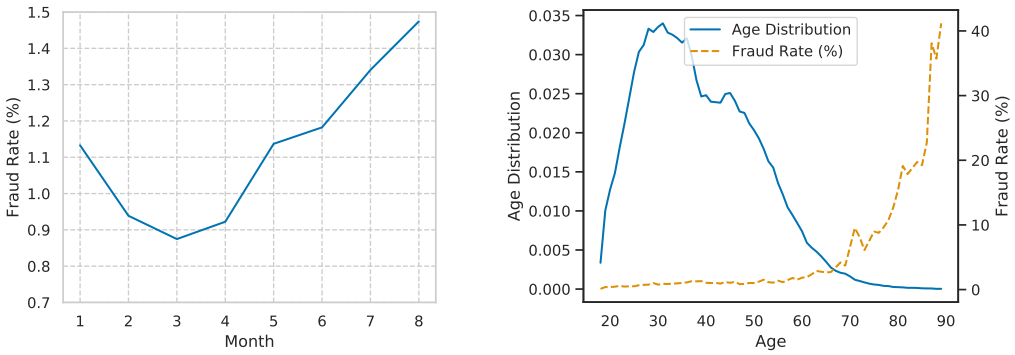
Each instance (row) of the dataset represents an individual application. All of the applications were made on an online platform, where explicit consent to store and process the gathered data was granted by the applicant. Each instance is labeled after a few months of activity, as by then it is apparent whether the account owner is fraudulent (positive label) or legitimate (negative label). A total of 67 features are stored for each applicant, 6 being categorical, 19 binary, and 42 numerical. Most features represent aggregations over the original information and context of the application (*e.g.*, count of the number of applications in the last hour).

The prevalence of fraud varies between 0.85% and 1.5% during the eight month period (see Figure 7a). We observe that these values are higher for the later months, which were used as validation (the 7th, or second-to-last month) and test (the 8th, or last month) data. Additionally, the distribution of applications also changes from month to month, ranging from 9.5% on the lower end to 15% on the higher end. The extremely high class imbalance (approximately 1 positive label for each 99 negative labels), the gradual change in fraud prevalence, together with naturally shifting consumer patterns along 8 months of real-world data are examples of common real-world challenges that are not often present in datasets from the fairness literature.

Due to the low fraud rate in the data, measuring accuracy of the models would lead to misleading results. A trivial classifier that constantly outputs legitimate predictions would achieve an accuracy close to 99%. To address this, a commonly used performance metric in the fraud detection industry is the true positive rate (TPR, or Recall), as it reflects the percentage of fraud that was caught by the ML model. Moreover, in order to keep attrition on legitimate customers low, a requirement of at most 5% false positive rate (FPR) is used, *i.e.*, at most 5% of legitimate (label negative) customers are wrongly blocked from opening a bank account. Additionally, due to the punitive nature of the classification

setting (a false positive negates financial services to a legitimate applicant), we aim to balance false positive rates between the different groups in the dataset.

As a protected attribute for this dataset, we selected the applicant's age. Specifically, we divide applicants into two groups: under 50 years old, and at or above 50 years old. There is a surprising but clear relation between fraudulent applications and reported age, so we expect that fairness w.r.t. the age group will make for a challenging setting. Figure 7b shows a plot of the age distribution, as well as the variation of fraud rate over age values.



(a) Fraud Rate by Month.

(b) Fraud Rate by Age.

Figure 7: Plot of the variation of Fraud Rate depending on the month of the application and applicant's age. The plot also contains distribution of Age over all the applications in the dataset.

G NOTATION

$\mathcal{L}$	the Lagrangian function, which uses the original constraints c; see Equation 2.
$\tilde{\mathcal{L}}$	the proxy-Lagrangian function, which uses the proxy constraints $\tilde{c}$; see Equation 7.
c	an inequality constraint function; it is deemed fulfilled if $c(\theta) \leq 0$; this function may be non-differentiable; examples include a constraint on TPR parity or parity of any other metric of the confusion matrix.
$\tilde{c}$	a proxy inequality constraint that serves as sub-differentiable proxy for the corresponding constraint c; see Equation 5.
l	an instance-wise loss function, i.e., $l : \mathcal{Y} \times \hat{\mathcal{Y}} \mapsto \mathbb{R}_+$, where $\mathcal{Y}$ is the set of possible labels and $\hat{\mathcal{Y}}$ is the set of possible predictions; see green line in Figure 1.
$\tilde{l}$	a sub-differentiable proxy for an instance-wise loss function; see blue and purple lines in Figure 1.
D	a dataset of samples $(x, y, s) \in D$, where $x \in \mathcal{X} \subseteq \mathbb{R}^n$ is the features, $y \in \mathcal{Y} \subseteq \mathbb{N}_0$ is the label, and $s \in \mathcal{S} \subseteq \mathbb{N}_0$ is the sensitive attribute.
$L_{(S=a)}$	a predictive loss function measured over data samples with sensitive attribute value $S = a$, $\{(x, y, s) : s = a, (x, y, s) \in D\}$; the subscript is omitted when measuring loss over the whole dataset D; examples include the false-negative rate or the squared error loss.
$\tilde{L}_{(S=a)}$	a sub-differentiable proxy for a predictive loss function measured over data samples with sensitive attribute value $S = a$, $\{(x, y, s) : s = a, (x, y, s) \in D\}$; the subscript is omitted when measuring loss over the whole dataset D.
λ_i	a Lagrange multiplier associated with constraint c_i and proxy constraint $\tilde{c}_i$.
$\mathcal{F}$	the space of strong learners.
$\mathcal{H}$	the space of weak learners.
f	a strong learner.
h	a weak learner.
$\mathcal{S}$	the range of random variable S; the letter S specifically is used for the sensitive attribute, and $\mathcal{S}$ for the different values the sensitive attribute can take.

H REPRODUCIBILITY CHECKLIST

This section provides further details regarding implementation and hardware used for our experiments. We follow the reproducibility checklist put forth by Dodge et al. (Dodge et al., 2019).

Regarding reported experimental results:

☑ Description of computing infrastructure.
- i7-8650U CPU, 32GB RAM.

☑ Average run-time for each approach.
- Folder `runtimes` of the supp. materials[3].

☑ Details of train/validation/test splits.
- See Section 3, and data notebooks in folder `notebooks` of the supp. materials[3].

☑ Corresponding validation performance for each reported test result.
- Folder `results` of the supp. materials[3].

☑ A link to implemented code[1].

Regarding hyperparameter search:

☑ Bounds for each hyperparameter.
- Folder `hyperparameters` of the supp. materials[3].

☑ Hyperparameter configurations for best-performing models.
- Folder `hyperparameters` of the supp. materials[3].

☑ Number of hyperparameter search trials.
- 100 trials.

☑ The method of choosing hyperparameter values.
- Random uniform sampling.

☑ Expected validation performance.
- Folder `others` of the supp. materials[3].

[3] `https://anonymous.4open.science/r/fairgbm-anonymous`