Project Report

Re-weighting the Dataset to Obtain a Bias-mitigated Model

Author: Qingyang Liu, Louis Wang
Supervisor: Prof.Yuekai Sun, Prof.Moulinath Banerjee

08/12/2022

1 Background Introduction

1.1 Racial Bias in Prediction Algorithms

A typical application of causal-inference machine learning problem is the prediction policy problem in the healthcare field, where commercial prediction systems use past patient data to evaluate their future healthcare needs. Researchers, however, discovered a non-negligible racial bias embedded in a current wide-using prediction algorithm. In the paper *Dissecting racial bias in an algorithm used to manage the health of populations* [Obermeyer et al., 2019] the authors focused on a live, scaled algorithm deployed nationwide and applied to roughly 200 million people in the United States. Based on the assumption that the treatment effect of a certain healthcare program is monotonic in the patient's health risk level, this prediction algorithm aimed at identifying the top 3 percent people with the greatest health risk to be automatically enrolled into the program, and also the top 45 percent people to be considered as candidates of enrollment.

Researcher argued that the usage of an artificial health risk score is problematic, since there exists health disparities conditional on risk score for two racial groups: White and Black. At the same level of algorithm-predicted risk, Blacks have significantly more illness burden than Whites. There is, however, little disparities in healthcare costs conditioned on predicted risk score.

The reason is inherent in how the prediction algorithm is designed. The algorithm takes in raw insurance claims data, such as demographics, insurance type, diagnosis and procedure codes, medications, and detailed costs. The chosen label is health costs per year instead of actual healthcare needs. The algorithm specifically excludes race. While this might make the algorithm seem unbiased, reality is a bit more complicated. Relevant literature suggests that Blacks on average generate lower costs than White given similar health conditions due to various socioeconomic factors. This shows that costs per year and race are not independent variables, which suggests that the algorithm is trained on an inherently biased label.

1.2 Authors' Potential Solution

The author suggests that a careful choice of label on which we train the prediction algorithm matters. By finding a less biased label, we can obtain less biased results. The author conducted experiments on the same dataset but with different labels, such as avoidable costs and active chronic conditions. Results of the experiments show that the predicted fraction of Blacks over the 97 percentile of risk score using active chronic conditions (26.7%) as the training label is the closest to the actual fraction, which is approximately 47%. This indicates that active chronic conditions is a less biased label compared with total cost and total avoidable cost. We should also notice that this result is far from an optimal solution, and it could be difficult to find or synthesize a label that minimizes the racial bias. Therefore we tried to explore a more efficient approach to resolve this problem, as we would discuss in the rest of our report.

2 Method We Used

2.1 Overview

The method that we are using is adopted from a paper by Heinrich Jiang and Ofir Nachum [Jiang & Nachum, 2020]. The paper aims to explain why biases are embedded in certain machine learning algorithms and to correct the biases without changing the dataset. The core method assumes that a set of unknown and unbiased labels exists and is overwritten by a biased agent who intends to provide accurate labels. The goal of this method is to obtain an unbiased machine learning classifier by re-weighting the data points without changing the labels. We will discuss later in this section that the re-weighting technique in this paper guarantees that training on the re-weighted dataset with the biased labels is equivalent to training with the unbiased labels that are unknown to us. We will also show that in the experiment section that on our dataset this method gives a fairly good fairness-accuracy trade-off compared to the baseline models.

2.2 Understanding Bias

In order to have at a precise understanding of bias, we need to first define our notion of fairness. We will use Jiang's and Nachum's definition of fairness. First, let X be the set of all data points where $X \sim P$ and $Y := \{0, 1\}$ be the labels. Then, let $h : X \to [0, 1]$ be our machine learning model and $c : X \times Y \to \mathbb{R}$ be our fairness constraint function. The definition of fairness is the following:

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

where $\langle h(x), c(x) \rangle := \sum_{y \in Y} h(y|x)c(x, y)$, $x \in X$ and $y \in Y$. More specifically, $h(y|x)$ follows a Bernoulli distribution where $h(x) = h(1|x)$ and $h(0|x) = 1 - h(1|x)$. This notion of fairness allows us to measure the degree of bias.

Given this notion of fairness, we can effectively model bias in a dataset. Before that, we need to establish some necessary preconditions. First, according to Jiang and Nachum,

we must start with a necessary assumption on the relationship between the set of biased labels and the set of unbiased labels: Suppose there are k protected groups with $c_1, c_2, \ldots, c_k$ being their respective fairness constraint functions. This means that according to our definition of fairness, $\mathbb{E}_{x\sim P}[\langle y_{true}(x), c_i(x)\rangle] = 0$ for $i = 1, \ldots, k$ where y_{true} is the unbiased model. Our assumption is that there exist $\epsilon_1, \ldots \epsilon_k \in \mathbb{R}$ such that y_{bias} solves the follwing constrained optimization problem:

$\underset{\hat{y}:X\to 0,1}{\operatorname{argmin}} \mathbb{E}_{x\sim P}[D_{KL}(\hat{y}(x)||y_{true}(x))]$
s.t. $\mathbb{E}_{x\sim P}[\langle \hat{y}(x), c_i(x)\rangle] = \epsilon_i$, for $i = 1, \ldots k$, with y_{bias} being the observed, biased labeling function, and D_{KL} being the KL-divergence.

In other words, this assumption is saying that there exists a function $\hat{y}$ such that it achieves our desired level of bias as defined by $\epsilon_1, \ldots, \epsilon_k$ while being as accurate as possible as measured by the KL-divergence.

This assumption leads to an important result, which we will now derive:

Suppose $\hat{y}(x) \sim Bernoulli$ s.t. $\hat{y}(x) = \hat{y}(1|x)$ and $\hat{y}(0|x) = 1 - \hat{y}(1|x)$. Then,

$$D_{KL}(\hat{y}(x)||y_{true}(x)) = Ber(\hat{y}(x)) \cdot log(\tfrac{\hat{y}(x)}{y_{true}(x)})$$

Therefore,

$$\mathbb{E}_{x\sim P}(D_{KL}(\hat{y}(x)||y_{true}(x))) = \sum_{i=1} P(x_i) \cdot Ber(\hat{y}(x_i)) \cdot log(\tfrac{\hat{y}(x_i)}{y_{true}(x_i)}), \text{ for } x_i \in X.$$

The constraint in the above constrained optimization problem can be simplified as

$$\mathbb{E}_{x\sim P}[\langle \hat{y}(x), c(x)\rangle] = \sum_{i=1} P(x_i) \cdot \hat{y}(x_i) c_j(x_i) \text{ for } c_j(x) \in c_1(x), \ldots, c_k(x).$$

The above simplifications allow us to reformulate the aforementioned constrained optimization function as

$$\min_{\hat{y}} \sum_{i=1} P(x_i) \cdot Ber(\hat{y}(x_i)) \cdot log(\tfrac{\hat{y}(x_i)}{y_{true}(x_i)}) \text{ s.t. } \sum_{i=1} P(x_i) \cdot \hat{y}(x_i) c_j(x_i) = 0 \text{ for } c_j(x) \in c_1(x), \ldots, c_k(x).$$

We can rewrite the problem in the Lagrange form:

$$L(\hat{y}, \lambda_1, \ldots, \lambda_k) = \sum_{i=1} P(x_i) \cdot Ber(\hat{y}(x_i)) \cdot log(\tfrac{\hat{y}(x_i)}{y_{true}(x_i)}) + \lambda_1(\sum_{i=1} P(x_i) \cdot \hat{y}(x_i) c_1(x_i)) + \cdots + \lambda_k(\sum_{i=1} P(x_i) \cdot \hat{y}(x_i) c_k(x_i)).$$

After taking the derivative of $L(\hat{y}, \lambda)$ with respect to $\hat{y}(x_i)$, we get that

$$\hat{y}(x_i) \propto y_{true}(x_i) \exp(-\lambda_1 c_1(x_i) - \cdots - \lambda_k c_k(x_i)).$$

This gives us the final corollary:

$$y_{true}(y|x) \propto \hat{y}(y|x) \exp(\lambda_1 c_1(x, y), \ldots, \lambda_k c_k(x, y)).$$

This result states that given the assumption, we can express the unbiased and unknown label function in terms of the observed and biased label function, the coefficients $\lambda_1, \ldots, \lambda_k$, and the constraint functions $c_1, \ldots, c_k$ given k protected groups. The constrained optimization function does not become our training objective. And without introducing an additional training objective, our model, previously defined as $h : X \to [0, 1]$, is more simple and can be more accurate compared to traditional Lagrangian approaches that takes the fairness constraints as penalties where certain relaxation of the constraints must be taken. In this method, the constrained optimization function helps us understand theoretically the relationship between y_{true} and $\hat{y}$ in our model. and when it comes to the practical aspect of model training, we would further propose a weighting algorithm based on the relationship we have derived, as we will discuss below.

2.3 Training Classifier

Now that we figure out the relationship between y_{true} and y_{bias}, a weighting algorithm is required to transform the current observations with biased labels into a redistributed counterpart with unbiased labels. Here Jiang and Nachum define the weight of any observation $w(x, y) = \tilde{w}(x, y)/\sum_{y'\in Y} \tilde{w}(x, y')$, where $\tilde{w}(x, y) = exp\{\sum_{k=1}^{K} \lambda_k c_k(x, y')\}$. Based on this definition, Jiang and Nachum show an important theorem:

> training classifier h on a **unweighted** objective $E_{x\sim\tilde{P}, y\sim y_{true}(x)}[l(h(x), y]$ over a **shifted distribution** $\tilde{P}$ is equivalent to training h on a **weighted objective** $x \sim P, y \sim y_{bias}(x)[w(x, y) \cdot l(h(x), y]$ on the **original distribution** P over X.

Based on the brief instruction in [Jiang & Nachum, 2020], we complete a more detailed proof for this theorem as below:

Given

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

that we have derived in 2.2, we could substitute $\tilde{w}(x, y) = \exp\{\sum_{k=1}^{K} \lambda_k c_k(x, y)\}$ into **(1)**, so we get:

$y_{true}(y|x) \propto y_{bias}(y|x) \cdot \tilde{w}(x, y)$ **(2)**.

Since $\tilde{w}(x, y) = w(x, y) \cdot \sum_{y'\in Y} \tilde{w}(x, y')$,

we could further get:

$y_{true}(y|x) \propto y_{bias}(y|x) \cdot w(x, y) \cdot \sum_{y'\in Y} \tilde{w}(x, y')$ **(3)**

Divide $\sum_{y'\in Y} \tilde{w}(x, y')$ on both sides, we get:

$\frac{y_{true}(y|x)}{\sum_{y'\in Y} \tilde{w}(x,y')} \propto y_{bias}(y|x) w(x, y)$ **(4)**

substitute $\tilde{w}(x, y') = exp\{\sum_{k=1}^{K} \lambda_k c_k(x, y')\}$ into **(1)** again, we are able to get:

$\phi(x) \cdot y_{true}(y|x) = y_{bias}(y|x) w(x, y)$ **(5)**

where $\phi(x) = \sum_{y'\in Y} w(x, y') y_{bias}(y'|x)$.

Notice that $\phi(x)$ only depends on the value of x since it takes a summation concerning all possible values of y. In this case, we could define a shifted distribution $\tilde{P}$ on X s.t. $\tilde{P} \propto \phi(x)P(x)$ **(6)**. The weighted objective on the original distribution P can be written as $E_{x\sim\tilde{P}, y\sim y_{true}(x)}[l(h(x), y]$, where $P(x) and y_{bias(y|x)}$ is the probabiity density function

of x and y respectively. By the definition of expectation,

$E_{x\sim\tilde{P},y\sim y_{true}(x)}[l(h(x),y]$
$= \int l[(h(x),y]\cdot w(x,y)y_{bias}(y|x)P(x)dxdy$
$= \int l[(h(x),y]\cdot \phi(x)\cdot y_{true}(y|x)P(x)dxdy$ (By **(5)**)
$= \int l[(h(x),y]\cdot y_{true}(y|x)\cdot(\phi(x)P(x))dxdy$
$\propto \int l[(h(x),y]\cdot y_{true}(y|x)\tilde{P}(x)dxdy$ (By **(6)**)
$= E_{x\sim\tilde{P},y\sim y_{true}(x)}[l(h(x),y]$

Therefore, the proof is completed.

Up to now, we have made it clear why a weighting algorithm is theoretically reasonable in training an unbiased classifier model. The problem now falls onto picking the right parameters which further determine the weight $w(x,y)$. Obviously, the fairness constraints $c_1,\ldots,c_k$ and the Lagrangian coefficients $\lambda_1,\ldots,\lambda_k$ matter.

2.4 Determining Coefficients λ

Since we have restricted the constraints $c_1, c_2, ..., c_k$ in the form of demographic parity, these constraint functions is already known to us. The only job is to determine the coefficients $\lambda_1,\ldots,\lambda_k$.

Because the distribution of X is being modified while we are re-weighting each sample points corresponding to their label values. A fairness-accuracy trade-off arises here. While re-weighting the biased labels leads to an improved fairness, in other words, an increased prediction rate for the protected class, training over a different distribution leads to an overall decreased accuracy on predictors other than the protected class. In order to find the optimized values of the coefficients for this trade-off, we will adopt an iterative method that Jiang and Nachum implemented[Jiang & Nachum, 2020], and the pseudocode is provided below. This algorithm adopts demographic parity as the fairness constraint as we presume earlier, thus, if the positive prediction rate for a protected group is lower than the overall positive prediction rate, the corresponding coefficient would increase in its value. This means that we increase the weights of the positively labeled data points of the protected group and decrease the weights of the negatively labeled data points, which would cause the model to provide more accurate predictions on the positively labeled observations and less accurate predictions on the negatively labeled observations.

Inputs: Learning rate η, number of loops T, training data $\mathcal{D}_{[n]} = \{(x_i,y_i)\}_{i=1}^N$, classification procedure H. constraints $c_1,...,c_K$ corresponding to protected groups $\mathcal{G}_1,...,\mathcal{G}_K$.
Initialize $\lambda_1,...,\lambda_K$ to 0 and $w_1 = w_2 = \cdots = w_n = 1$.
Let $h := H(\mathcal{D}_{[n]}, \{w_i\}_{i=1}^n)$
for $t = 1,...,T$ **do**
  Let $\Delta_k := \mathbb{E}_{x\sim\mathcal{D}_{[n]}}[\langle h(x), c_k(x)\rangle]$ for $k\in[K]$.
  Update $\lambda_k = \lambda_k - \eta\cdot\Delta_k$ for $k\in[K]$.
  Let $\widetilde{w_i} := \exp\left(\sum_{k=1}^K \lambda_k\cdot 1[x\in\mathcal{G}_k]\right)$ for $i\in[n]$
  Let $w_i = \widetilde{w_i}/(1+\widetilde{w_i})$ if $y_i = 1$, otherwise $w_i = 1/(1+\widetilde{w_i})$ for $i\in[n]$
  Update $h = H(\mathcal{D}_{[n]}, \{w_i\}_{i=1}^n)$
end for
Return h

In summary, this algorithm finds the optimal weights iteratively by performing the demographic parity fairness constraint on the dataset; updating the coefficients by subtracting the constraint violation times a fixed step size; computing the weights based on the given coefficients as derived previously; retrain the classifier given the new weights with the classifier being any weighted loss function.

This algorithm can also be applied to other notions of fairness, such as equal opportunity and and disparate impact, with the only change being the fairness constraint functions $c_1,\ldots,c_k$.

3 Experiments

3.1 Dataset

The dataset we primarily works on is inherited from [Obermeyer et al., 2019]. To be specific, this dataset is collected from the health data of all primary care patients enrolled in risk-based contracts in a large academic hospital from 2013 to 2015. To the aim of simplicity, all patients with race other than white are marked as Black, as what their race identity are coded in this dataset.

The dataset groups several 'outcome' vectors as the potential labels, including the cost, health condition, program enrollment status and the commercial risk score at a given calendar year t. The remaining variables, which are indexed to the year prior to the outcomes aforementioned(t-1), are used primarily as predictors in our experiment.

3.2 Data Preprocessing

We mostly adopt the data preprocessing steps used in Obermeyer's paper. When the targeted dataset is synthesized, three labels are derived from the 'outcome' vectors, including the total cost at time t(**cost.total.t**), the avoidable cost at time t(**cost.avoid.t**), and the total number of active chronic illnesses at time t(**gagne.sum.t**). We take the log of cost.total.t and cost.avoid.t for the purpose that these two numerical labels are more evenly distributed on its range, and hopefully we could curtail adverse effects from any outliers. Figure 1, 2 shows how the two labels are distributed before and after taking the log.

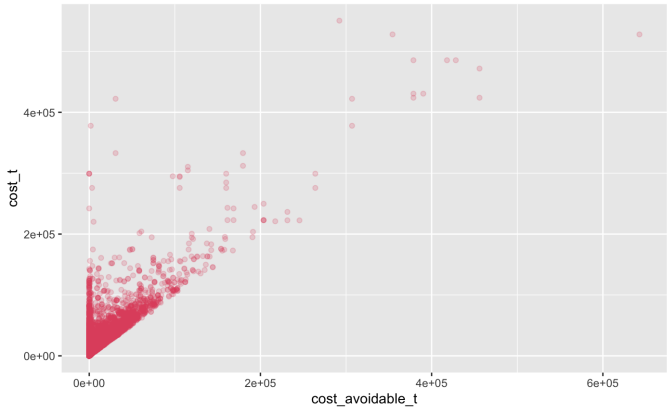

FIGURE 1: scatterplot of cost.total.t and cost.avoidable.t
Both cost indicators are extremely concentrated in low value.

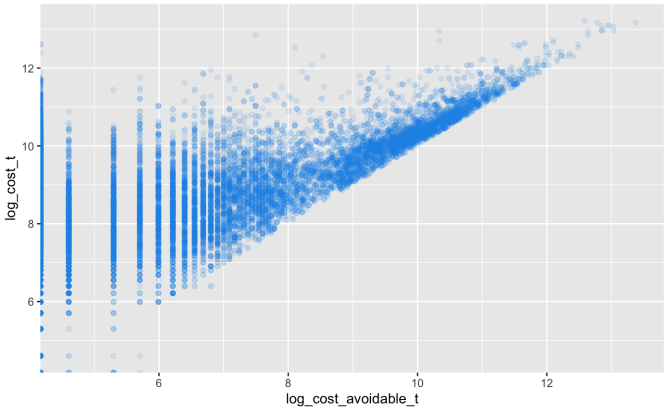

FIGURE 2: scatterplot of log.cost.total.t and log.cost.avoidable.t Both cost indicators are more evenly distributed.

In our experiment of applying this algorithm on the target dataset, we want to emulate the conditions in accordance with the assumptions on which this algorithm bases as much as possible. This requires us to create two protected groups in replacement of the single 'race' predictor – in this case, the white group and the black group – for later implementation of the re-weighting algorithm based on the demographic parity.

So far in our discussion of our method, we have presumed the labels to be binary labels. However, in the actual dataset, the potential labels we could adopt are all continuous variables. Therefore, we need to binarize the labels in order to properly implement this method. We binarize these three aforementioned labels according to specified quantile–55% and 97%–that have been predetermined in Obermeyer's paper as two important thresholds to determine whether the subject should be considered as a candidates of program enrollment. That is, for each 'label' vector, we convert all entries above the threshold value to 1, and all remaining entries to 0. After binarizing our labels, we can then feed our dataset into the algorithm.

3.3 Experimental Design

Before introducing the re-weighting algorithm into the model learning process, we need a baseline model without any modification on the predictors' distribution to see the performance concerning both prediction accuracy and fairness violation on our protected groups. As what has been implemented in Jiang's approach, we basically replicate their code of model training using Scikit-Learn's logistic regression. The hyperparameters are remained under default settings.

Then we involve the iterative re-weighting algorithm into the logistic regression to mitigate in-process fairness violations. In the practical implementation of Algorithm 1, we settle the number of iteration to be 100, and the learning rate to be 1 in accordance with all benchmark models in Jiang's work.

For each biased label primarily discussed in Obermeyer's paper, we train a vanilla logistic regression model without any constraints and a re-weighted model with the optimal λ that algorithm 1 has chosen for us. Then we collect the test error rates and the fairness violations for both model to see whether this method works well on our target dataset.

Furthermore, we want to re-confirm that the λ determined by our algorithm indeed builds up the optimal weight with which our model could reach a fairly good balance between prediction accuracy and bias mitigation. So there will also be a supplemental rounds of experiments where we feed a range of numbers around the chosen * into the weight calculation, that is, using $a \cdot \lambda^*$ where a take its range in [-1, 3] on a 0.01 grid.

3.4 Results

The result of the test error rate and the fairness violation are presented in table 12. At the 97-percentile threshold, we train three groups of vanilla model and the re-weighted model respectively with three labels. For those at 55-quantile threshold, only two labels – log.cost.total.t and gagne.sum.t – are available to be binarized. We find it unrealistic to implement the binarization on the label log.cost.avoidable.t at its 55 quantile, because there are over 55 percentile(35886 out of 48784) of our observations that have value 0 on the log.cost.avoidable.t column, which is also the smallest value of this column.

With a 97% label cutoff, we find that models trained with label log.cost.total.t and label log.cost.avoidable.t shows a small decrease on its fairness violation compared with the results of their unconstrained counterparts. It is also surprising at first to find that the test error rate of all three re-weighted models are decreased as well instead of having a tradeoff between the test error rate and the fairness violation. We further confirm that the training error rates of all models with 97% binary cutoff are also surprisingly lower than the unconstrained results. It could be the case where our vanilla regression model is not optimized enough(even though the overall test error rates are all absolutely low) because of this extreme binary cutoff, which we will discuss later concerning the validity of models under the 97% binary cutoff.

TABLE 1: Experiment Results: at 97th percentile cutoff

	no constraints		re-weighting.algo.	
	Te.Er.	*Fair.Vio.*	*Te.Er.*	*Fair.Vio*
log.cost.total.t	0.0361	0.0100	0.0349	0.0073
gagne.sum.t	0.0333	0.0162	0.0315	0.0160
log.cost.avoid.t	0.0337	0.0088	0.0331	0.0074

As for the models with 55% binary cutoff, we still cannot observe a bias-accuracy tradeoff on the test set(Table 2). However, we do observe an increase in the training error rate and an decrease for fairness violation when we add the fairness constraints into the training process for both labels (Table 3). The theoretical tradeoff happens on the training set but not on the test set, which is also explainable because the model we train with no constraints might not be able to capture all the features in the test set, while the re-weighted model might accidentally perform better on this specific test set.

Back to the prediction statistics on the test set, we are delighted to see a drastic percentage drop of fairness violation with both labels: 87.5% decrease with label log.cost.total.t and 83.8% decrease with label gagne.sum.t. At the same time, we even witness a slight decrease of test error rate with label log.cost.total.t, but still an obvious increase of test error rate with gagne.sum.t.

TABLE 2: Experiment Results: at 55th percentile cutoff

	no constraints		re-weighting.algo.	
	Te.Er.	*Fair.Vio.*	*Te.Er.*	*Fair.Vio*
log.cost.total.t	0.3380	0.0440	0.3294	0.0091
gagne.sum.t	0.2181	0.1036	0.3803	0.0211

TABLE 3: Experiment Results: at 55th percentile cutoff

	no constraints		re-weighting.algo.	
	Tr.Er.	*Fair.Vio.*	*Tr.Er.*	*Fair.Vio*
log.cost.total.t	0.3365	0.0280	0.3406	0.0035
gagne.sum.t	0.2221	0.0858	0.3798	0.0139

Another concern is about the threshold level itself. A 97-quantile cutoff is unavoidably leading to a very imbalanced dataset. Even if the overall prediction accuracy on the test set can be pretty impressive, most of the time we would get an unfair prediction towards to minority group. In our case, the only top 3% observations are assigned with $y = 1$, and the remaining 97% observations have been assigned with $y = 0$.

The false negative rate with binarizing cutoff at 97% cutoff, as we have expected, is extremely high as we can see from table 4. No matter with which label we train the model and whether we add fairness constraints or not, the false negative rates are all above 0.8. This means models trained under 97% cutoff all have very poor performance in classifying the original top 3% patients into the positive group. We care so much about the false negative rate because our model is intended to predict which group of people should be enrolled into the healthcare program based on their healthy risk. If our model gives very high false negative rate, those with extremely high health risk are very likely to be misclassified into the negative group, thus the purpose of this prediction system in finding the most demanding patients is unlikely to achieve.

Looking at Table 5 where we train models with 55% binary cutoff, all false negative rate is considerably lower compared to their counterparts with 97% cutoff. When we are using log.cost.total.t as the binary label, there is no big difference between the vanilla model and our re-weighted model regarding to these two rates. However, with gagne.sum.t as the label, we can see a tremendous decrease of FNR after implementing the weighting algorithm upon the model training, meanwhile an increase of FPR that cannot be ignored also arises in the re-weighted model. This dramatically increased error contributes to the overall poorer prediction accuracy when we adopt gagne.sum.t as our label, as we show in Table 2.

TABLE 4: false positive rate and false negative rate at 97% cutoff

	log.cost.total.t		gagne.sum.t	
	FPR	FNR	FPR	FNR
no constraints	0.0105	0.8703	0.0093	0.8319
re-weighting	0.0091	0.8787	0.0079	0.8170

TABLE 5: false positive rate and false negative rate at 55% cutoff

	log.cost.total.t		gagne.sum.t	
	FPR	FNR	FPR	FNR
no constraints	0.2730	0.4146	0.1349	0.5741
re-weighting	0.24581	0.4276	0.5741	0.0192

3.5 Supplementary Experiments

This part shows how well our model with the chosen coefficient λ^* has performed when we shift the coefficient in a certain range and impose them respectively on the model training process. The binary threshold is fixed at 55%. Experiment results are plotted as follows.

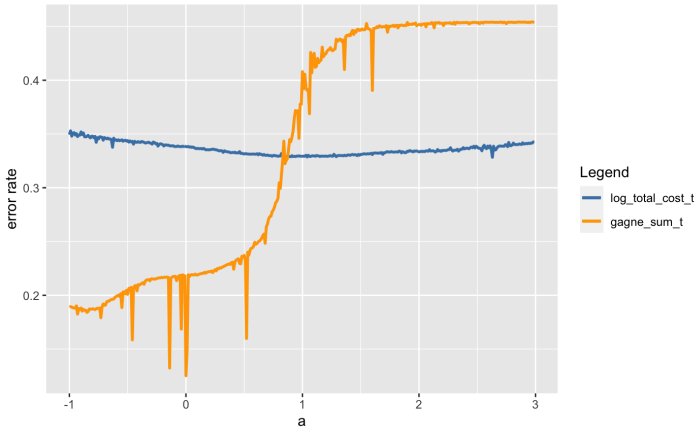

FIGURE 3: Test error rate of model trained with different weights

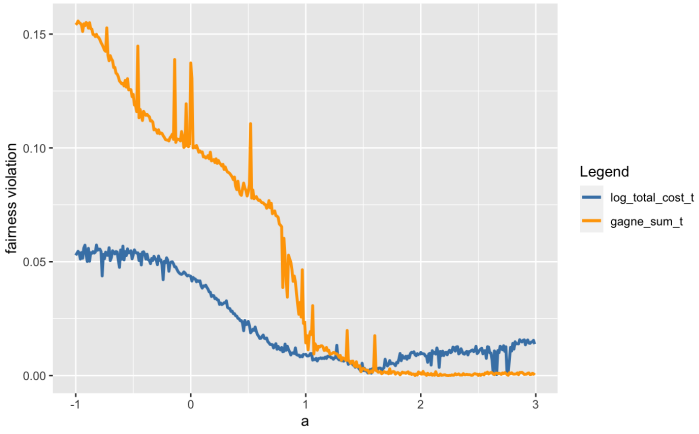

FIGURE 4: fairness violation of model trained with different weights

Figure 3 shows how the test error rate changes when we train the model with coefficient $\lambda = a \cdot \lambda^*$ where a is between -1 and 3. It is expected that when $a = 0$. the coefficient $\lambda = 0$, thus there's no re-weighting upon the training process, and we will end up with the vanilla regression model. The test error rate is expected to be the lowest when a is very close to 0. As we can see from Figure 3, the test error rate of models trained with label gagne.sum.t reaches its lowest

point at $a = 0$. And as we increase the absolute value of coefficients, namely, moving to the left or right of $a = 0$ along the x-axis, the test error rate becomes higher in varying degrees. On the contrast, the test error rate of models with label log.cost.total.t hardly manifests such trend. Overall, the test error rate does not vary too much while the weight upon the data distribution changes.

Figure 4 shows how the fairness violation changes as the weighting coefficients change. It is also expected that when $a = 1$, namely when the λ used to calculate the weight is indeed the optimal λ picked by Algorithm 1, the fairness violation could be mitigated. However, the result shows that models with either labels have the lowest value of fairness violation when a is around 1.5. Nevertheless, the fairness violation drops sharply when the λ approaches the optimal λ^* from the left hand side of the x-axis. And similar to the pattern of test error rate, the curves using label log.cost.total.t is flatter than curves using label gagne.sum.t.

From the perspective of bias-accuracy trade-off, we see an obvious complementary trends of test error rate and fairness violation when we adopt gagne.sum.t as the label. In contrast to this, we cannot observe an inverse trends between the curve of error rate and fairness violation when log.cost.total.t is adopted as the label.

4 Label Choice Under This Method

In [Obermeyer et al., 2019], the authors intend to find the best label with which the model reduces the racial bias the most. In [Jiang & Nachum, 2020], the authors try to prove a generic method in balancing the model accuracy and bias mitigation by showing that the implementation of their algorithm on several benchmark Machine Learning datasets outperforms other existing approaches in fixing the algorithmic bias problem. We replicate Jiang and Nachum's methods on the specific dataset in Obermeyer's paper, and different labels results in completely different effects, which serves as important evidences in choosing the appropriate label.

From the previous session, we see that models with the variable representing the total medical cost as the binary label is quite robust to fairness constraints in Jiang and Nachum's method. On the other hand, models with the variable representing the chronic disease condition show greater differences in the level of both prediction accuracy and fairness violation. Though we have not figure out underlying reasons of such a difference, at least we are able to conclude that the choice of gagne.sum.t as the training label leads to a more ideal performance with regard to the theoretical basis of Jiang's method. it also in accordance with the experiment results of the benchmark dataset in [Jiang & Nachum, 2020].

From a more practical perspective, we do think the false negative rate and false positive rate are as critical as the overall test error rate, if not more important. The false negative rate determines whether the most demanding patients will get enough attention from the healthcare system. The false positive rate also affects the economy of this prediction system, as we try to avoid too many patients being classified as people with health emergency when they are not actually in such urgent condition. It is always a hard task in finding the optimal balance between FNR and FPR, but at least in a health risk prediction system, it is our common sense that we would rather overestimate the risk than underestimate it and overlook those with real high health risk. As we have shown earlier, the re-weighted model with gagne.sum.t as the training label gives a drastic decrease in its FNR, whereas as a complementary effect the FPR also increases. The re-weighted model with log.cost.total.t as the label are, again, fairly robust in the aspect of FPR and FNR.

5 Conclusion

In general, we say that the label gagne.sum.t is so far the optimal label choice when implementing the weighting method upon our dataset, not only because of the consistency between our experiment results and the theoretical expectation, but also due to very low false negative rate of the re-weighted model. To some extent, we reach a fairness-accuracy win-win situation, where the racial bias is reduced with a reasonable trade-off in the overall test error rate, and the false negative rate that tops our priority list is almost mitigated to 0.

6 Future Developments

So far, the work we have summarized in this report is still a tip of the iceberg in the process bias mitigation for this specific dataset. The exisiting experiment results present some inconsistency with our expectations especially with the label log.cost.total.t. In the following researches, we would like to uncover the cause of such label-related differences in implementing this method.

More importantly, we need more comparisons between different approaches other than Jiang's method to see how models with different labels perform, therefore a comprehensive evaluation of each label choice can be consummated, and a more convincing validation concerning the advantage of Jiang's method over other approaches can be proposed.

References

[Jiang & Nachum, 2020] Jiang, H. & Nachum, O. (2020). Identifying and correcting label bias in machine learning. In International Conference on Artificial Intelligence and Statistics (pp. 702–712).: PMLR.

[Obermeyer et al., 2019] Obermeyer, Z., Powers, B., Vogeli, C., & Mullainathan, S. (2019). Dissecting racial bias in an algorithm used to manage the health of populations. Science, 366(6464), 447–453.