Group Fairness by Probabilistic Modeling with Latent Fair Decisions

YooJung Choi, Meihua Dang, Guy Van den Broeck
Computer Science Department
University of California, Los Angeles
{yjchoi,mhdang,guyvdb}@cs.ucla.edu

Abstract
Machine learning systems are increasingly being used to make impactful decisions such as loan applications and criminal justice risk assessments, and as such, ensuring fairness of these systems is critical. This is often challenging as the labels in the data are biased. This paper studies learning fair probability distributions from biased data by explicitly modeling a latent variable that represents a hidden, unbiased label. In particular, we aim to achieve demographic parity by enforcing certain independencies in the learned model. We also show that group fairness guarantees are meaningful only if the distribution used to provide those guarantees indeed captures the real-world data. In order to closely model the data distribution, we employ probabilistic circuits, an expressive and tractable probabilistic model, and propose an algorithm to learn them from incomplete data. We show on real-world datasets that our approach not only is a better model of how the data was generated than existing methods but also achieves competitive accuracy. Moreover, we also evaluate our approach on a synthetic dataset in which observed labels indeed come from fair labels but with added bias, and demonstrate that the fair labels are successfully retrieved.

1 Introduction
As machine learning algorithms are being used more and more in real-world decision making scenarios, there is growing concern that these methods may produce decisions that discriminate against particular groups of people. The relevant applications include online advertising, hiring, loan approvals, and criminal risk assessment (Datta, Tschantz, and Datta 2015; Barocas and Selbst 2016; Chouldechova 2017; Berk et al. 2018). To address these concerns, various methods have been proposed to quantify and ensure fairness in automated decision making systems (Chouldechova 2017; Dwork et al. 2012; Feldman et al. 2015; Kusner et al. 2017; Kamishima et al. 2012; Zemel et al. 2013). A widely used notion of fairness is demographic parity, which states that sensitive attributes such as gender or race must be statistically independent of the class predictions.

In this paper, we study the problem of enforcing demographic parity in probabilistic classifiers. In particular, we focus on the fact that class labels in the data are often biased, and propose a latent variable approach that treats the observed labels as biased proxies of hidden, fair labels that satisfy demographic parity. The process that generated bias is modeled by a probability distribution over the fair label, observed label, and other features including the sensitive attributes. Moreover, we show that group fairness guarantees for a probabilistic model hold in the real world only if the model accurately captures the real-world data. Therefore, the goal of learning a fair probabilistic classifier also entails learning a distribution that achieves high likelihood.

Our first contribution is to systematically derive the assumptions of a fair probabilistic model in terms of independence constraints. Each constraint serves the purpose of explaining how the observed, biased labels come from hidden fair labels and/or ensuring that the model closely represents the data distribution. Secondly, we propose an algorithm to learn probabilistic circuits (PCs) (Vergari, Di Mauro, and Van den Broeck 2019), a type of tractable probabilistic models, so that the fairness constraints are satisfied. Specifically, this involves encoding independence assumptions into the circuits and developing an algorithm to learn PCs from incomplete data, as we have a latent variable. Finally, we evaluate our approach empirically on synthetic and real-world datasets, comparing against existing fair learning methods as well as a baseline we propose that does not include a latent variable. The experiments demonstrate that our method achieves high likelihoods that indeed translate to more trustworthy fairness guarantees. It also has high accuracy for predicting the true fair labels in the synthetic data, and the predicted fair decisions can still be close to unfair labels in real-world data.

2 Related Work
Several frameworks have been proposed to design fairness-aware systems. We discuss a few of them here and refer to Romei and Ruggieri (2014); Barocas, Hardt, and Narayanan (2019) for a more comprehensive review.

Some of the most prominent fairness frameworks include individual fairness and group fairness. Individual fairness (Dwork et al. 2012) is based on the idea that similar individuals should receive similar treatments, although defining similarity between individuals can be challenging. On the other hand, group fairness aims to equalize some statistics across groups defined by sensitive attributes. These include equality of opportunity (Hardt, Price, and Srebro 2016) and demographic (statistical) parity (Calders and Verwer 2010;
Kamiran and Calders 2009) as well as its relaxed notion of disparate impact (Feldman et al. 2015; Zafar et al. 2017).

There are several approaches to achieve group fairness, which can be broadly categorized into (1) pre-processing data to remove bias (Zemel et al. 2013; Kamiran and Calders 2009; Calmon et al. 2017), (2) post-processing of model outputs such as calibration and threshold selection (Hardt, Price, and Srebro 2016; Pleiss et al. 2017), and (3) in-processing which incorporates fairness constraints directly in learning or optimization (Corbett-Davies et al. 2017; Agarwal et al. 2018; Kearns et al. 2018). Some recent works on group fairness also consider bias in the observed labels, both for evaluation and learning (Fogliatto, G’Sell, and Chouldechova 2020; Blum and Stangl 2020; Jiang and Nachum 2020). For instance, Blum and Stangl (2020) study empirical risk minimization (ERM) with various group fairness constraints and showed that ERM constrained by demographic parity does not recover the Bayes optimal classifier under one-sided, single-group label noise (this setting is subsumed by ours). In addition, Jiang and Nachum (2020) developed a pre-processing method to learn fair classifiers under noisy labels, by reweighting according to an unknown, fair labeling function. Here, the observed labels are assumed to come from a biased labeling function that is the “closest” to the fair one; on the other hand, we aim to find the bias mechanism that best explains the observed data.

We would like to point out that while pre-processing methods have the advantage of allowing any model to be learned on top of the processed data, it is also known that certain modeling assumptions can result in bias even when learning from fair data (Choi et al. 2020). Moreover, certain post-processing methods to achieve group fairness are shown to be suboptimal under some conditions (Woodworth et al. 2017). Instead, we take the in-processing approach to explicitly optimize the model’s performance while enforcing fairness.

Many fair learning methods make use of probabilistic models such as Bayesian networks (Calders and Verwer 2010; Mancuhan and Clifton 2014). Among those, perhaps the most related to our approach is the latent variable naive Bayes model by Calders and Verwer (2010), which also assumes a latent decision variable to make fair predictions. However, they make a naive Bayes assumption among features. We relax this assumption and will later demonstrate how this helps in more closely modeling the data distribution, as well as providing better fairness guarantees.

### 3 Latent Fair Decisions

We use uppercase letters (e.g., $X$) for discrete random variables (RVs) and lowercase letters ($x$) for their assignments. Negation of a binary assignment $x$ is denoted by $\overline{x}$. Sets of RVs are denoted by bold uppercase letters ($\mathbf{X}$), and their joint assignments by bold lowercase ($\mathbf{x}$). Let $S$ denote a sensitive attribute, such as gender or race, and let $\mathbf{X}$ be the non-sensitive attributes or features. In this paper, we assume $S$ is a binary variable for simplicity, but our method can be easily generalized to multiple multi-valued sensitive attributes. We have a dataset $\mathcal{D}$ in which each individual is characterized by variables $S$ and $\mathbf{X}$ and labeled with a binary decision/class variable $D$.

One of the most popular and yet simple fairness notions is demographic (or statistical) parity. It requires that the classification is independent of the sensitive attributes; i.e., the rate of positive classification is the same across groups defined by the sensitive attributes. Since we focus on probabilistic classifiers, we consider a generalized version introduced by Pleiss et al. (2017), sometimes also called strong demographic parity (Jiang et al. 2019):

**Definition 1** (Generalized demographic parity). Suppose $f$ is a probabilistic classifier and $p$ is a distribution over variables $\mathbf{X}$ and $S$. Then $f$ satisfies demographic parity w.r.t. $p$ if:

$$E_p[f(\mathbf{X}, S) \mid S = 1] = E_p[f(\mathbf{X}, S) \mid S = 0].$$

Probabilistic classifiers are often obtained from joint distributions $P(\cdot)$ over $\mathbf{D}, \mathbf{X}, S$ by computing $P(D \mid \mathbf{X}, S)$. Then we say the distribution satisfies demographic parity if $P(D = 1 \mid S = 1) = P(D = 1 \mid S = 0)$, i.e., $D$ is independent of $S$.

### 3.1 Motivation

A common fairness concern when learning decision making systems is that the dataset used is often biased. In particular, observed labels may not be the true target variable but only its proxy. For example, re-arrest is generally used as a label for recidivism prediction, but it is not equivalent to recidivism and may be biased. We will later show how the relationship between observed label and true target can be modeled probabilistically using a latent variable.

Moreover, probabilistic group fairness guarantees hold in the real world only if the model accurately captures the real world distribution. In other words, using a model that only achieves low likelihood w.r.t. the data, it is easy to give false guarantees. For instance, consider a probabilistic classifier $f(\mathbf{X}, S)$ over a binary sensitive attribute $S$ and non-sensitive attribute $\mathbf{X}$ shown below:

<table>
<thead>
<tr>
<th>$S, \mathbf{X}$</th>
<th>$f(\mathbf{X}, S)$</th>
<th>$P_{true}(\mathbf{X} \mid S)$</th>
<th>$E_{P_{true}}[f(\mid S)]$</th>
<th>$Q(\mathbf{X} \mid S)$</th>
<th>$E_Q[f(\mid S)]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>0.8</td>
<td>0.7</td>
<td>0.65</td>
<td>0.5</td>
<td>0.55</td>
</tr>
<tr>
<td>1.0</td>
<td>0.3</td>
<td>0.3</td>
<td>0.5</td>
<td>0.5</td>
<td>0.55</td>
</tr>
<tr>
<td>0.1</td>
<td>0.7</td>
<td>0.4</td>
<td>0.5</td>
<td>0.5</td>
<td>0.55</td>
</tr>
<tr>
<td>0.0</td>
<td>0.4</td>
<td>0.6</td>
<td>0.5</td>
<td>0.5</td>
<td>0.55</td>
</tr>
</tbody>
</table>

Suppose in the data, the probability of $X = 1$ given $S = 1$ (resp. $S = 0$) is 0.7 (resp. 0.4). Then this classifier does not satisfy demographic parity, as the expected prediction for group $S = 1$ is $0.8 \cdot 0.7 + 0.3 \cdot 0.3 = 0.65$ while for group $S = 0$ it is 0.52. On the other hand, suppose you have a distribution $Q$ that incorrectly assumes the feature $X$ to be uniform and independent of $S$. Then you would conclude, incorrectly, that the prediction is indeed fair, with the average prediction for both protected groups being 0.55. Therefore, to provide meaningful fairness guarantees, we need to model the data distribution closely, i.e., with high likelihood.

### 3.2 Modeling with a Latent Fair Decision

We now describe our proposed latent variable approach to address the aforementioned issues. We suppose there is a hidden variable that represents the true label without discrimination. This latent variable is denoted as $D_f$ and used for prediction instead of $D$; i.e., decisions for future instances can be made by inferring the conditional probability $P(D_f \mid e)$ given some
Figure 1: Bayesian network structures that represent the proposed fair latent variable approach (left) and model without a latent variable (right). Abusing notation, the set of features \( X \) is represented as a single node, but refers to some local Bayesian network over \( X \).

There are several challenges in modeling a fair probability distribution. First, as shown previously, fairness guarantees hold with respect to the modeled distribution, and thus we want to closely model the data distribution. A possible approach is to learn a deep generative model such as a generative adversarial network (GANs) (Goodfellow et al. 2014). However, then we must resort to approximate inference, or deal with models that have no explicit likelihood, and the fairness guarantees no longer hold. An alternative is to use models that allow exact inference such as Bayesian networks. Unfortunately, marginal inference, which is needed to make predictions \( \Pr(D \mid e) \), is \#P-hard for general BNs (Roth 1996). Tree-like BNs such as naive Bayes allow polytime inference, but they are not expressive enough to accurately capture the real world distribution. Hence, the second challenge is to also support tractable exact inference without sacrificing expressiveness.

Lastly, the probabilistic modeling method we choose must be able to encode the independencies outlined in the previous section, to satisfy demographic parity and to learn a meaningful relationship between the latent fair decision and other variables. In the following, we give some background on probabilistic circuits (PCs) and show how they satisfy each of the above criteria. Then we will describe our proposed algorithm to learn fair probabilistic circuits from data.
4.1 Probabilistic Circuits

**Representation** Probabilistic circuits (PCs) (Vergari et al. 2020; Choi, Vergari, and Van den Broeck 2020) refer to a family of tractable probabilistic models including arithmetic circuits (Darwiche 2002, 2003), sum-product networks (Poon and Domingos 2011), cutset networks (Rahman, Kothalkar, and Gogate 2014), and and-or search spaces (Marinescu and Dechter 2005). A probabilistic circuit \( C = (G, \theta) \) over RVs \( X \) is characterized by its structure \( G \) and parameters \( \theta \). The circuit structure \( G \) is a directed acyclic graph (DAG) such that each inner node is either a sum node or a product node, and each leaf (input) node is associated with a univariate input distribution. We denote the distribution associated with leaf \( n \) by \( f_n(x) \). This may be any probability mass function, a special case being an indicator function such as \( [X = 1] \). Parameters \( \theta \) are each associated with an input edge to a sum node. Note that a subcircuit rooted at an inner node of a PC is itself a valid PC. Figure 2 depicts an example probabilistic circuit.\(^1\)

Let \( \text{ch}(n) \) be the set of children nodes of an inner node \( n \). Then a probabilistic circuit \( C \) over RVs \( X \) defines a joint distribution \( \text{Pr}_C(X) \) in a recursive way as follows:

\[
\text{Pr}_n(x) = \begin{cases} 
  f_n(x) & \text{if } n \text{ is a leaf node} \\
  \prod_{c \in \text{ch}(n)} \text{Pr}_c(x) & \text{if } n \text{ is a product node} \\
  \sum_{c \in \text{ch}(n)} \theta_{n,c} \text{Pr}_c(x) & \text{if } n \text{ is a sum node}
\end{cases}
\]

Intuitively, a product node \( n \) defines a factorized distribution, and a sum node \( n \) defines a mixture model parameterized by weights \( \{\theta_{n,c}\}_{c \in \text{ch}(n)} \). \( \text{Pr}_n \) is also called the output of \( n \).

**Properties and inference** A strength of probabilistic circuits is that (1) they are expressive, achieving high likelihoods on density estimation tasks (Rahman and Gogate 2016; Liang, Bekker, and Van den Broeck 2017; Peharz et al. 2020), and (2) they support tractable probabilistic inference, enabled by certain structural properties. In particular, PCs support efficient marginal inference if they are smooth and decomposable. A circuit is said to be smooth if for every sum node all of its children depend on the same set of variables; it is decomposable if for every product node its children depend on disjoint sets of variables (Darwiche and Marquis 2002). Given a smooth and decomposable probabilistic circuit, computing the marginal probability for any partial evidence is reduced to simply evaluating the circuit bottom-up. This also implies tractable computation of conditional probabilities, which are ratios of marginals. Thus, we can make predictions in time linear in the size of the circuit.

Another useful structural property is determinism; a circuit is deterministic if for every complete input \( x \), at most one child of every sum node has a non-zero output. In addition to enabling tractable inference for more queries (Choi and Darwiche 2017), it leads to closed-form parameter estimation of probabilistic circuits given complete data. We also exploit this property for learning PCs with latent variables, which we will later describe in detail.

**Encoding independence assumptions** Next, we demonstrate how we encode the independence assumptions of a fair distribution as in Figure 1a in a probabilistic circuit. Recall the example PC in Figure 2: regardless of parameterization, this circuit structure always encodes a distribution where \( D \) is independent of \( X \) given \( S \) and \( D_f \). To prove this, we first observe that the four product nodes in the second layer each correspond to four possible assignments to \( S \) and \( D_f \). For instance, the left-most product node returns a non-zero output only if the input sets both \( S = 1 \) and \( D_f = 1 \). Effectively, the sub-circuits rooted at these nodes represent conditional distributions \( \text{Pr}(D, X|s, d_f) \) for assignments \( s, d_f \). Because the distributions for \( D \) and \( X \) factorize, we have \( \text{Pr}(D, X|s, d_f) = \text{Pr}(D|s, d_f) \cdot \text{Pr}(X|s, d_f) \), thereby satisfying the conditional independence \( D \perp X|D_f, S \).

We also need to encode the independence between \( D_f \) and \( S \). In the example circuit, each edge parameter \( \theta_i \) corresponds to \( \text{Pr}(s, d_f) \) for a joint assignment to \( S, D_f \); e.g. \( \theta_1 = \text{Pr}(S = 1, D_f = 1) \). With no restriction on these parameters, the circuit structure does not necessarily imply \( D_f \perp S \). Thus, we introduce auxiliary parameters \( \phi_s \) and \( \phi_{d_f} \), representing \( \text{Pr}(S = 1) \) and \( \text{Pr}(D_f = 1) \), respectively, and enforce that the circuit parameters for \( \text{Pr}(S, D_f) \) factorize as follows:

\[
\phi_s = \theta_1 + \theta_2, \quad \phi_{d_f} = \theta_3 + \theta_4, \\
\theta_1 = \phi_s \cdot \phi_{d_f}, \quad \theta_2 = \phi_s \cdot (1 - \phi_{d_f}), \\
\theta_3 = (1 - \phi_s) \cdot \phi_{d_f}, \quad \theta_4 = (1 - \phi_s) \cdot (1 - \phi_{d_f}).
\]

Hence, when learning these parameters, we limit the degree of freedom such that the four edge parameters are given by two free variables \( \phi_s \) and \( \phi_{d_f} \) instead of the four \( \theta_i \) variables.

Next, we discuss how to learn a fair probabilistic circuit with latent variable from data. This consists of two parts: learning the circuit structure and estimating the parameters of a given structure. We first study parameter learning in the next section, then structure learning in Section 4.3.

**4.2 Parameter Learning**

Given a complete data set, maximum-likelihood parameters of a smooth, decomposable, and deterministic PC can be computed in closed-form (Kisa et al. 2014). For an edge between a sum node \( n \) and its child \( c \), the associated maximum-
likelihood parameter for a complete dataset $D$ is given by:

$$
\theta_{n,c} = \frac{F_D(n,c)}{\sum_{c \in \text{ch}(n)} F_D(n,c)} \quad (1)
$$

Here, $F_D(n,c)$ is called the circuit flow of edge $(n,c)$ given $D$, and it counts the number of data samples in $D$ that “activate” this edge. For example, in Figure 2, the edges activated by sample $\{D_f = 1, S = 1, d, x\}$, for any assignments $d, x$, are colored red.\(^2\)

However, our proposed approach for fair distribution includes a latent variable, and thus must be learned from incomplete data. One of the most common methods to learn parameters of a probabilistic model from incomplete data is the Expectation Maximization (EM) algorithm (Koller and Friedman 2009; Darwiche 2009). EM iteratively completes the data by computing the probability of unobserved values (E-step) and estimates the maximum-likelihood parameters from the expected dataset (M-step).

We now propose an EM parameter learning algorithm for PCs that does not explicitly complete the data, but rather utilizes circuit flows. In particular, we introduce the notion of expected flows, which is defined as the following for a given circuit $C = (G, \theta)$ over RVs $Z$ and an incomplete dataset $D$:

$$
EF_{D,\theta}(n, c) := \mathbb{E}_{Pr_c}[F_{D,\theta}(n, c)] = \sum_{D_i \in D} \sum_{x \in \text{set}[D_i]} Pr_c(z|D_i) \cdot F_{\theta}(n, c).
$$

Here, $D_i$ denotes the i-th sample in the dataset, and $z \models D_i$ are the possible completions of sample $D_i$. For example, in Figure 2, the expected flows of the edges highlighted in red and green, given a sample $\{S = 1, d, x\}$, are $Pr_c(D_f = 1 | S = 1, d, x)$ and $Pr_c(D_f = 0 | S = 1, d, x)$, respectively. Similar to circuit flows, the expected flows for all edges can be computed with a single bottom-up and top-down evaluation of the circuit. Then, we can perform both the E- and M-step by the following closed-form solution.

**Proposition 1.** Given a smooth, decomposable, and deterministic circuit with parameters $\theta$ and an incomplete data $D$, the parameters for the next EM iteration are given by:

$$
\theta_{n,c}^{(\text{new})} = EF_{D,\theta}(n, c)/\sum_{c \in \text{ch}(n)} EF_{D,\theta}(n, c).
$$

Note that this is very similar to the ML estimate from complete data in Eq.1, except using expected flows instead of circuit flows. Moreover, the expected flow can be computed even if each data sample has different variables missing; thus, the EM method can naturally handle missing values for other features as well. We refer to Appendix A for details on computing the expected flows and proof for above proposition.

**Initial parameters using prior knowledge** Typically the EM algorithm is run starting from randomly initialized parameters. While the algorithm is guaranteed to improve the likelihood at each iteration until convergence, it still has the problem of multiple local maxima and identifiability, especially when there is a latent variable involved (Koller and Friedman 2009). Namely, we can converge to different learned models with similar likelihoods but different parameters for the latent fair variable, thus resulting in different behaviors in the prediction task. For example, for a given fair distribution, we can flip the value of $D_f$ and the parameters accordingly such that the marginal distribution over $S, X, D$, as well as the likelihood on the dataset, is unchanged. However, this clearly has a significant impact on the predictions which will be completely opposite.

Therefore, instead of random initialization, we encode prior knowledge in the initial parameters that determine $Pr(D|D_f)$. In particular, it is obvious that $D_f$ should be equal to $D$ if the observed labels are already fair. Furthermore, for individual predictions, we would want $D_f$ to be close to $D$ as much as possible while ensuring fairness. Thus, we start the EM algorithm from a conditional probability $Pr(d|s, d_f) = [d = d_f]$.

### 4.3 Structure Learning

Lastly, we describe how a fair probabilistic circuit structure is learned from data. As described previously, top layers of the circuit are fixed in order to encode the independence assumptions of our latent variable approach. On the other hand, the sub-circuits over features $X$ can be learned to best fit the data. We adopt the STRUDEL algorithm to learn the structures (Dang, Vergari, and Van den Broeck 2020).\(^3\) Starting from a Chow-Liu tree initial distribution (Chow and Liu 1968), STRUDEL performs a heuristic-based greedy search over possible candidate structures. At each iteration, it first selects the edge with the highest circuit flow and the variable with the strongest dependencies on other variables, estimated by the sum of pairwise mutual information. Then it applies the split operation – a simple structural transformation that “splits” the selected edge by introducing new sub-circuits conditioned on the selected variable. Intuitively, this operation aims to model the data more closely by capturing the dependence among variables (variable heuristic) appearing in many data samples (edge heuristic). After learning the structure, we update the parameters of the learned circuit using EM as described previously.

### 5 Experiments

We now empirically evaluate our proposed model FAIRPC on real-world benchmark datasets as well as synthetic data.

**Baselines** We first compare FAIRPC to three other probabilistic methods: fair naive Bayes models (2NB and LfNB) by Calders and Verwer (2010) and PCs without latent variable (NLATPC) as described in Section 3. We also compare against existing methods that learn discriminative classifiers satisfying group fairness: (1) FAIRL (Zafar et al. 2017), which learns a classifier subject to co-variance constraints;
In terms of log-likelihoods, both PC-based methods outperform NB models, which aligns with our motivation for relaxing the naive Bayes assumption—to better fit the data distribution. Furthermore, models with latent variables outperform their corresponding non-latent models, i.e., LATNB outperforms 2NB and FAIRPC outperforms NLATPC. This validates our argument made in Section 3 that the latent variable approach can achieve higher likelihood than enforcing fairness directly in the observed label. Next, we compare the methods using F1-score as there is class imbalance in these datasets. Although it is measured with respect to possibly biased labels, FAIRPC achieves competitive performance, demonstrating that the latent fair decision variable still exhibits high similarity with the observed labels. Lastly, FAIRPC achieves the lowest discrimination scores in COMPAS and Adult datasets by a significant margin. As expected, PCs also achieve lower discrimination scores than their counterpart NB models, as they fit the data distribution better.

**Discriminative classifiers** Next we compare FAIRPC to existing fair classification methods. Figure 3a shows the trade-off between predictive performance and fairness. We add two other baselines to the plot: RAND, which makes random predictions, and LR, which is an unconstrained logistic regression classifier. They represent the two ends of the fairness-accuracy tradeoff. RAND has no predictive power but low discrimination, while LR has high accuracy but unfair. Informally, the further above the line between these baselines, the better the method optimizes this tradeoff.

On COMPAS and Adult datasets, our approach achieves a good balance between predictive performance and fairness guarantees. In fact, it achieves the best or close to best accuracy and F1-score, again showing that the latent decision variable is highly similar to the observed labels even though the explicit objective is not to predict the unfair labels. However, on German dataset, while FAIRLR and WEIGHT achieve the best performance on average, the estimates for all models including the trivial baselines are too noisy to draw a statistically significant conclusion. This may be explained by the fact that the dataset is relatively small with 1000 samples.

### 5.2 Synthetic Data
As discussed previously, ideally we want to evaluate against the true target labels, but they are generally unknown in real-world data. Therefore, we also evaluate on synthetic data with fair ground-truth labels in order to evaluate whether our model indeed captures the hidden process of bias and makes accurate predictions.

**Generating Data** We generate data by constructing a fair PC $C_{\text{true}}$ to represent the “true distribution” and sampling from it. The process that generates biased labels $d$ is represented by the following (conditional) probability table:

| $D_f$ | $S$ | $d_f,s$ | $P(D_f = 1 | D_f = d_f, S = s)$ |
|-------|-----|---------|---------------------------------|
|       |     |         | 1.1 1.0 0.1 0.0                  |
| $Pr(C = 1)$ | 0.5 | 0.3 | $Pr(D = 1 | D_f = d_f, S = s)$ | 0.8 0.9 0.1 0.4 |

Here, $S = 1$ is the minority group, and the unfair label $D$ is in favor of the majority group: $D$ is more likely to be positive for
the majority group $S = 0$ than for $S = 1$, for both values of fair label $D_f$ but at different rates. To evaluate on a wide range of datasets, we randomly generate the sub-circuits of $C_{true}$ over features $X$ as tree distributions, randomly initializing the parameters with Laplace smoothing. We generated different synthetic datasets with the number of non-sensitive features ranging from 10 to 30, using 10-fold CV for each.

**Results** We first test FAIRPC, LATNB, NLATPC and NLATPC on the generated datasets. Figure 3b (top) illustrates the accuracy and discrimination scores on separate test sets with fair decision labels.

In terms of accuracy, PCs outperform NBs, and latent variable approaches outperform non-latent ones, which shows that adopting density estimation to fit the data and introducing a latent variable indeed help improve the performance.

When comparing the average discrimination score for each method, 2NB and NLATPC always have negative scores, showing that the non-latent methods are more biased towards the majority group; while LATNB and FAIRPC are more equally distributed around zero on the x-axis, thus demonstrating that a latent fair decision variable helps to correct this bias. While both latent variable approaches achieve reasonably low discrimination on average, FAIRPC is more stable and has even lower average discrimination score than LATNB. Moreover FAIRPC also outperforms the other probabilistic methods in terms of likelihood; see Appendix B.

We also compare FAIRPC to FAIRR, REDUCTION, and REWEIGHT, the results visualized in Figure 3b (bottom). Our method achieves a much higher accuracy w.r.t. the generated fair labels; for instance, the average accuracy of FAIRPC is around 0.17 higher than that of FAIRR. Also, we are still being comparable in terms of discrimination score, illustrating the benefits of explicitly modeling the latent fair decision.

**5.3 Additional experiments**

Appendix B includes learning curves, statistical tests, and detailed performance of our real-world data experiments, as well as the following additional experiments. We empirically validated that initializing parameters using prior knowledge as described in Section 4.2 indeed converges closer to the true distribution of $P(r|S, D_f)$ than randomly initializing parameters. In addition, as mentioned in Section 4.2, our method can be applied even on datasets with missing values, with no change to the algorithm. We demonstrate this empirically and show that our approach still gets comparably good performance for density estimation.

**6 Conclusion**

In this paper, we proposed a latent variable approach to learning fair distributions that satisfy demographic parity, and developed an algorithm to learn fair probabilistic circuits from incomplete data. Experimental evaluation on simulated data showed that our method consistently achieves the highest log-likelihoods and a low discrimination score. It also accurately predicts true fair decisions, and even on real-world data where fair labels are not available, our predictions remain close to the unfair ones.

**Acknowledgments**

This work is supported in part by NSF grants #CCF-1837129, #IIS-1956441, #IIS-1943641, DARPA grant #N66001-17-2-4032, a Sloan Fellowship, and gifts from Intel and Facebook Research.
References


