Certifying Fairness of Probabilistic Circuits

Nikil Roashan Selvam[1] **Guy Van den Broeck**[1] **YooJung Choi**[1]

[1]University of California, Los Angeles

Abstract

With the increased use of machine learning systems for decision making, questions about their fairness properties start to take center stage. A recently introduced notion of fairness asks whether the model exhibits a *discrimination pattern*, in which an individual—characterized by (partial) feature observations—receives vastly different decisions merely by disclosing some sensitive attributes. Existing work on checking the presence of such patterns is limited to naive Bayes classifiers, which make strong independence assumptions. This paper proposes an algorithm to search for discrimination patterns in a more general class of probabilistic models—probabilistic circuits. If a model is not fair, it may be useful to quickly find discrimination patterns and distill them for better interpretability. As such, we also propose a sampling-based approach to more efficiently mine discrimination patterns, and introduce new classes of discrimination patterns: minimal, maximal, and Pareto optimal.

1 INTRODUCTION

Machine learning systems are increasingly being used for decision making in a variety of areas [5, 2, 10, 17]. Consequently, there has been growing efforts to address biases in the training data and model architecture leading to certain individuals or groups receiving unfavorable treatment based on some sensitive attributes such as gender and race [1, 14, 16, 23, 21, 27].

In this paper, we investigate the fairness properties of probabilistic models that represent joint distributions over the decision variable as well as the features. Such models are ubiquitous in decision-making systems for various real-world applications [19, 28, 15]. In particular, they can be used to make classifications by inferring the probability of the class given some observations, naturally handling missing features at prediction time. While many existing work on algorithmic fairness assume that predictions are always made with complete observations of features, the notion of discrimination pattern [4] explicitly aims to address fairness of decisions made with partial information. Specifically, it refers to individuals, characterized by some partial assignments to the features, who may see a significant discrepancy in the prediction after additionally disclosing some sensitive attributes. However, as a model could exhibit as many as an exponential number of discrimination patterns, using this notion to audit and analyze the model can be challenging.

We make two main contributions. The first is to introduce special classes of discrimination patterns—minimal, maximal, and Pareto optimal patterns—which can "summarize" a large number of patterns, making them great targets to find in a model in order to discover and understand its unfairness. The second contribution is algorithms to find discrimination patterns. While the existing algorithm is limited to naive Bayes models, our proposed methods can be applied to a more general class of models called probabilistic circuits (PCs) [29]. We propose a search-based algorithm that can find all discrimination patterns in a PC or otherwise certify that there exists none and thus the PC is fair. We also introduce a sampling-based method, which can no longer prove fairness, but can far more efficiently find many discrimination patterns if they exist. Empirical evaluation on three benchmark datasets demonstrates that our search algorithm is able to find all discrimination patterns while only traversing a fraction of the space of possible patterns. Furthermore, we show that the sampling-based approach is indeed significantly faster in pattern mining, while still returning similar summary patterns as the exact approach.

2 DISCRIMINATION PATTERNS

Random variables and their assignments are denoted by upper (X) and lowercase (x) letters, respectively, and their sets by respective bold letters ($\mathbf{X}, \mathbf{x}$). The set of possible values of $\mathbf{X}$ is denoted by $\mathsf{val}(\mathbf{X})$. D denotes a binary decision variable, and d its assignment that represents a favorable

Accepted for the 5th Workshop on Tractable Probabilistic Modeling at UAI (TPM 2022).

decision. A set of discrete variables (features) $\mathbf{Z}$ are used to make decisions, and a subset of variables $\mathbf{S} \subset \mathbf{Z}$ are designated as *sensitive attributes*, such as race or gender.

Definition 1 (Discrimination patterns [4])**.** Let P be a distribution over $D \cup \mathbf{Z}$, and $\mathbf{x}, \mathbf{y}$ be joint assignments to $\mathbf{X} \subseteq \mathbf{S}$, $\mathbf{Y} \subseteq \mathbf{Z} \setminus \mathbf{X}$, respectively. For some $\delta \in [0, 1]$, we say $(\mathbf{x}, \mathbf{y})$ form a *discrimination pattern* w.r.t. P and δ if:

$$|P(d \mid \mathbf{x}, \mathbf{y}) - P(d \mid \mathbf{y})| > \delta.$$

The LHS is referred to as its *discrimination score*, $\Delta(\mathbf{x}, \mathbf{y})$.

Intuitively, a discrimination pattern corresponds to individuals who would see a significant difference in the probability of getting a favorable decision just by disclosing some sensitive information. A model is said to be δ-fair iff it exhibits no discrimination pattern with respect to δ. On the other hand, if there exist discrimination patterns, it would be useful for domain experts or users to examine them to better understand the behavior of the classifier and improve its fairness properties. However, even a simple classifier could exhibit a large number of discrimination patterns. Thus, we next propose new classes of discrimination patterns that can be used as representatives for a large number of patterns, thereby being more amenable for interpretations.

A natural way to choose the most "interesting" patterns may be by ranking them by their discrimination scores, and focusing on a few instances that are the most discriminatory. While it may be useful to study the most problematic patterns and address them, they do not necessarily provide insight into other discrimination patterns that exist. Instead, we propose the notion of maximal and minimal patterns that can summarize groups of patterns, namely their extensions and contractions. An extension of a pattern $(\mathbf{x}, \mathbf{y})$, denoted by $(\mathbf{x}', \mathbf{y}') \supset (\mathbf{x}, \mathbf{y})$, can be generated by adding an assignment to the pattern: $\mathbf{x} \subseteq \mathbf{x}'$, $\mathbf{y} \subseteq \mathbf{y}'$, and either $\mathbf{x} \subset \mathbf{x}'$ or $\mathbf{y} \subset \mathbf{y}'$. Conversely, $(\mathbf{x}, \mathbf{y})$ is called a contraction of $(\mathbf{x}', \mathbf{y}')$.

Definition 2 (Maximal & minimal patterns)**.** Let Σ denote a set of discrimination patterns w.r.t. a distribution P and threshold δ. The set of *maximal patterns* $\Sigma_{\text{max}} \subseteq \Sigma$ consists of all patterns $(\mathbf{x}, \mathbf{y}) \in \Sigma$ that are not a complete assignment (i.e. $\mathbf{Z} \setminus (\mathbf{X} \cup \mathbf{Y}) \neq \emptyset$) and:

$$\forall (\mathbf{x}', \mathbf{y}') \supset (\mathbf{x}, \mathbf{y}),\ (\mathbf{x}', \mathbf{y}') \notin \Sigma.$$

The set of *minimal patterns* $\Sigma_{\text{min}} \subseteq \Sigma$ consists of all patterns $(\mathbf{x}, \mathbf{y}) \in \Sigma$ such that:

$$\begin{aligned} &\forall\, (\mathbf{x}', \mathbf{y}') \supset (\mathbf{x}, \mathbf{y}),\ (\mathbf{x}', \mathbf{y}') \in \Sigma \\ \text{and} \quad &\forall\, (\mathbf{x}'', \mathbf{y}'') \subset (\mathbf{x}, \mathbf{y}) \text{ s.t. } \mathbf{x}'' \neq \emptyset,\ (\mathbf{x}'', \mathbf{y}'') \notin \Sigma. \end{aligned}$$

In other words, a maximal pattern is a discrimination pattern such that none of its extensions are discrimination patterns. As the name suggests, an extension of a maximal pattern cannot also be maximal, because by definition it will not be a discrimination pattern. Hence, an individual with attributes $\mathbf{x}$ and $\mathbf{y}$ who may see a discrimination in the decision by disclosing their sensitive information would no longer receive such treatment if they additionally share other features, whatever their values may be. On the other hand, we say a discrimination pattern is minimal if all of its extensions and none of its contractions are discriminatory. Thus, a single minimal pattern effectively represents a large set of discrimination patterns, whose size is exponential in the number of unobserved features. Note that by definition a minimal pattern cannot be a maximal pattern, and vice versa.

As a case study, we train a probabilistic model[1] on the COMPAS dataset, which exhibits 7445, 2338, and 1164 discrimination patterns for $\delta = 0.01, 0.05$, and 0.1 respectively. On the other hand, this model has 170, 74, and 0 maximal patterns, and only 103, 10, and 1 minimal patterns, for respective values of threshold δ. Interestingly, none of the 74 maximal patterns for $\delta = 0.05$ includes an assignment to the variable regarding 'supervision level', suggesting that there are many instances where an individual would not see an unfair prediction if the supervision level is additionally known. Remarkably, the single minimal pattern in the case of $\delta = 0.1$ can represent 512 patterns (its extensions) out of 1164 total discrimination patterns in the model.

Another important consideration when studying discrimination patterns is their probability. Recall that each pattern $(\mathbf{x}, \mathbf{y})$ represents a group of people sharing the attributes $\mathbf{x}$ and $\mathbf{y}$. Then the probability $P(\mathbf{x}, \mathbf{y})$ corresponds to the proportion of the population that could be affected. Even though any unfairness is equally undesirable for minority groups (i.e. lower probability) as it is for majority groups, patterns that are so specific and have exceedingly low probability would not be as insightful when auditing a model for real-world fairness concerns. As such, we propose to summarize discrimination patterns by constructing a set of patterns such that one cannot increase the discrimination score without decreasing the probability, and vice versa.

Definition 3 (Pareto optimal patterns)**.** The set of *Pareto optimal patterns* $\Sigma_{\text{PO}} \subseteq \Sigma$ consists of the patterns $(\mathbf{x}, \mathbf{y}) \in \Sigma$ such that $\forall (\mathbf{x}', \mathbf{y}') \in \Sigma \setminus \{(\mathbf{x}, \mathbf{y})\}$,

$$\Delta(\mathbf{x}, \mathbf{y}) > \Delta(\mathbf{x}', \mathbf{y}') \text{ or } P(\mathbf{x}, \mathbf{y}) > P(\mathbf{x}', \mathbf{y}').$$

Pareto optimal patterns can be a very effective way to study fairness of a probabilistic model, as it significantly reduces the number of discrimination patterns one would examine. For example, the model trained on the COMPAS with 2388 and 1164 discrimination patterns w.r.t. $\delta = 0.05$ and 0.1, respectively, has only 38 and 28 Pareto optimal patterns. That is, for $\delta = 0.1$, each of the $1164 - 28 = 1136$ patterns has discrimination score and probability that are both dominated by those of some Pareto optimal pattern.

[1] See Sec. 4 for details of the trained model.

3 FINDING DISCRIMINATION PATTERNS IN A PC

As discussed in Sec. 2, the probability of a pattern corresponds to the proportion of the affected subpopulation, according to the probabilistic model. Therefore, a meaningful analysis of discrimination patterns depends on how well the model captures the population distribution. For instance, the existing algorithm to discover discrimination patterns assumes naive Bayes classifiers, which make strong independence assumptions and are generally too restrictive to fit real-world distributions. We instead consider a more expressive type of models called probabilistic circuits, which refer to a family of probabilistic models that support tractable inference, encompassing arithmetic circuits [8], sum-product networks [25], and-or graphs [11], probabilistic sentential decision diagrams [18], and more. They were shown to achieve competitive likelihoods in various density estimation tasks [7, 20, 24]. In addition, classical graphical models with bounded treewidth [6], including naive Bayes, and their mixtures [22] can also easily be represented as PCs.

PCs allow tractable inference of certain probabilistic queries, based on which structural properties they satisfy. The first inference task we need is conditional probabilities, to compute the discrimination score (Def. 1) of a pattern. Moreover, we would also like to compute probabilities of patterns—that is, marginal probabilities given some partial observations. PCs support efficient marginal and conditional inference if they satisfy two structural properties called smoothness and decomposability. A PC is *smooth* if for every sum node its children include exactly the same set of variables, and it is *decomposable* if for every product node its children depend on disjoint sets of variables [9]. Given these properties, computing any marginal probability can be done through a single feedforward evaluation of the PC, thus taking linear time in the size of the circuit. Hence, we will assume smooth and decomposable PCs throughout this paper.

Search Algorithm To certify whether a PC is δ-fair, we search for discrimination patterns in it. If the search concludes without finding any pattern, then we know that the PC is δ-fair; otherwise, we return all or some of the discrimination patterns based on the criteria discussed in Sec. 2. More precisely, we adopt a branch-and-bound search approach (Alg. 2 in the appendix). At each search step, it checks whether the current assignments form a discrimination pattern, and explores extensions by recursively adding variable assignments. Note that while we mainly present the algorithm that returns all discrimination patterns, we can easily tweak it to return the top-k most discriminating patterns: by keeping a running list of top-k patterns and using the k-th highest score as the threshold instead of δ.

As there are exponentially many potential patterns, we rely on a good upper bound to effectively prune the search tree. In particular, we

Algorithm 1 SAMPLE-DISC-PATTERNS$(\mathcal{C}, \mathbf{Z})$

Input: a PC $\mathcal{C}$ over variables $D \cup \mathbf{Z}$ and a threshold δ
Output: a set of sampled discrimination patterns Σ

$\Sigma \leftarrow \{\}$
while not timeout **do**
  $(\mathbf{x}, \mathbf{y}) \leftarrow (\{\}, \{\})$
  while $|\mathbf{x}| + |\mathbf{y}| < n$ **do**
    for $(\mathbf{x}', \mathbf{y}') \in \mathsf{extensions}(\mathbf{x}, \mathbf{y}))$ **do**
      if $\Delta(\mathbf{x}', \mathbf{y}') > \delta$ **then** $\Sigma \leftarrow \Sigma \cup \{(\mathbf{x}', \mathbf{y}')\}$
    $(\mathbf{x}, \mathbf{y}) \leftarrow \mathsf{sample}_{\text{weight:}\Delta(\mathbf{x}'\mathbf{y}')}(\mathsf{extensions}(\mathbf{x}, \mathbf{y}))$
return Σ

use the following as our bound $UB(\mathbf{x}, \mathbf{y}, \mathbf{E}) = \max\{|\max_{\mathbf{u}} P(d \mid \mathbf{x}, \mathbf{y}, \mathbf{u}) - \min_{\mathbf{u}} P(d \mid \mathbf{y}, \mathbf{u})|, |\min_{\mathbf{u}} P(d \mid \mathbf{x}, \mathbf{y}, \mathbf{u}) - \max_{\mathbf{u}} P(d \mid \mathbf{y}, \mathbf{u})|\}$ where $\mathbf{U}$ can be any subset of $\mathbf{Z} \setminus (\mathbf{X} \cup \mathbf{Y} \cup \mathbf{E})$—i.e., the remaining variables to extend the current pattern. The core component of above bound is maximizing or minimizing the conditional probability of the form $P(d \mid \mathbf{y}, \mathbf{u})$ over the values of some $\mathbf{U}$ for a given $\mathbf{y}$. We now show how such optimization can be done tractably for certain classes of PCs.

We use two key observations (formally proved in the appendix). First, to maximize or minimize a conditional probability given some (partial) assignments to a set of free variables, it suffices to consider only their complete assignments. Second, maximizing $P(d \mid \mathbf{x}, \mathbf{u})$ over $\mathbf{U} \subseteq \mathbf{Z} \setminus \mathbf{X}$ is equivalent to maximizing $\frac{P(\mathbf{x}, \mathbf{u} \mid d)}{P(\mathbf{x}, \mathbf{u} \mid \overline{d})}$. Combining these observations, we see that the upper bound $UB(\mathbf{x}, \mathbf{y}, \mathbf{E})$ can be computed easily if we can efficiently maximize and minimize quantities of the form $P(\mathbf{x}, \mathbf{u} \mid d)/P(\mathbf{x}, \mathbf{u} \mid \overline{d})$. In fact, we derive an algorithm with worst-case quadratic time complexity (in the size of the circuit) for PCs that satisfy additional structural constraints. Deferring the algorithmic details and proof of correctness to the appendix, here we instead provide high-level insights to these additional tractability conditions. First, Vergari et al. [30] shows the necessary structural conditions (called compatibility and determinism) such that the quotient of two PCs can be computed tractably and represented as another circuit representation that allows for linear-time optimization [3]. Then all we need is to represent the conditional distributions $P(\mathbf{Z} \mid d)$ and $P(\mathbf{Z} \mid \overline{d})$ as two PCs satisfying those conditions. If the decision variable D appears at the top of the PC over $D \cup \mathbf{Z}$, then the two subcircuits rooted at each child of the root node exactly corresponds to the conditional distributions given D.

Lastly, the special types of discrimination patterns introduced in Sec. 2 can be obtained with minor tweaks to the search algorithm. Details are discussed in Sec. B.

Sampling Algorithm Certifying that there exists no discrimination pattern, among exponentially many possible assignments, is a very hard problem. In real-world settings, one may simply be interested in quickly studying examples

of unfairness that may be present in the model. In fact, this is the goal for many existing fairness auditing tools [26]: to find patterns of bias (potentially using different fairness definitions) for the developer or user to examine. Hence, we introduce efficient sampling-based methods to mine discrimination patterns. While these methods cannot necessarily certify a model to be δ-fair, they can very quickly find a large number of patterns, as we will later show empirically.

Our proposed approach is summarized in Alg. 1. At a high level, each run of the sampling algorithm starts from an empty assignment and incrementally adds one attribute at a time until a complete assignment is obtained. The attribute to be added (i.e. the immediate extension to be explored) at each step is sampled at random with a likelihood proportional to the discrimination score of the resulting assignment. Any assignment explored along the way with a discrimination score above the threshold is added to our set of patterns. Intuitively, at each assignment, we greedily use the discrimination score of its immediate extension as a heuristic for future extensions being discrimination patterns. Observe that the sampling algorithm is computationally inexpensive overall as the only circuit evaluations that need to be performed are a few feed-forward evaluations (linear time) to compute conditionals at each assignment explored. Lastly, it is worth noting that after the discrimination patterns have been sampled, we can utilize similar techniques as described in Sec. 3 to efficiently summarize them through minimal, maximal, and Pareto optimal patterns.

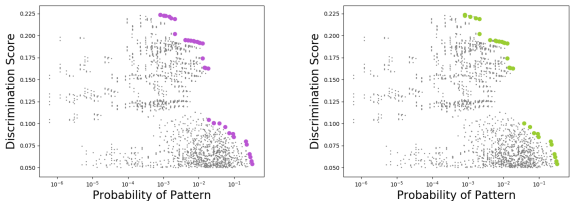
4 EMPIRICAL EVALUATION

We evaluate our algorithms on three datasets: *COMPAS* which is used for recidivism prediction and *Adult* [13] and *Income* [12] for predicting income levels. Each dataset is discretized and has the redundant features and unique values removed. We learn a PC from each dataset using STRUDEL [7], which returns deterministic and structured decomposable PCs as required by our search algorithm.

Exact Search We first evaluate the efficiency of our branch-and-bound search algorithm to find discrimination patterns. As our approach is the first non-trivial method for a general class of probabilistic circuits, we see whether it is more efficient than a naive solution that enumerates all possible patterns. We observe that pruning is effective, resulting in consistent speedup, including some significant improvement in performance as high as 24x speedup on the Adult dataset. We refer the reader to the appendix for complete results. Moreover, note that our method must compute an upper bound at every search step, which has a worst-case quadratic time complexity. However, we see that pruning the search space still improves the overall run time of the algorithm, even with this extra computation. For example, our method explores a little less than half the search space for top-k discrimination patterns with $\delta = 0.1$ on the COMPAS dataset, with runtime 60% that of naive enumeration.

Table 1: Number of discrimination patterns and highest score found by exact search and sampling.

Dataset	Time	δ	# Patterns found		Highest score	
			Exact	Sampling	Exact	Sampling
COMPAS	3s	0.05	347	**751**	**0.2236**	0.2230
		0.10	210	**347**	**0.2236**	0.2230
Income	5s	0.05	209	**1090**	0.1076	**0.1658**
		0.10	3	**225**	0.1076	**0.1659**
Adult	600s	0.05	37167	**113763**	0.6725	**0.6871**
		0.10	30982	**99578**	0.6725	**0.6844**



(a) Exact Pareto front (b) Sampled Pareto front

Figure 1: Discrimination score and probability of all patterns (grey) and summary patterns (colored) for COMPAS.

Sampling The primary motivation for the sampling algorithm is not only to quickly audit a model for discrimination patterns, but in particular to quickly analyze the most interesting patterns. Thus, we first evaluate how efficiently the sampling algorithm is able to find the most discriminatory pattern (which we can find using exact search). The speedups relative to naive enumeration for Compas, Income, and Adult are: 26x, 17x, 48x on average over 10 independent trials (with standard deviation 54x, 14x, 53x, respectively).

For a more direct comparison, we run the exact search algorithm and the sampling method with a specified timeout. We report both the number of patterns found and the highest score in different settings (Tab. 1). We observe that the sampling algorithm consistently outperforms the exact search in both the number of patterns found the scores of patterns found. Thus, we can reliably use the sampling approach to quickly mine many discrimination patterns of significance.

Lastly, we evaluate whether the patterns returned by the sampling algorithm are "interesting". To that end, we analyze the top 10 discrimination patterns produced by our sampling algorithm on COMPAS with a 3 second timeout and find that they indeed correspond to some of the most discriminating patterns in the model (See Sec. D). Furthermore, we compare the Pareto optimal patterns from the sampling algorithm (Fig. 1b) to the true Pareto front obtained through exact search (Fig. 1a) and find that the sampling algorithm contains most of the same patterns in its Pareto front despite its short timeout. More concretely, the Pareto front from the sampling approach contains 30 patterns, of which 27 are in the true set of Pareto optimal patterns (there are 38 total).

References

[1] Solon Barocas and Andrew D Selbst. Big data's disparate impact. *Calif. L. Rev.*, 104:671, 2016.

[2] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, page 0049124118782533, 2018.

[3] Arthur Choi and Adnan Darwiche. On relaxing determinism in arithmetic circuits. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning (ICML)*, 2017.

[4] YooJung Choi, Golnoosh Farnadi, Behrouz Babaki, and Guy Van den Broeck. Learning fair naive bayes classifiers by discovering and eliminating discrimination patterns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10077–10084, 2020.

[5] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

[6] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 1968.

[7] Meihua Dang, Antonio Vergari, and Guy Van den Broeck. Strudel: A fast and accurate learner of structured-decomposable probabilistic circuits. *International Journal of Approximate Reasoning*, 140: 92–115, jan 2022. ISSN 0888-613X.

[8] Adnan Darwiche. A differential approach to inference in bayesian networks. *Journal of the ACM*, 50(3): 280–305, 2003.

[9] Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[10] Amit Datta, Michael Carl Tschantz, and Anupam Datta. Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination. *Proceedings on privacy enhancing technologies*, 2015(1): 92–112, 2015.

[11] Rina Dechter and Robert Mateescu. AND/OR search spaces for graphical models. *Artif. Intell.*, 171(2-3): 73–106, 2007.

[12] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[13] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

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

[15] Thomas L. Griffiths, Nick Chater, Charles Kemp, Amy Perfors, and Joshua B. Tenenbaum. Probabilistic models of cognition: exploring representations and inductive biases. *Trends in Cognitive Sciences*, 14 (8):357 – 364, 2010. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2010.05.004. URL `http://www.sciencedirect.com/science/article/pii/S1364661310001129`.

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

[17] Loren Henderson, Cedric Herring, Hayward Derrick Horton, and Melvin Thomas. Credit where credit is due?: Race, gender, and discrimination in the credit scores of business startups. *The Review of Black Political Economy*, 42(4):459–479, 2015.

[18] Doga Kisa, Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Probabilistic sentential decision diagrams. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2014.

[19] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[20] Anji Liu, Stephan Mandt, and Guy Van den Broeck. Lossless compression with probabilistic circuits. In *International Conference on Learning Representations (ICLR)*, apr 2022.

[21] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Fairness through causal awareness: Learning latent-variable models for biased data. *arXiv preprint arXiv:1809.02519*, 2018.

[22] Marina Meila and Michael I Jordan. Learning with mixtures of trees. *Journal of Machine Learning Research*, 1(Oct):1–48, 2000.

[23] Razieh Nabi and Ilya Shpitser. Fair inference on outcomes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[24] Robert Peharz, Steven Lang, Antonio Vergari, Karl Stelzner, Alejandro Molina, Martin Trapp, Guy Van den Broeck, Kristian Kersting, and Zoubin Ghahramani. Einsum networks: Fast and scalable learning of tractable probabilistic circuits. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, jul 2020.

[25] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.

[26] 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.

[27] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. Interventional fairness: Causal database repair for algorithmic fairness. In *Proceedings of the 2019 International Conference on Management of Data*, pages 793–810, 2019.

[28] Frank A Sonnenberg and J Robert Beck. Markov models in medical decision making: a practical guide. *Medical decision making*, 13(4):322–338, 1993.

[29] Antonio Vergari, YooJung Choi, Robert Peharz, and Guy Van den Broeck. Probabilistic circuits: Representations, inference, learning and applications. *AAAI Tutorial*, 2020.

[30] Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso, and Guy Van den Broeck. A compositional atlas of tractable circuit operations for probabilistic inference. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, dec 2021.

A COMPUTING UPPER BOUNDS

A.1 DISCRIMINATION SCORE

We first formalize the two observations we used to derive our upper bound on the discrimination score.

Lemma 1. Let P be a distribution over $D \cup \mathbf{Z}$ and $\mathbf{x}$ a joint assignment to $\mathbf{X} \subseteq \mathbf{Z}$. Also denote $\mathbf{V} = \mathbf{Z} \setminus \mathbf{X}$. Then for any $\mathbf{U} \subseteq \mathbf{Z} \setminus \mathbf{X}$ the following holds:

$$\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} P(d \mid \mathbf{x}, \mathbf{u}) \leq \max_{\mathbf{v} \in \mathsf{val}(\mathbf{V})} P(d \mid \mathbf{x}, \mathbf{v})$$

That is, to maximize a conditional probability given some (partial) assignments for a set of free variables, it suffices to consider only the complete assignments to those variables. Analogously, this statement holds for minimization as well, with the direction of inequality reversed.

Proof of Lemma 1. Consider any $\mathbf{U} \subset \mathbf{Z} \setminus \mathbf{X}$ and $W \notin \mathbf{U}$. It suffices to show that

$$\forall\, \mathbf{u} \in \mathsf{val}(\mathbf{U}), \quad P(d \mid \mathbf{x}, \mathbf{u}) \leq \max_{w \in \mathsf{val}(W)} P(d \mid \mathbf{x}, \mathbf{u}, w),$$

as the lemma then follows via a simple inductive argument. Denote $\mathsf{val}(W) = \{w_1, w_2, \ldots, w_n\}$. To show that there is at least one $w \in \mathsf{val}(W)$ such that $P(d \mid \mathbf{x}, \mathbf{u}) \leq P(d \mid \mathbf{x}, \mathbf{u}, w)$ for any $\mathbf{u}$, we will show that $P(d \mid \mathbf{x}, \mathbf{u}) > P(d \mid \mathbf{x}, \mathbf{u}, w_i)$ for $i = 1, \ldots, n-1$ implies that $P(d \mid \mathbf{x}, \mathbf{u}) < P(d \mid \mathbf{x}, \mathbf{u}, w_n)$. First, for all $i \leq n-1$ we have:

$$\begin{aligned} &P(d \mid \mathbf{x}, \mathbf{u}) > P(d \mid \mathbf{x}, \mathbf{u}, w_i) \\ &\implies P(d, \mathbf{x}, \mathbf{u}) P(\mathbf{x}, \mathbf{u}, w_i) > P(\mathbf{x}, \mathbf{u}) P(d, \mathbf{x}, \mathbf{u}, w_i). \end{aligned}$$

By taking the sum of both sides of the above inequality, we get:

$$\begin{aligned} &\sum_{i=1}^{n-1} P(d, \mathbf{x}, \mathbf{u}) P(\mathbf{x}, \mathbf{u}, w_i) > \sum_{i=1}^{n-1} P(\mathbf{x}, \mathbf{u}) P(d, \mathbf{x}, \mathbf{u}, w_i) \\ &\implies P(d, \mathbf{x}, \mathbf{u}) P(\mathbf{x}, \mathbf{u}) - \sum_{i=1}^{n-1} P(\mathbf{x}, \mathbf{u}) P(d, \mathbf{x}, \mathbf{u}, w_i) \\ &\qquad > P(d, \mathbf{x}, \mathbf{u}) P(\mathbf{x}, \mathbf{u}) - \sum_{i=1}^{n-1} P(d, \mathbf{x}, \mathbf{u}) P(\mathbf{x}, \mathbf{u}, w_i) \\ &\implies \frac{P(d, \mathbf{x}, \mathbf{u}) - \sum_{i=1}^{n-1} P(d, \mathbf{x}, \mathbf{u}, w_i)}{P(\mathbf{x}, \mathbf{u}) - \sum_{i=1}^{n-1} P(\mathbf{x}, \mathbf{u}, w_i)} > \frac{P(d, \mathbf{x}, \mathbf{u})}{P(\mathbf{x}, \mathbf{u})} \\ &\implies P(d \mid \mathbf{x}, \mathbf{u}, w_n) > P(d \mid \mathbf{x}, \mathbf{u}) \end{aligned}$$

□

Lemma 2. Let P be a distribution over $D \cup \mathbf{Z}$, $\mathbf{x}$ an assignment to $\mathbf{X} \subseteq \mathbf{Z}$, and $\mathbf{U} \subseteq \mathbf{Z} \setminus \mathbf{X}$. Then,

$$\arg\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} P(d \mid \mathbf{x}, \mathbf{u}) = \arg\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{P(\mathbf{x}, \mathbf{u} \mid d)}{P(\mathbf{x}, \mathbf{u} \mid \overline{d})}.$$

Proof. Since $P(d \mid \mathbf{x}, \mathbf{u}) = \frac{P(d, \mathbf{x}, \mathbf{u})}{P(d, \mathbf{x}, \mathbf{u}) + P(\overline{d}, \mathbf{x}, \mathbf{u})} = \frac{1}{1 + P(\overline{d}, \mathbf{x}, \mathbf{u}) / P(d, \mathbf{x}, \mathbf{u})}$, we obtain that

$$\begin{aligned} \arg\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} P(d \mid \mathbf{x}, \mathbf{u}) &= \arg\min_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{P(\overline{d}, \mathbf{x}, \mathbf{u})}{P(d, \mathbf{x}, \mathbf{u})} \\ = \arg\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{P(d, \mathbf{x}, \mathbf{u})}{P(\overline{d}, \mathbf{x}, \mathbf{u})} &= \arg\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{P(\mathbf{x}, \mathbf{u} \mid d)}{P(\mathbf{x}, \mathbf{u} \mid \overline{d})}. \end{aligned}$$

□

Combining these observations, we see that the following upper bound

$$\begin{aligned} \max\{&\left| \max_{\mathbf{u}} P(d \mid \mathbf{x}, \mathbf{y}, \mathbf{u}) - \min_{\mathbf{u}} P(d \mid \mathbf{y}, \mathbf{u}) \right|, \\ &\left| \min_{\mathbf{u}} P(d \mid \mathbf{x}, \mathbf{y}, \mathbf{u}) - \max_{\mathbf{u}} P(d \mid \mathbf{y}, \mathbf{u}) \right| \} \end{aligned}$$

can be computed easily if we can efficiently maximize and minimize quantities of the form $P(\mathbf{x}, \mathbf{u} \mid d) / P(\mathbf{x}, \mathbf{u} \mid \overline{d})$ over values of $\mathbf{U} = \mathbf{Z} \setminus \mathbf{X}$ for some given evidence $\mathbf{x} \in \mathsf{val}(\mathbf{X})$. We now describe our algorithm to compute the upper bound on discrimination score. The pseudocode of our algorithm to maximize such ratio is given in Alg. 3. Again, we assume that the root of the PC is effectively a decision node on D, and thus its children represent the conditional distributions $P(\mathbf{z} \mid d)$ and $P(\mathbf{z} \mid \overline{d})$. Hence we can run Alg. 3 by giving those two children nodes as inputs. Moreover, we can easily tweak the algorithm to minimize the ratio, by changing Line 9 to return the minimum over non-zero values of the recursive calls if they exist, or zero otherwise.

The algorithm assumes PCs that satisfy two structural constraints: determinism and compatibility. A circuit is deterministic if the children of every sum node have disjoint supports (denoted by $\mathsf{supp}(n)$). In other words, for every complete assignment $\mathbf{z}$, at most one of the children nodes will have a non-zero output. In addition, two circuits are compatible if they are: (1) smooth and decomposable; and (2) any pair of product nodes, one from each circuit, that are defined over the same set of variables decompose the variables in the same way. We refer the readers to [30] for a more detailed discussion of compatibility.

Proof of Correctness. We proceed via induction. For the leaves, as they are compatible, by definition their supports are either identical or completely disjoint. Thus, the maximum ratio is 1, 0, or undefined (we also propagate 0 in this case).

Next, consider two compatible product nodes. As they decompose the variables identically, we can order their children nodes such that $n(\mathbf{z}) = \prod_i n_i(\mathbf{z}_i)$ and $m(\mathbf{z}) = \prod_i m_i(\mathbf{z}_i)$, where n_i and m_i are over the same set of variables $\mathbf{Z}_i$. Let us write $\mathbf{U}_i = \mathbf{U} \cap \mathbf{Z}_i$ and $\mathbf{X}_i = \mathbf{X} \cap \mathbf{Z}_i$.

Algorithm 2 SEARCH-DISC-PATTERNS$(\mathbf{x}, \mathbf{y}, \mathbf{E})$

Input: a PC $\mathcal{C}$ over variables $D \cup \mathbf{Z}$ and a threshold δ
Output: a set of discrimination patterns Σ
Data: current pattern $(\mathbf{x}, \mathbf{y}) \leftarrow (\{\}, \{\})$; excluded variables $\mathbf{E} \leftarrow \{\}$

$\Sigma \leftarrow \{\}$
for each $z \in \mathsf{val}(Z)$ for some variable $Z \in \mathbf{Z} \setminus (\mathbf{X} \cup \mathbf{Y} \cup \mathbf{E})$ **do**
  if $Z \in \mathbf{S}$ **then**
    if $\Delta(\mathbf{x} \cup \{z\}, \mathbf{y}) > \delta$ **then** $\Sigma \leftarrow \Sigma \cup \{(\mathbf{x} \cup \{z\}, \mathbf{y})\}$
    if $UB(\mathbf{x} \cup \{z\}, \mathbf{y}, \mathbf{E}) > \delta$ **then** ▷ extend $\mathbf{x}$ and recurse
      $\Sigma \leftarrow \Sigma \cup$ SEARCH-DISC-PATTERNS$(\mathbf{x} \cup \{\mathbf{z}\}, \mathbf{y}, \mathbf{E})$
  if $\Delta(\mathbf{x}, \mathbf{y} \cup \{z\}) > \delta$ **then** $\Sigma \leftarrow \Sigma \cup \{(\mathbf{x}, \mathbf{y} \cup \{z\})\}$
  if $UB(\mathbf{x}, \mathbf{y} \cup \{z\}, \mathbf{E}) > \delta$ **then** ▷ extend $\mathbf{y}$ and recurse
    $\Sigma \leftarrow \Sigma \cup$ SEARCH-DISC-PATTERNS$(\mathbf{x}, \mathbf{y} \cup \{z\}, \mathbf{E})$
if $UB(\mathbf{x}, \mathbf{y}, \mathbf{E} \cup \{Z\}) > \delta$ **then** ▷ exclude Z and recurse
  $\Sigma \leftarrow \Sigma \cup$ SEARCH-DISC-PATTERNS$(\mathbf{x}, \mathbf{y}, \mathbf{E} \cup \{Z\})$
return Σ

Algorithm 3 Best Ratio with Evidence: $\mathrm{BR}(n, m)$

Input: deterministic and compatible PCs n and m over $\mathbf{Z}$; an assignment $\mathbf{x} \in \mathsf{val}(\mathbf{X})$ for $\mathbf{X} \subset \mathbf{Z}$
Output: $\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} n(\mathbf{x}, \mathbf{u}) / m(\mathbf{x}, \mathbf{u})$ where $\mathbf{U} = \mathbf{Z} \setminus \mathbf{X}$

1: **if** n, m are leaf nodes **then**
2: **if** $\mathsf{supp}(n) \cap \mathsf{supp}(m) \neq \emptyset$ and $n(\mathbf{x}) \neq 0, m(\mathbf{x}) \neq 0$ **then**
3: $\mathrm{BR}(n, m) \leftarrow 1$
4: **else**
5: $\mathrm{BR}(n, m) \leftarrow 0$
6: **else if** n, m are product nodes **then**
7: $\mathrm{BR}(n, m) \leftarrow \prod_{i=1}^{|\mathsf{ch}(n)|} \mathrm{BR}(n_i, m_i)$
8: **else** ▷ n, m are sum nodes
9: $\mathrm{BR}(n, m) \leftarrow \max_{n_i \in \mathsf{ch}(n), m_j \in \mathsf{ch}(m)} \frac{\theta_i}{\theta_j} \mathrm{BR}(n_i, m_j)$

Then, we have:

$$\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{n(\mathbf{x}, \mathbf{u})}{m(\mathbf{x}, \mathbf{u})} = \max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{\prod_i n_i(\mathbf{x}_i, \mathbf{u}_i)}{\prod_i m_i(\mathbf{x}_i, \mathbf{u}_i)} = \prod_i \max_{\mathbf{u}_i \in \mathsf{val}(\mathbf{U}_i)} \frac{n_i(\mathbf{x}_i, \mathbf{u}_i)}{m_i(\mathbf{x}_i, \mathbf{u}_i)},$$

leading to Line 7 in Alg. 3.

Finally, consider two deterministic sum nodes. Then for any $\mathbf{z}$, at most one children each of n and m would evaluate non-zero values. That is, the sum nodes can effectively be treated as maximization nodes: e.g. $n(\mathbf{z}) = \sum_i n_i(\mathbf{z}) = \max_i n_i(\mathbf{z})$. Moreover, among all pairs of children n_i, m_j, the ratio $n_i(\mathbf{z})/m_j(\mathbf{z})$ for any fixed $\mathbf{z}$ would be non-zero for at most one pair (again, we treat the ratio that is undefined as 0). Therefore, we have:

$$\frac{n(\mathbf{z})}{m(\mathbf{z})} = \frac{\sum_i n_i(\mathbf{z})}{\sum_j m_j(\mathbf{z})} = \frac{\max_i n_i(\mathbf{z})}{\max_j m_j(\mathbf{z})} = \max_{i,j} \frac{n_i(\mathbf{z})}{m_j(\mathbf{z})}.$$

Thus, we can break down the maximization as the following, corresponding to Line 9:

$$\max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{n(\mathbf{x}, \mathbf{u})}{m(\mathbf{x}, \mathbf{u})} = \max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \max_{i,j} \frac{n_i(\mathbf{x}, \mathbf{u})}{m_j(\mathbf{x}, \mathbf{u})} = \max_{i,j} \max_{\mathbf{u} \in \mathsf{val}(\mathbf{U})} \frac{n_i(\mathbf{x}, \mathbf{u})}{m_j(\mathbf{x}, \mathbf{u})}.$$

□

A.2 DIVERGENCE SCORE

Divergence score was introduced to address the fact that the probability of a pattern, as well as its discrimination score, is an important consideration.

Definition 4 (Divergence score [4])**.** Let P be a probability distribution over $D \cup \mathbf{Z}$ and δ some threshold in $[0.1]$. Further suppose $\mathbf{x}$ and $\mathbf{y}$ are joint assignments to $\mathbf{X} \subseteq \mathbf{S}$ and $\mathbf{Y} \subseteq \mathbf{Z} \setminus \mathbf{X}$, respectively. Then the *divergence score* of $(\mathbf{x}, \mathbf{y})$ is:

$$\min_Q \ \mathrm{D}_{\mathrm{KL}}(P \parallel Q)$$
$$\text{s.t. } \Delta(\mathbf{x}, \mathbf{y}) \leq \delta,\ P(d, \mathbf{z}) = Q(d, \mathbf{z}),\ \forall\, \mathbf{z} \not\supset \mathbf{x} \cup \mathbf{y}$$

where $\mathrm{D}_{\mathrm{KL}}(P \parallel Q) = \sum_{d,\mathbf{z}} P(d, \mathbf{z}) \log(P(d, \mathbf{z})/Q(d, \mathbf{z}))$.

Informally, aims to quantify how much the distribution P needs to be changed in order to remove the discrimination pattern $(\mathbf{x}, \mathbf{y})$. Thus, the patterns with highest divergence scores would tend to have both high discrimination score as well as high probability.

We can search for the top-k patterns ranked by their divergence score using a similar branch-and-bound approach. Choi et al. [4] gives the following upper bound on the divergence scores of extensions of an assignment $(\mathbf{x}, \mathbf{y})$:

$$P(d, \mathbf{x}, \mathbf{y}) \log \frac{\max_{\mathbf{z} \models \mathbf{xy}} P(d \mid \mathbf{z})}{\min_{\mathbf{z} \models \mathbf{y}} P(d \mid \mathbf{z})}$$

$$+ P(\bar{d}\mathbf{xy}) \log \frac{\max_{\mathbf{z} \models \mathbf{xy}} P(\bar{d} \mid \mathbf{z})}{\min_{\mathbf{z} \models \mathbf{y}} P(\bar{d} \mid \mathbf{z})},$$

where $\mathbf{z} \models \mathbf{xy}$ denotes a complete assignment to $\mathbf{Z}$ that agrees with $\mathbf{x}$ and $\mathbf{y}$ on their assignments to variables in $\mathbf{X}$ and $\mathbf{Y}$, respectively.

Observe that this upper bound once again requires efficient maximization and minimization of conditional probability of extensions. Thus, Alg. 3 allows us to leverage this upper bound and straightforwardly extend our exact search algorithm to mine divergence patterns as well.

A.3 RELATIVE DISCRIMINATION SCORE

We define discrimination patterns using an absolute difference in conditional probabilities, but one may wish to characterize discrimination using a quantity that is proportional to the initial prediction probability. For instance, a prediction that goes from 0.15 to 0.05 after disclosing the sensitive attributes could be seen as more problematic than one that goes from 0.8 to 0.7, but they would have the same discrimination score. We can alternatively define a score based on relative difference as the following.

Definition 5 (Relative Degree of Discrimination)**.** Let P be a probability distribution over $D \cup \mathbf{Z}$, and $\mathbf{x}$ and $\mathbf{y}$ be joint assignments to $\mathbf{X} \subseteq \mathbf{S}$ and $\mathbf{Y} \subseteq \mathbf{Z} \setminus \mathbf{X}$, respectively. The *relative discrimination score* of pattern $\mathbf{x}, \mathbf{y}$, defined as $\Delta'(\mathbf{x}, \mathbf{y}) = \frac{P(d|\mathbf{x},\mathbf{y})}{P(d|\mathbf{y})}$.

We can mine discrimination patterns under this notion as well, by using a similar approach to derive the upper bound. That is, to get an upper bound on the relative discrimination score, we independently minimize/maximize $P(d \mid \mathbf{x}, \mathbf{y})$ and $P(d \mid \mathbf{y})$ over extensions, as described in Sec. 3 and Sec. A.1.

B RECOVERING SUMMARY PATTERNS

We can tweak the search algorithm to keep track of the Pareto front of discrimination patterns found so far in each search step. Concretely, we maintain an ordered container storing the probability and discrimination score in increasing order of the former and decreasing order of the latter.

Similarly, finding the set of maximal patterns is almost identical to searching for all discrimination patterns. In particular, when exploring a pattern in the search tree, we declare it to be maximal if no extension of it can possibly be a discrimination pattern, determined by the quadratic-time upper bound on discrimination score.

For minimal patterns, we derive a sub-quadratic time algorithm to examine a set of patterns and recover the minimal ones. Suppose we are given a set of patterns Σ. While a trivial algorithm to extract minimal patterns is quadratic in the number of patterns, we can recover the minimal patterns in time $\mathcal{O}\left(P \cdot N^2 + PNlog(PN)\right)$, where P is the number of potential patterns and $N = |\mathbf{Z}|$. Note that this is particularly of interest when the number of patterns is very large.

First, we pre-process the patterns to identify candidate minimal patterns, which are patterns all of whose extensions are also patterns. Note that we get this for for free in the case of exact search with minor modifications. Consider the poset of all possible assignments $\mathbf{xy}$ ordered by inclusion. We traverse this graph in level order, while maintaining a queue of nodes to visit and a set of assignments that we do not want to visit S. At the beginning of each level, we expand the nodes in S. Then, for each node in the queue for the current level that is not in S, if it is a candidate minimal pattern, we mark it as minimal, and add all its children to S. Otherwise, we add every child not in S to the queue for the next level.

C SAMPLING WITH MEMOIZATION

At a high level, there are key two additions to Alg. 1 in Alg. 4.

First, the weights to sample from immediate extensions are no longer merely their discrimination scores. Instead, we maintain an estimator $\Phi(\mathbf{x}, \mathbf{y})$ at each assignment corresponding to the expected discrimination score of an extension of $(\mathbf{x}, \mathbf{y})$ (not just the immediate extensions by a single variable). $\Phi(\mathbf{x}, \mathbf{y})$ is initialized as the discrimination score for every assignment $\mathbf{x}, \mathbf{y}$. After each sampling run (which refers to the complete extension path taken from the empty assignment $(\{\}, \{\})$ to a complete assignment), we backtrack to update $\Phi(\mathbf{x}, \mathbf{y})$ with our new information about average score of pattern encountered on the path after that assignment. More precisely, one can think of $\Phi(\mathbf{xy})$ as tracking the average discrimination score of a path of extensions from $(\mathbf{x}, \mathbf{y})$, averaged over all extension paths explored so far. Observe that in contrast to Alg. 1, consecutive sampling runs are no longer independent, and later runs have a more informed heuristics for exploring the search space.

Second, at any particular assignment $(\mathbf{x}, \mathbf{y})$, the extensions are no longer directly sampled in proportion to $\Phi(\mathbf{x}, \mathbf{y})$. Instead, we instead introduce a power factor of $\Gamma(\mathbf{x}, \mathbf{y}) = \left(1 + \frac{|\mathbf{x}|+|\mathbf{y}|}{n}\right)$ as a heuristic for how strongly we wish to adhere to our estimator in picking our path. One can view this as a control for exploration versus exploitation as $\Gamma(\mathbf{x}, \mathbf{y})$ varies from 1 to 2. Intuitively, we are more open to exploration early on in our path as given a target pattern $(\mathbf{x}', \mathbf{y}')$, there are initially exponentially many paths to reach it. However, we prefer to exploit our estimator $\Phi(\mathbf{x}, \mathbf{y})$ as the number of extension paths to $(\mathbf{x}', \mathbf{y}')$ reduces later in the sam-

Algorithm 4 SAMPLE-DISC-PATTERNS($\mathcal{C}, \mathbf{Z}$)

Input: a PC $\mathcal{C}$ over variables $D \cup \mathbf{Z}$ and a threshold δ
Output: a set of sampled discrimination patterns Σ

$\Sigma \leftarrow \{\}$
$\Phi(\mathbf{x}, \mathbf{y}) \leftarrow \Delta(\mathbf{x}, \mathbf{y}) \quad \forall \mathbf{xy}$
$\sigma(\mathbf{x}, \mathbf{y}) \leftarrow 1 \quad \forall \mathbf{xy}$
repeat ▷ generate samples until timeout
  $(\mathbf{x}, \mathbf{y}) \leftarrow (\{\}, \{\})$
  $p \leftarrow []$
  while $|\mathbf{x}| + |\mathbf{y}| < n$ **do**
    for $(\mathbf{x}', \mathbf{y}') \in \mathsf{extensions}(\mathbf{x}, \mathbf{y}))$ **do** ▷ extensions by a single variable
      if $\Delta(\mathbf{x}', \mathbf{y}') > \delta$ **then** $\Sigma \leftarrow \Sigma \cup \{(\mathbf{x}', \mathbf{y}')\}$
    $(\mathbf{x}, \mathbf{y}) \leftarrow \mathsf{sample}_{\text{weight:}\Phi(\mathbf{x},\mathbf{y})^{\left(1+\frac{|\mathbf{x}|+|\mathbf{y}|}{n}\right)}}(\mathsf{extensions}(\mathbf{x}, \mathbf{y}))$
    $p \leftarrow p + (\mathbf{x}, \mathbf{y})$
  for $(\mathbf{x}, \mathbf{y}) \in \mathsf{reversed}(p))$ **do** ▷ update estimates
    $t \leftarrow \frac{\Sigma_{i=|\mathbf{x}|+|\mathbf{y}|}^{n} \Phi(p[i])}{n-|\mathbf{x}|-|\mathbf{y}|}$
    $\sigma(\mathbf{x}, \mathbf{y}) \leftarrow \sigma(\mathbf{x}, \mathbf{y}) + 1$
    $\Phi(\mathbf{x}, \mathbf{y}) \leftarrow \frac{\Phi(\mathbf{x},\mathbf{y}) \cdot (\sigma(\mathbf{x},\mathbf{y})-1)+t}{\sigma(\mathbf{x},\mathbf{y})}$
until timeout
return Σ

pling run. We note that this is particularly of significance in settings where the number of variables (and consequently the search space) is large, as for most practical purposes we are interested in quickly finding the most interesting patterns, and not necessarily interested in exploring the search space to extract all possible patterns.

D ADDITIONAL EXPERIMENTAL RESULTS

We mine the top-k patterns for two ranking heuristics (discrimination and divergence score), three values of k (1, 10, 100), and three threshold values δ (0.01, 0.05, 0.1). Tab. 3 reports the speedup in terms of the proportion of the search space visited by our algorithm compared to the naive approach. Note that only the settings in which $\delta = 0.1$ for ranking by discrimination score are reported, because the results are identical for smaller values of δ. We observe that pruning is effective, resulting in consistent speedup, including some significant improvement in performance as high as 24x speedup in the case of mining top-k divergence patterns on the Adult dataset.

Tab. 2 details the speedup of the sampling algorithm relative to naive enumeration, in terms of the number of assignments explored to find the top-1 pattern; each result is the average over 10 independent random trials. Recall that each sampling instance merely requires a few feed-forward evaluations of the circuit (linear time) for computing marginals. Hence, from the table it is clear that the sampling algorithm is much quicker than exact search in finding the patterns with highest scores.

Table 2: Average speedup of sampling v.s. naive enumeration to find top 1 pattern (in terms of proportion of search space explored).

	Discrimination		Divergence	
Dataset	Avg	StdDev	Avg	StdDev
Compas	26x	54x	29x	15x
Income	17x	14x	143x	6x
Adult	48x	53x	49480x	2113x

We also report an extended version of the results reported in Tab. 1[2]. In particular, we compare the number and scores of patterns found by exact search and Alg. 4 with a fixed timeout in various settings (dataset, type of score, and threshold). We find that Alg. 4 consistently outperforms exact search in both number of patterns found and score of highest pattern found across different settings.

[2]The pre-processed data, trained models, and code are available at https://github.com/UCLA-StarAI/PC-DiscriminationPatterns

Table 3: Dataset statistics (number of examples, number of sensitive features S, non-sensitive features N, and number of potential patterns) and speedup of top-k search v.s. naive enumeration, in terms of the fraction of search space explored.

Dataset	Size	S	N	# Pat.	k	Disc. $\delta=0.1$	Divergence $\delta=0.01$	Divergence $\delta=0.05$	Divergence $\delta=0.10$
COMPAS	48834	4	3	15K	1	2.73x	2.17x	1.40x	1.16x
					10	2.68x	1.85x	1.26x	1.10x
					100	2.52x	1.46x	1.13x	1.04x
Income	195665	2	6	11K	1	1.22x	1.50x	1.32x	1.13x
					10	1.20x	1.40x	1.26x	1.08x
					100	1.13x	1.31x	1.15x	1.02x
Adult	32561	4	9	11M	1	1.32x	24.20x	16.72x	10.88x
					10	1.31x	20.44x	14.75x	9.82x
					100	1.29x	16.10x	11.87x	8.40x

Table 4: Number of patterns and highest score of pattern found by exact search and sampling.

Score	Delta	Exact	Sampling	Score	Delta	Exact	Sampling
Discrimination	0.01	584	**980**	Discrimination	0.01	**0.2236**	0.2226
	0.05	347	**751**		0.05	**0.2236**	0.2230
	0.1	210	**347**		0.1	**0.2236**	0.2230
Divergence	0.01	586	**1186**	Divergence	0.01	0.0015	**0.0071**
	0.05	577	**1644**		0.05	0.0002	**0.0009**

(a) COMPAS dataset with a 3 second timeout

Score	Delta	Exact	Sampling	Score	Delta	Exact	Sampling
Discrimination	0.01	953	**2236**	Discrimination	0.01	0.1076	**0.1658**
	0.05	209	**1090**		0.05	0.1076	**0.1658**
	0.1	3	**225**		0.1	0.1076	**0.1658**
Divergence	0.01	840	**2366**	Divergence	0.01	0.0046	**0.0100**
	0.05	818	**2179**		0.05	0.0004	**0.0023**

(b) Income dataset with a 5 second timesout

Score	Delta	Exact	Sampling	Score	Delta	Exact	Sampling
Discrimination	0.01	42467	**127855**	Discrimination	0.01	0.6725	**0.6935**
	0.05	37167	**113763**		0.05	0.6725	**0.6871**
	0.1	30982	**99578**		0.1	0.6725	**0.6844**
Divergence	0.01	35792	**133780**	Divergence	0.01	0.0317	**0.2125**
	0.05	35292	**130232**		0.05	0.0162	**0.1098**

(c) Adult dataset with a 600 second timeout