Machine Learning (2024) 113:1421–1443
https://doi.org/10.1007/s10994-023-06411-z

# Principled diverse counterfactuals in multilinear models

**Ioannis Papantonis**[1] · **Vaishak Belle**[1,2]

Received: 2 June 2022 / Revised: 3 April 2023 / Accepted: 3 October 2023 /
Published online: 10 January 2024
© The Author(s) 2024

**Abstract**
Machine learning (ML) applications have automated numerous real-life tasks, improving both private and public life. However, the black-box nature of many state-of-the-art models poses the challenge of model verification; how can one be sure that the algorithm bases its decisions on the proper criteria, or that it does not discriminate against certain minority groups? In this paper we propose a way to generate diverse counterfactual explanations from multilinear models, a broad class which includes Random Forests, as well as Bayesian Networks.

**Keywords** XAI · Counterfactuals · Multilinear models

## 1 Introduction

The wide adoption of machine learning models in critical applications (Lv et al., 2021; Weissler et al., 2021; Díaz et al., 2019; Vaishya et al., 2020) has sparked a great interest into developing approaches that allow for gaining insights about a model's decision-making process. This is motivated by the fact that many state-of-the-art models function as black-boxes, i.e. their internal reasoning is elusive, which leads to a great challenge; how can one be sure that a model bases its decisions on the proper criteria, or that it does not discriminate against certain groups?

Bayesian networks (BNs) have been traditionally deployed in applications where such considerations are crucial, due to their ability to clearly represent relationships between variables, and incorporate causal information. One of the most celebrated properties of BNs is their ability to compute counterfactual quantities of the from "what would have

Editors: Alireza Tamaddoni-Nezhad, Alan Bundy, Luc De Raedt, Artur d'Avila Garcez, Sebastijan Dumančić, Cèsar Ferri, Pascal Hitzler, Nikos Katzouris, Denis Mareschal, Stephen Muggleton, Ute Schmid

✉ Ioannis Papantonis
i.papantonis@sms.ed.ac.uk

Vaishak Belle
vaishak@ed.ac.uk

[1] The University of Edinburgh, Edinburgh, UK

[2] Alan Turing Institute, London, UK

been the value of $Y$, had $X$ been equal to $x$?", which has been extensively utilized in high-stakes applications, such as in finance (Dhar, 1998; Fatum & Hutchison, 2010), healthcare (Prosperi et al., 2020; VanderWeele, 2020), and criminal justice (Mishler et al., 2021; Sampson et al., 2006).

Having said that, one of the challenges of computing counterfactual quantities is that they require a very careful specification of the mechanisms that underlie the interactions of all the variables included in the model. This is a highly non-trivial task, usually involving domain-expert knowledge as well as hand-crafting the final models. However, ML models often involve a prohibitively high number of variables to allow for accurately specifying every interaction. Complications like this, have led to the emergence of a related line of research within the field of explainability in AI (XAI), where the objective is to identify simplified "computational" counterfactuals. The term computational reflects the underlying idea of this approach, where a classification model is treated as a function, and, given an instance, $X$, the objective is to find a counterfactual instance, $Y$, such that $X$ and $Y$ are as close as possible, but the model predicts a different class for each of them.

This line of research has led to the development of a general framework for producing counterfactual instances for any differentiable classification model, as described in Wachter et al. (2017). Building on top of that, the authors in Mothilal et al. (2020) proposed a method for generating diverse counterfactuals for differentiable models, while the work in Russell (2019) addressed some technical challenges, proposing a new framework that is based on mixed integer programming (MIP).

In this work, we extend the framework in Russell (2019) to the non-linear case, so it allows for generating computational counterfactuals for multilinear models. This model class includes Bayesian network classifiers (BNCs) (Friedman et al., 1997), as well as decision trees, and random forests. In order to address the BNC case we utilize discriminative SPNs (Gens & Domingos, 2012), since they subsume all BNCs, allowing for studying the latter under a unified framework. More specifically, we present the following contributions:

- We show that by taking advantage of a model's multilinear structure, it is possible to formulate an integer linear program (ILP) that is guaranteed to generate valid counterfactuals.
- We demonstrate how one can seamlessly generate multiple diverse counterfactuals using the presented framework.
- We demonstrate how to apply it to decision trees (DTs), random forests (RFs), as well as discriminative SPNs, possibly resulting into an infinite set of counterfactuals.
- We show that the presented framework generalizes other existing frameworks, while we also discuss how it can be easily adjusted to generate alternative forms of explanations.

## 2 Related work

Counterfactuals have a long standing history within philosophy (Lewis, 1974; Ruben, 1990), as well as within the causal modelling community (Pearl, 2009). Furthermore, they have found many applications into XAI, where they have gained significant traction in recent years, partly because there is evidence suggesting that non-technical audience feels more comfortable interpreting such explanations over alternatives, such as propositional rules (Binns et al., 2018). Moreover, counterfactuals inherently convey a notion of "closeness" to the actual world, in the sense that they allow for detecting a set of minimal

changes that can alter a model's decision. In the context of XAI, computing a counterfactual instance corresponds to finding a solution to the following problem:

$$\begin{aligned} &\arg\min_{x'} d(x, x') \\ &s.t.\, f(x') = k \end{aligned} \tag{1}$$

where $f(\cdot)$ is a classifier, $d(\cdot, \cdot)$ is a distance function, $x$ is the factual instance, and $k$ is the category we would like the counterfactual instance, $x'$, to be classified into.

One of the most influential approaches to solving this problem, was presented in the seminal work of Wachter et al. (2017), which is based on Lagrange multipliers, assuming the classifier is differentiable. Building on top of the results in Wachter et al. (2017), Russell (2019) proposed a modified framework, based on *mixed integer programming* (MIP), to generate counterfactuals for linear models. As the author notes, this approach resolves some technical issues of Wachter et al. (2017), while it also provides a principled way for generating diverse counterfactuals, since utilizing only a single counterfactual can be overly restrictive, impeding a better model understanding. In this context, diversity refers to generating multiple different counterfactual instances for the same factual instance, by forcing the resulting counterfactual to respect various constraints. This is equivalent to incorporating additional constraints in (1), on top of $f(x') = k$. For example, a feature, $x_i$, could be forced to lie within a certain range, which corresponds to adding the constraint $\alpha \leq x_i \leq \beta$. This is important for many applications, since it allows for inspecting a model from multiple angles, which can facilitate uncovering biased or otherwise undesired behaviour.

In addition to the above, the MIP approach to generating counterfactuals has been explored in a series of additional works as well. In Cui et al. (2015); Tolomei et al. (2017) the authors propose such a method, especially designed for tree ensemble models. The resulting optimization problems are guaranteed to output a counterfactual instance (or possibly an infinite set of counterfactuals), however they are only applicable to tree models, and they do not support incorporating diversity constraints. Alternative MIP formulations can be found in Kanamori et al. (2020, 2021), which are applicable to linear models, however it is again unclear how to incorporate diversity constraints, as well as whether it is possible to apply these methods to BNCs. Other approaches, such as Karimi et al. (2020); Mohammadi et al. (2021), utilize satisfiability modulo theory (SMT) solvers in order to generate counterfactuals, by first encoding a model, such as a neural network, as a SMT formula, and then using a SMT solver to find satisfying assignments. Such approaches readily allow for generating diverse counterfactuals, however, they require normalized distance functions, so it is unclear whether it is possible to encode the relative importance/cost of modifying a feature. For example, it may be that a categorical feature is much more probable to be in state $k$, compared to the remaining states, however, the proposed distance functions treat all states as equally probable, leading to counterfactuals that may be far from the data manifold. It is also worth noting that SMT solvers result in explicit variable assignments, so they do not support infinite counterfactual sets, rather only a single counterfactual is computed each time the solver in invoked.

Apart from the aforementioned approaches, the problem of generating counterfactual instances has been considered from alternative angles as well. For example, the method in Verma et al. (2022) formulates this problem as a Markov decision process and then utilizes reinforcement learning techniques to compute counterfactuals. This approach is applicable to any model, while it also allows for employing complex distance functions, however it is unclear whether diversity constraints or infinite counterfactual sets are supported by this framework. In a different line of work (Shih et al., 2018; Shi et al., 2020; Choi et al., 2020),

counterfactual generation is based on utilizing intermediate architectures, such as OBDDs (Bryant, 1992). At the core of these works lies the idea of compiling a classifier into a structure that supports counterfactual generation in polynomial time. An advantage of this approach is that such models usually support answering a number of different queries in polynomial time, not only counterfactuals. That being said, the compiled model can be exponentially larger than the original one, while at the same time this approach supports only a single distance function, thus posing limitations on the expressiveness and flexibility of the resulting counterfactuals. The interested readers are referred to Verma et al. (2020) for an in-depth review of the literature related to counterfactual explanations.

## 3 Background

In this section we are going to briefly introduce the models we are going to utilize in the following.

### 3.1 Decision trees

Decision trees (DTs) are tree-like structures that contain a set of conditional control statements, such as $X \leq a$. Each assignment is consistent with exactly one root-to-leaf path, corresponding to the model's outcome. The control statements are arranged in a hierarchical manner, where intermediate nodes represent decisions and leaf nodes can be either class labels (for classification problems) or continuous quantities (for regression problems).

The majority of decision tree learning algorithms operate in a top-down manner, iteratively partitioning the whole dataset into smaller ones, conditioning on the values of the feature that contains the most information, in each iteration. This has led to the development of a number of metrics that quantify the amount of information that is gained, when splitting the dataset according to a specific feature, such as Gini impurity (Bishop, 2006). In turn, these metrics can be used in order to design algorithms that learn DTs from data, such as CART (Moore II, 1987).

An advantage of employing DTs is that their internal rule-based architecture is relatively easy to inspect, allowing for assessing the quality of the model. This is one of the major reasons why DTs are usually utilized in cases where the model's understandability is essential, or in fields like medicine. However, large DTs containing a lot of rules are not easy to interpret anymore, requiring additional explainability tools in order to reason about their internal behaviour (Belle & Papantonis, 2020).

### 3.2 Random forests

As we discussed in the previous section, DTs have been employed in various applications due to the transparency they exhibit, at least as long as they are kept at a reasonable size. However, one of their major limitations is their tendency to overfit the given dataset, leading to high variance models that fail to maintain good performance when dealing with new data.

Random forests (RFs) aim at overcoming this challenge by combining multiple trees, resulting in more stable models with lower variance. The main insight underlying this approach is to sample with replacement from the whole dataset in order to construct multiple new datasets, thus implementing the idea of bagging (Breiman, 1996).

Following that, a decision tree is trained over each of these newly acquired datasets, leading to an ensemble of independent trees. Then, in prediction time, an aggregation measure, such as majority voting (for classification) or averaging (for regression), combines the predictions of each tree in order to generate the prediction of the whole forest.

The procedure described above results in very expressive and accurate models, however this comes at the expense of interpretability, since the whole forest is far more challenging to explain, compared to single decision trees. This has led to the development of various techniques that attempt to explain the inner reasoning of a RF (Belle & Papantonis, 2020).

### 3.3 Sum-product networks

Sum-product networks (SPNs) are rooted directed graphical models that provide for an efficient way of expressing a joint distribution that is defined over a Bayesian network (BN). Assuming all variables are binary (or categorical, in general) SPNs encode this distribution as a multilinear function, $\sum_{\mathbf{x}} f(\mathbf{x}) \prod_{n=1}^{N} \mathbb{1}_{x_n}$. Here $f(\cdot)$ is the (possibly unormalized) probability distribution of the BN, $\mathbf{x}$ is a vector containing all the variables of the model, i.e., $x_1, \ldots, x_N$, the summation is over all possible states, and $\mathbb{1}_{x_n}$ is the indicator function (Darwiche, 2003). In its simplest form, this function contains $2^N$ terms, however, when context-specific independence among the variables is present, it is possible to obtain a compact factorized representation, that is not exponential in the number of the model's variables.

SPNs are strictly more efficient than distributions that are defined over BNs using CPDs, since any such distribution can be transformed to a SPN in polynomial time and space, while the converse is not true (Zhao et al., 2015). Furthermore, SPNs generalize a number of well known models (Rooshenas & Lowd, 2014), such as latent tree (Choi et al., 2011). On top of that, computing marginal or conditional probabilities in SPNs is linear in its size, making them an appealing candidate for practical applications. Since we are considering classification problems, we are interested in discriminative SPNs (Gens & Domingos, 2012), that encode the conditional distribution of a target variable given some predictors, while they also subsume Bayesian network classifiers (BNCs) (Darwiche, 2003). For example, Fig. 1a,b show the two different representations of a naive Bayes classifier.

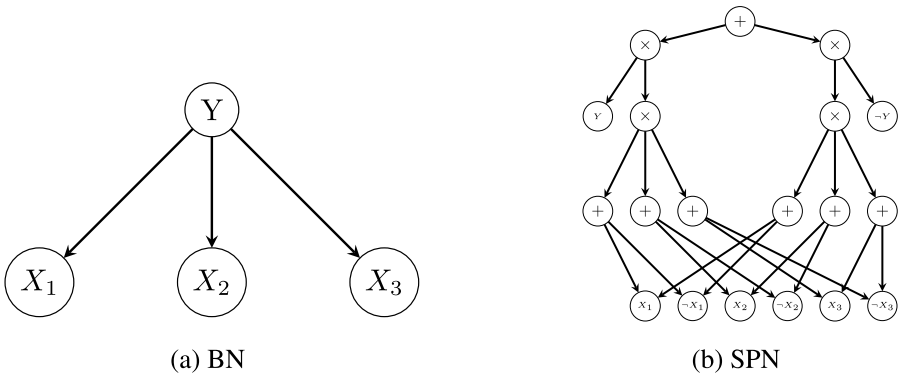



**Fig. 1** BN and SPN representations of the same Naive Bayes model

## 4 Problem derivation

In this section we introduce our approach for generating counterfactuals, inspired by Russell (2019), but addressing one of its key limitations; the range of models it applies to. Specifically, we extend the existing framework to multilinear models, such as DTs and BNCs, as well as ensembles thereof that utilize majority voting, such as RFs. In what follows we assume that all variables are binary, to allow for an easier presentation. However, we provide an extension to the non-binary case, in Sect. 8.

Before going any further, we begin with defining a quantity similar to the decision function, developed in Shih et al. (2018), as follows:

**Definition 1** Let $G : \mathbf{X} \to \{0, 1\}$ be a binary classification function, and $P_0^G(\mathbf{X}), P_1^G(\mathbf{X})$ be multilinear polynomials of indicator variables, where all coefficient are equal to 1 and there is no constant term. Then $P_0^G(\mathbf{X})$ (respectively, $P_1^G(\mathbf{X})$) is called the 0-decision (resp. 1-decision) polynomial of $G$, iff $G(\mathbf{X}) = 0 \Leftrightarrow P_0^G(\mathbf{X}) = 1$ (resp. $G(\mathbf{X}) = 1 \Leftrightarrow P_1^G(\mathbf{X}) = 1$)

Decision polynomials provide for a multilinear representation of arbitrary binary classifiers. In Shih et al. (2018), decision functions play a similar role, however there is no requirement for them to be multilinear. In our work, we have this additional condition in order to be able to derive an optimization problem in ILP format.

In the remaining of this section, we derive some results that hold for decision polynomials, in general. In the following subsections we make the necessary adjustments to apply the developed framework to DTs, RTs and BNCs. The next proposition follows immediately from the definition, and will be used extensively throughout the rest:

**Proposition 2** *Let $G : \mathbf{X} \to \{0, 1\}$, and $P_0^G(\mathbf{X}), P_1^G(\mathbf{X})$ be the decision polynomials. Then* $\forall \mathbf{x} \in \mathbf{X}\ \ P_0^G(\mathbf{x}) + P_1^G(\mathbf{x}) = 1$

The following statement is a simple observation that since each term of a decision polynomial is equal to either 0 or 1, in order for the polynomial to output 0, each term has to be equal to 0.

**Proposition 3** *Let $G : \mathbf{X} \to \{0, 1\}$, and $P_0^G(\mathbf{X}), P_1^G(\mathbf{X})$ be the decision polynomials. Assuming $P_0^G(\mathbf{X}) = T_1(\mathbf{X}) + T_2(\mathbf{X}) + \cdots + T_k(\mathbf{X})$, where each $T_i \in \{0, 1\}$, then $P_0^G(\mathbf{X}) = 0 \Rightarrow T_1(\mathbf{X}) = T_2(\mathbf{X}) = \cdots = T_k(\mathbf{X}) = 0$. The same holds for $P_1^G(\mathbf{X})$.*

Proposition 3 implies that in order to make sure that a decision polynomial outputs 0, it is enough to make sure that each monomial equals 0. The next challenge is due to the fact that these monomials are products of indicator functions, not linear combinations of them. This situation impedes the formulation of generating counterfactuals as a linear optimization problem. A key insight for overcoming this difficulty is that since indicator functions can be equal to either 0 or 1, making sure that not all of them are equal to 1 is sufficient to guarantee that their product is equal to 0. The following proposition states a simple condition that leads to this outcome.

**Proposition 4** *Let $X_1, X_2, \ldots \cdot X_k \in \{0, 1\}$, then $X_1 \cdot X_2 \cdots X_{k-1} \cdot X_k = 1 \Rightarrow X_1 + X_2 + \cdots + X_k = k$ and $X_1 \cdot X_2 \cdots X_{k-1} \cdot X_k = 0 \Rightarrow X_1 + X_2 + \cdots + X_k \leq k - 1$.*

At this point, Propositions 3 and 4 already provide for a set of constraints that are sufficient to ensure that a datapoint is classified as either 0 or 1. For example, if the goal is to generate an instance that belongs in the 1-class, then it is enough to consider the 0-decision polynomial and for each term, say $X_1 \cdot X_2 \cdots X_{k-1} \cdot X_k$, add the constraint $X_1 + X_2 + \cdots + X_k \leq k - 1$. This procedure guarantees that the solution to the problem, $\mathbf{X}$, satisfies $P_0^G(\mathbf{X}) = 0 \Rightarrow P_1^G(\mathbf{X}) = 1$, so it is classified as 1.

However, having said that, storing both polynomials requires additional resources, while it could also be the case that one of them is significantly smaller than the other one, so it would be preferable to express the problem in terms of this polynomial to end up with a more compact optimization problem. A natural way to address this situation would be to define a set of constraints that when satisfied force a term in the decision polynomial to be equal to 1, and the rest equal to 0. The following proposition provides such a set of constraints:

**Proposition 5** *Let* $P_0(\mathbf{X}) = X_{11} \cdot X_{12} \cdots X_{1k} + X_{21} \cdot X_{22} \cdots X_{2m} + \cdots + X_{n1} \cdot X_{n2} \cdots X_{nl}$, *where each* $X_i \in \{0, 1\}$, *be the* 0*-decision polynomial of a model. Furthermore, let the constraints* $X_{11} + X_{12} \cdots + X_{1k} \geq k \cdot \delta_1, X_{21} + X_{22} \cdots + X_{2m} \geq m \cdot \delta_2, \ldots, X_{n1} + X_{n2} \cdots + X_{nl} \geq l \cdot \delta_n, \sum_{i=1}^{n} \delta_i = 1$, *where* $\delta_i \in \{0, 1\}$. *If an assignment,* $\mathbf{X'}$, *satisfies these constraints, then* $P_0(\mathbf{X'}) = 1$. *An analogous statement holds for* $P_1(\mathbf{X})$.

**Require:** A multilinear model $M$
  Compile the $k$-DP of $M$, where $k \in \{0, 1\}$
  Form the objective function
  **if** the counterfactual has to be classified in the $k$-class **then**
    Add the constraints in Proposition 5 to the problem
  **else**
    Apply the inequality constraints in Proposition 4 to every term of the DP
  **end if**

***Proof*** Let $\mathbf{X}$ be an assignment that satisfies all the constraints. From the constraint $\sum_{i=1}^{n} \delta_i = 1$ we have that there is a $j$, such that $\delta_j = 1$. This $\delta_j$ appears in an additional constraint of the form $X_{j1} + X_{j2} \cdots + X_{jp} \geq p \cdot \delta_j \Rightarrow X_{j1} + X_{j2} \cdots + X_{jp} \geq p$. However, it also holds that $X_{j1} + X_{j2} \cdots + X_{jp} \leq p$, so putting these two expressions together we have that $X_{j1} + X_{j2} \cdots + X_{jp} = p \Rightarrow X_{j1} \cdot X_{j2} \cdots X_{jp} = 1$, by proposition 4, which means that $P_0(\mathbf{X}) = 1$. □

We have now developed most of the the necessary machinery to formulate a counterfactual generating optimization problem, shown in Algorithm 1. In the following subsections we address the first two points in Algorithm 1, providing ways to recover the DPs of DTs, RFs and BNCs, as well as discussing how to set the weights of the weighted $l_1$ norm, which is going to serve as our objective function. Furthermore we provide some adjustments that need to be made in order to take into account the characteristics of the aforementioned models.

## 5 Counterfactuals for DTs

Decision trees can be naturally seen as a collection of rules, so in this section we will examine how this set of rules can be used in order to construct a tree's decision polynomial. Transforming DTs to equivalent rule-based classifiers is a well studied problem (Quinlan, 1987). They key observation however, is that it is possible to derive a multilinear representation of a DT over the set of rules it naturally induces.

An example of the general process can be seen in Fig. 2a, which contains a very simple decision tree. It is defined over two continuous variables, $X_1, X_2$, but it can also be seen as a function over its internal rules, $X_1 \leq 10, X_2 \leq 50, X_2 \leq 20$. Utilizing the latter, and traversing the DT bottom-up, it is not difficult to see that the decision polynomials are:

$$P_1^G(X_1, X_2) = \mathbb{1}[X_1 \leq 10] \cdot \mathbb{1}[X_2 \leq 50] + (1 - \mathbb{1}[X_1 \leq 10]) \cdot \mathbb{1}[X_2 \leq 20],$$
$$P_0^G(X_1, X_2) = \mathbb{1}[X_1 \leq 10] \cdot (1 - \mathbb{1}[X_2 \leq 50]) + (1 - \mathbb{1}[X_1 \leq 10]) \cdot (1 - \mathbb{1}[X_2 \leq 20]),$$

where $\mathbb{1}$ is the indicator function.

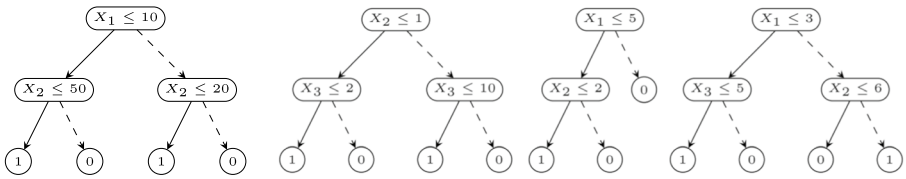
The 1-DT contains all the rules that the DT utilizes to classify an instance in the 1-category, while the 0-DT follows an analogous reasoning. In both polynomials, all monomials are monic, as well as there is no constant term. Furthermore, since for each possible assignment only one root-to-leaf path will be satisfied, each polynomial outputs either 0 or 1, so they are indeed valid DPs. This process exemplifies the general reasoning, which remains unaltered, no matter how large a DT is.

Having the decision polynomials, we are now ready to put all the pieces together. To this end, let $d = (X_1, X_2)$ be a factual datapoint of interest. We utilize the weighted $l_1$ norm, $\|\cdot\|_{1,w}$, and the rule representation of the DT to define the distance between two points as follows:

$$\|d - d'\|_{1,w} = w_1 |\mathbb{1}[X_1 \leq 10] - \mathbb{1}[X_1' \leq 10]| + w_2 |\mathbb{1}[X_2 \leq 20] - \mathbb{1}[X_2' \leq 20]| + w_3 |\mathbb{1}[X_2 \leq 50] - \mathbb{1}[X_2' \leq 50]|,$$

where $w_1, w_2, w_3$ are constants. This is the objective function of the final optimization problem. The last step is to remove the absolute values from the objective function. This is simple to do, since the values of the indicators $\mathbb{1}[X_1 \leq 10], \mathbb{1}[X_2 \leq 20], \mathbb{1}[X_2 \leq 50]$ are known quantities, and $0 \leq \mathbb{1}[\cdot] \leq 1$.

To go on with our example let us also assume that $d(X_1, X_2)$ satisfies $X_1 \leq 10, 20 < X_2 \leq 50$, so it is classified into the 1 class, and that we want to utilize the 0-DP. Applying Proposition 5, the final optimization problem is:




(a) A Decision Tree (b) A Random Forest

**Fig. 2** Examples of decision trees and random forests

$$
\begin{aligned}
&min\ \ w_1(1-\mathbb{1}[X_1' \leq 10]) + w_2\mathbb{1}[X_2' \leq 20] + w_3(1-\mathbb{1}[X_3' \leq 50]) \\
&\quad s.t.\ \mathbb{1}[X_1' \leq 10] + (1-\mathbb{1}[X_2' \leq 50]) \geq 2 \cdot \delta_1, \\
&\quad (1-\mathbb{1}[X_1' \leq 10)] + (1-\mathbb{1}[X_2' \leq 20]) \geq 2 \cdot \delta_2, \\
&\quad \delta_1 + \delta_2 = 1
\end{aligned}
$$

The solution of this problem is guaranteed to be classified as 0. Of course it depends on the values of $w_1, w_2, w_3$, but it is going to be an infinite set of solutions, regardless. For example, if the resulting solution turns out to be $\mathbb{1}[X_1' \leq 10] = 1, \mathbb{1}[X_2' \leq 20] = \mathbb{1}[X_2' \leq 50] = 0$, then every element of the set $\{(X_1, X_2') : X_2' > 50\}$ is a valid counterfactual to $d$, with respect to the decision tree. This is an extension of the framework in Russell (2019), where the outcome was a single point.

Finally, we discuss the amount of constraints that has to be added within the model. As the DPs encode the root-to-leaf paths of the decision tree, the amount of constraints depends on the number of distinct root-to-leaf paths, $m$. The added flexibility of expressing our framework using either of the two DPs, allows for efficiently handling situations that would be otherwise problematic. For example, if there is a DT having only one path that leads to a 0-leaf, and all the remaining ones lead to a 1-leaf, then we can encode everything using the 0-DP in a highly efficient manner, using a single constraint, instead of $m-1$ ones. This demonstrates that the worst-case scenario is when there is an equal number of 0-leaf and 1-leaf paths, in which case the cost of encoding the constraints is the same, no matter which DP is utilized. This means that in the worst case $O(\frac{m}{2})$ constraints would be necessary, each one involving $O(p)$ variables, where p is the length of the longest path in the tree.

## 6 Counterfactuals for RFs

In this section, we examine how to handle ensembles of multilinear models, using RFs as an example. Although the process is similar in spirit, incorporating information from multiple models poses an additional challenge. For example, looking at Fig. 2b we can verify that the 1-DP of each tree is:

$$
\begin{aligned}
&T_1 : P_1^{T_1}(X_1, X_2, X_3) = \mathbb{1}[X_2 \leq 1] \cdot \mathbb{1}[X_3 \leq 2] + (1-\mathbb{1}[X_2 \leq 1]) \cdot \mathbb{1}[X_3 \leq 10], \\
&T_2 : P_1^{T_2}(X_1, X_2, X_3) = \mathbb{1}[X_1 \leq 5] \cdot \mathbb{1}[X_2 \leq 2], \\
&T_3 : P_1^{T_3}(X_1, X_2, X_3) = \mathbb{1}[X_1 \leq 3] \cdot \mathbb{1}[X_3 \leq 5] + (1-\mathbb{1}[X_1 \leq 3]) \cdot (1-\mathbb{1}[X_2 \leq 6])
\end{aligned}
$$

As usual, each individual polynomial encodes all the 0 or 1 assignments of each individual tree, but how can we combine them all together so they encode the behaviour of the forest? A first remark is that in order to make sure that the model outputs, for example, 1, it is enough to enforce a constraint that at most one 0-decision polynomial outputs 1. This would mean that the outcome of at least 2 out of the 3 decision trees is equal to 1, so the whole forest has an output of 1, assuming majority voting.

This kind of reasoning can be applied to ensembles will an arbitrary number of trees, and will be the base of extending the current framework. In fact, it turns out that this

approach corresponds to a generalization of the one we presented for DTs, as we show at the end of this section. The following proposition provides for a way to encode the fact that a decision polynomial is equal to 1.

**Proposition 6** *Let* $P_0^G(\mathbf{X}) = X_{11} \cdot X_{12} \cdots X_{1k} + X_{21} \cdot X_{22} \cdots X_{2m} + \cdots + X_{n1} \cdot X_{n2} \cdots X_{nl}$, *where each* $X_i \in \{0, 1\}$, *be the* 0*-decision polynomial of a model. Furthermore, let the constraints* $X_{11} + X_{12} \cdots + X_{1k} - k \leq \delta - 1, X_{21} + X_{22} \cdots + X_{2m} - m \leq \delta - 1, \ldots, X_{n1} + X_{n2} \cdots + X_{nl} - l \leq \delta - 1$ , *where* $\delta \in \{0, 1\}$. *Then,* $\delta = 0 \Rightarrow P_0^G(\mathbf{X}) = 0$. *An analogous statement holds for* $P_1^G(\mathbf{X})$.

***Proof*** If $\delta = 0$, the constraints can be rewritten as:

$$\begin{aligned} &X_{11} + X_{12} \cdots + X_{1k} \leq k - 1 \\ &X_{21} + X_{22} \cdots + X_{2m} \leq m - 1 \\ &\cdots \\ &X_{n1} + X_{n2} \cdots + X_{nl} \leq l - 1 \end{aligned}$$

By Proposition 4, this means that all terms in the 0-DP are zero, so $P_0(\mathbf{X}) = 0$. □

Proposition 6 can be used as an indicator of whether a DT outputs 0 or 1, but it can be easily extended so it applies to a RF. For example, assuming we utilize the 0-DPs to generate an instance that is classified as 1, adding this set of constraints for every DT in the RF and demanding that at least half of the corresponding indicators are equal to 0, we enforce that the majority of the DTs have an outcome equal to 1, so the whole forest outputs 1.

Furthermore, as it was the case with DTs, utilizing Proposition 6 and both DPs it is now possible to state all the necessary constraints to ensure the desired outcome. However, the same considerations as before apply to the RF case, so it would be desirable to be able to express the optimization problem in terms of a single DP. As it turns out, it is possible to extend Proposition 5 so it can handle the RF case as well:

**Proposition 7** *Let* $T_1, T_2, \ldots, T_m$ *be the DTs of a RF* $F$. *For each* $T_j$, *consider* $P_0^{T_j}(\mathbf{X})$ *and add all the constraints appearing in Proposition 5, except for the last one, which is replaced by* $\sum_{i=1}^{n} \delta_{ji} = \delta_j$, *where* $\delta_{ji}$ *appears in the i-th constraint of the j-th tree and* $\delta_j \in \{0, 1\}$ *is a newly introduced variable. Finally, add the constraint* $\sum_{i=1}^{m} \delta_i > \left\lfloor \frac{m-1}{2} \right\rfloor$. *If an assignment,* $X'$, *satisfies these constraints, then* $P_0^F(X') = 1$.

***Proof*** The last constraint enforces that more that half of the $\delta_i$'s are equal to 1. The result follows, since each $\delta_i$ is an indicator a DT's outcome. This means that the majority of the DTs classify the resulting instance in the desired category. □

These results connect the behaviour of a single model to the behaviour of the ensemble, allowing to control the number of models that output a certain outcome. However, tree ensembles present an additional challenge that needs to be addressed; that is, we need to make sure that the solution of the optimization problem is consistent. In this setting, we use the term consistency in the sense that if the solution dictates that a condition of the form

$X \leq \alpha$ holds, then all the conditions of the form $X \leq \beta$, where $\alpha \leq \beta$ hold as well. Furthermore, by the same reasoning, if a condition $X \leq \alpha$ does not hold, then no condition $X \leq \beta$, where $\alpha \geq \beta$ should hold. To this end, we have the following definition:

**Definition 8** Let $T_1, T_2, \ldots, T_n$ be DTs and $X$ one of the variables in their scope. We define:

- $F_x = \{X \leq a : X \leq a \in F(T_i), i \in \{1, 2, \ldots, n\}\}$, where $F(T_i)$ is the set of all the internal rules in $T_i$. In turn, $F_x$ is the set of all the rules among all the trees that involve variable $X$.
- Furthermore, let $X \leq a$ be an element of $F_x$, and define $F_x^+(X \leq a) = \{X \leq b$: $X \leq b \in F_x, b \geq a\}$, the set of rules involving $X$ where the threshold is larger than $a$, and $F_x^-(X \leq a) = \{X \leq b : X \leq b \in F_x, b \leq a\}$, the rules where the threshold is smaller than $a$.

The following proposition provides a way to achieve consistency by enforcing a set of constraints:

**Proposition 9** *Let $T_1, T_2, \ldots, T_n$ be DTs that form a RF. Then, the constraints $\sum_{f_i \in F_x^+(X \leq a)} \mathbb{1}[f_i] \geq |F_x^+(X \leq a)| \cdot \mathbb{1}[X \leq a]$ and $\sum_{f_i \in F_x^-(X \leq a)} \mathbb{1}[f_i] \leq |F_x^-(X \leq a)| \cdot \mathbb{1}[X \leq a]$, guarantee that the final solution is consistent wrt the feature $X \leq a$.*

***Proof*** Let $X \leq a$ be a feature in a RF. The first case we are going to examine is when this rule is not satisfied, meaning that $\mathbb{1}[X_1 \leq a] = 0$. Then, the first constraint reduces to $\sum_{f_i \in F_x^+(X \leq a)} \mathbb{1}[f_i] \geq 0$, which always holds. The second constraint however becomes $\sum_{f_i \in F_x^-(X \leq a)} \mathbb{1}[f_i] \leq 0$, which means that $\sum_{f_i \in F_x^-(X \leq a)} \mathbb{1}[f_i] = 0$, forcing all rules within $F_x^-(X \leq a)$ to be false as well, thus guaranteeing consistency wrt to the feature $X \leq a$.

The other case we need to examine is when $\mathbb{1}[X_1 \leq a] = 1$. Then, the first constraint becomes $\sum_{f_i \in F_x^+(X \leq a)} \mathbb{1}[f_i] \geq |F_x^+(X \leq a)|$, which implies that all rules within $F_x^+(X \leq a)$ are also satisfied. The second constraint becomes $\sum_{f_i \in F_x^-(X \leq a)} \mathbb{1}[f_i] \leq |F_x^-(X \leq a)|$, which always holds. □

Looking at Proposition 9 we see that two constraints per feature are enough to guarantee consistency. We can now examine the number of constraints that are required in order to generate a counterfactual set from a RF. Clearly, we have to include the counterfactual generating constraints as well as the consistency ones. The former, amounts to incorporating the DP of each tree in the forest. As discussed in the previous section, assuming there are $N$ trees, $O(\frac{Nm}{2})$ constraints are required in the worst case, where $m$ is the maximum number of distinct paths among all $N$ trees. For the latter, we have to add two constraints per feature,meaning that $O(NF^*)$, where $F^* = max_{T_1,\ldots,T_N}(F_{T_i})$, constraints are required. Combining these together, in the worst case $O(N(\frac{m}{2} + F^*))$ constraints are needed to define a counterfactual generating problem.

We are now ready to demonstrate how to generate counterfactuals for RFs, by combining Propositions 6, 5 and 9. Returning to our running example, let $d = (X_1, X_2, X_3)$ be a datapoint that satisfies the conditions $X_1 \leq 3$, $X_2 \leq 1$, $X_3 \leq 2$, meaning that all 3 DTs classify $d$ as 1. Assuming we utilize the 1-DPs, the following generates a set of counterfactuals that are classified as 0:

$$
\begin{aligned}
&\min w_1(1-\mathbb{1}[X_1 \le 5]) + w_2(1-\mathbb{1}[X_1 \le 3]) + w_3(1-\mathbb{1}[X_2 \le 1]) \\
&\quad + w_4(1-\mathbb{1}[X_2 \le 2]) + w_5(1-\mathbb{1}[X_2 \le 6]) + w_6(1-\mathbb{1}[X_3 \le 2]) \\
&\quad + w_7(1-\mathbb{1}[X_3 \le 10]) + w_8(1-\mathbb{1}[X_3 \le 5]) \text{ s.t.} \\
&\quad \left.\begin{array}{l}
\mathbb{1}[X_2 \le 1] + \mathbb{1}[X_3 \le 2] - 2 \le \delta_1 - 1, \\
1 - \mathbb{1}[X_2 \le 1] + \mathbb{1}[X_3 \le 10] - 2 \le \delta_1 - 1 \\
\mathbb{1}[X_1 \le 5] + \mathbb{1}[X_2 \le 2] - 2 \le \delta_2 - 1, \\
\mathbb{1}[X_1 \le 3] + \mathbb{1}[X_3 \le 5] - 2 \le \delta_3 - 1 \\
1 - \mathbb{1}[X_1 \le 3] + 1 - \mathbb{1}[X_2 \le 6] - 2 \le \delta_3 - 1, \\
\delta_1 + \delta_2 + \delta_3 \le 1
\end{array}\right\} 0 \text{ generating constraints} \\
&\quad \mathbb{1}[X_1 \le 5] \ge \mathbb{1}[X_1 \le 3] \Big\} X_1 \text{ consistency constraints} \\
&\quad \left.\begin{array}{l}
\mathbb{1}[X_2 \le 2] + \mathbb{1}[X_2 \le 6] \ge 2\mathbb{1}[X_2 \le 1], \\
\mathbb{1}[X_2 \le 6] \ge \mathbb{1}[X_2 \le 2] \\
\mathbb{1}[X_2 \le 1] \le \mathbb{1}[X_2 \le 2], \\
\mathbb{1}[X_2 \le 1] + \mathbb{1}[X_2 \le 2] \le 2\mathbb{1}[X_2 \le 6]
\end{array}\right\} X_2 \text{consistency constraints} \\
&\quad \left.\begin{array}{l}
\mathbb{1}[X_3 \le 5] + \mathbb{1}[X_3 \le 10] \ge 2\mathbb{1}[X_3 \le 2], \\
\mathbb{1}[X_3 \le 10] \ge \mathbb{1}[X_3 \le 5] \\
\mathbb{1}[X_3 \le 2] \le \mathbb{1}[X_3 \le 5], \\
\mathbb{1}[X_3 \le 2] + \mathbb{1}[X_3 \le 5] \le 2\mathbb{1}[X_3 \le 10]
\end{array}\right\} X_3 \text{ consistency constraints}
\end{aligned}
$$

Finally, we show that the optimization schema for RTs is indeed a generalization of the DT one. Without loss of generality, we can assume that we are using the 0-DP to formulate the optimization problem, since the same argument applies to the other case as well. Let $T$ be a DT, and let us first consider the case of generating an instance that is classified as 1. To this end, we will utilize the constraints in Proposition 6, treating $T$ as a trivial RF, $F$, comprised of just a single tree.

The 0-DP of $F$ is identical to the 0-DP of $T$, so $P_0^F(X) = P_0^T(X) = X_{11} \cdot X_{12} \cdots X_{1k} + X_{21} \cdot X_{22} \cdots X_{2m} + \cdots + X_{n1} \cdot X_{n2} \cdots X_{nl}$. Following the procedure in proposition 6, we have to add the constraints:

$$
\begin{cases}
X_{11} + X_{12} \cdots + X_{1k} - k \le \delta_1 - 1 \\
X_{21} + X_{22} \cdots + X_{2m} - m \le \delta_1 - 1 \\
\cdots \\
X_{n1} + X_{n2} \cdots + X_{nl} - l \le \delta_1 - 1 \\
\delta_1 \le 0
\end{cases}
\xRightarrow{\delta_1 = 0}
\begin{cases}
X_{11} + X_{12} \cdots + X_{1k} \le k - 1 \\
X_{21} + X_{22} \cdots + X_{2m} \le m - 1 \\
\cdots \\
X_{n1} + X_{n2} \cdots + X_{nl} \le l - 1
\end{cases}
$$

where the implication follows from the fact that since $\delta_1 \ge 0$, the constraint $\delta_1 \le 0$ leads to $\delta_1 = 0$. Looking at the right-hand side, we see that these are exactly the constraints that result from Proposition 4, thus establishing the desired equivalence for this case.

Furthermore, we have to examine the case of generating an instance that is classified as 0, using the 0-DT. Again, treating $T$ as a trivial RF, Proposition 7 can be used to obtain the following set of sufficient constraints:

$$\begin{cases} X_{11} + X_{12} \cdots + X_{1k} \geq k \cdot \delta_1 \\ X_{21} + X_{22} \cdots + X_{2m} \geq m \cdot \delta_2 \\ \cdots \\ X_{n1} + X_{n2} \cdots + X_{nl} \geq l \cdot \delta_n \\ \sum_{i=1}^{n} \delta_i = \delta \\ \delta > 0 \end{cases} \overset{\delta=1}{\Longrightarrow} \begin{cases} X_{11} + X_{12} \cdots + X_{1k} \geq k \cdot \delta_1 \\ X_{21} + X_{22} \cdots + X_{2m} \geq m \cdot \delta_2 \\ \cdots \\ X_{n1} + X_{n2} \cdots + X_{nl} \geq l \cdot \delta_n \\ \sum_{i=1}^{n} \delta_i = 1 \end{cases}$$

where this time the implication follows from the fact that $\delta \in \{0, 1\}$, so the constraint $\delta > 0$ leads to $\delta = 1$. Again, these are exactly the constraints in Proposition 5. Additionally, since there is only a single tree in forest, it is not necessary to include the consistency constraints, because inconsistencies only arise when combining multiple trees. This concludes the proof of the claim that the RF constraints generalize the DT ones.

## 7 Counterfactuals for BNCs

The last class of models we are going to incorporate within our framework are BNCs over binary variables, representing them as SPNs. Retrieving the DPs of an SPN is relatively straightforward, utilizing their interpretation as a collection of tree models (Zhao et al., 2016). Of course, this means that in the worst case an SPN is a collection of an exponential amount of trees, one for each joint variable assignment. In turn, this means that in this scenario an exponential amount of constraints is needed in order to encode a decision polynomial. This is in tune with known complexity results that utilize tractable structures to compute counterfactuals (Shih et al., 2018).

Despite that, SPNs have been particularly powerful in applications where there is context-specific independence (Boutilier et al., 1996) among the variables, providing very compact representations. This means that although the worst case scenario is exponential, there are situations where it is possible to define the optimization problem using significantly fewer constraints. For example, let us assume that the 0-DP of the SPN in Fig. 1b is equal to $P_0(X_1, X_2, X_3) = X_1X_2X_3 + X_1X_2(1 - X_3)$. The terms in the polynomial imply that when $X_1 = X_2 = 1$, a datapoint is classified in the 0 class, regardless of what value $X_3$ has. In turn, we end up with the reduced 0-DP $P_0(X_1, X_2, X_3) = X_1X_2$.

The above process can be repeated iteratively, eliminating variables that are not relevant, given some context, just like $X_3$ was irrelevant, given the context $X_1 = X_2 = 1$. A simple way to achieve this elimination is whenever encountering two terms differing in only one factor, to substitute both of them with a new term that is equal to their common factors. For example, let us assume that this time the 0-DP is equal to $P_0(X_1, X_2, X_3) = X_1X_2X_3 + X_1X_2(1 - X_3) + X_1(1 - X_2)X_3 + X_1(1 - X_2)(1 - X_3)$. It is not difficult to observe that this polynomial is equal to 1, only when $X_1 = 1$, meaning it can be reduced to a simpler form. Applying our strategy leads to:

$$\begin{aligned} P_0(X_1, X_2, X_3) &= \underbrace{X_1X_2X_3 + X_1X_2(1 - X_3)}_{X_3 \text{is irrelevant, given} X_1 = X_2 = 1} + \underbrace{X_1(1 - X_2)X_3 + X_1(1 - X_2)(1 - X_3)}_{X_3 \text{is irrelevant, given} X_1 = 1, X_2 = 0} \\ &= \underbrace{X_1X_2 + X_1(1 - X_2)}_{X_2 \text{is irrelevant, given} X_1 = 1} = X_1, \end{aligned}$$

which exactly matches our observation. In the same way, we can handle decision polynomials, in general, possibly leading to a significant reduction in size, whenever sufficient context-specific information is available.

## 8 Parameters, prime implicants, the non-binary case and diversity

### 8.1 Parameters

In this section we will discuss some approaches to set the weights, $\mathbf{w}$, in the $\ell_1$ norm as well as some possible extensions. In the original work of Wachter et al. (2017), the inverse of the median absolute deviation (MAD) of a feature is utilized:

$$\text{MAD}_k = \text{median}_{j \in D}(\mid X_{j,k} - \text{median}_{l \in D}(X_{l,k}) \mid) \tag{2}$$

where $D$ is the dataset, and $X_{i,k}$ denotes the value of feature $k$, in data point $i$.

As the authors argue, some advantages of this particular choice is that it captures the intrinsic volatility of a feature, as well as it is more robust to outliers, compared to using the standard deviation. However, MAD is inappropriate when using binary features, since in this case it is always equal to zero (Russell, 2019). However, for the DT and RF cases, although the optimization problem is expressed in terms of binary variables, reflecting the nature of their intrinsic splitting rules, the variables themselves can be continuous. This results in an interesting situation, where both the MAD and the inverse standard deviation are valid weighting options. For example, consider a feature $k$ and branching rule of the form $r = (X_k \leq \alpha)$, where $X \in \mathbb{R}$ is a feature and $\alpha \in \mathbb{R}$ is a constant. We can now define the set of all instances in the dataset that satisfy $r$:

$$S_r = \{d \in D : X_{d,k} \leq \alpha\}$$

Now it is possible to calculate MAD with respect to rule $r$, by simply replacing all appearances of $D$ with $S_r$, in (2). This methodology provides a way to utilize MAD to generate the weights for DTs and RFs, however, this is not applicable to BNCs, where the inverse standard deviation should be used.

A special case worth mentioning arises when all weights are equal to 1. Then, the resulting distance, $\| \cdot \|_{1,\mathbf{1}}$, reduces to the Hamming distance (Shi et al., 2020), and the solution of the minimization problem reflects the smallest number of changes that are necessary for the model to change its output. As a matter of fact, this number has already gained significant attention within an emergent line of research regarding explainability approaches in Bayesian classifiers, where it is known as the *robustness* of a classifier (Shi et al., 2020). However, existing methods are applicable only when utilizing the Hamming distance, which does not allow for assigning different weights to features. In this sense, our framework extends current approaches, since it allows for computing the robustness of a classifier under alternative metrics, that admit non-uniform feature weights, reflecting the relative importance of each term.

### 8.2 Prime implicants

Although the focus of this work has been on generating counterfactuals, it is possible to generate *prime implicant (PI) explanations* (Shih et al., 2018) as well, by making

a few minor adjustments. Unlike counterfactual explanations that compute a minimal set of changes enough to alter the model's decision, PI-explanations compute a minimal set of feature values that is enough to maintain the model's decision, no matter the values of the remaining features. Moreover, as it was the case with the counterfactual explanations, the proposed framework allows for assigning non-uniform weights to each feature, something that is not possible using symbolic approaches, such as Shi et al. (2020).

The procedure of computing the prime implicants of an instance is a simple modification of the one developed for computing counterfactuals. Let us assume the instance of interest is $\mathbf{X} = (X_1, X_2, \dots, X_n)$. Furthermore, without loss of generality, we can assume that it is classified as 1, by the model. To compute the prime implicants of X, we have to form the objective function (with all coefficients equal to 1) and all the constraints that are necessary so the solution to the optimization problem is classified as 1, both of which should be performed in the same way as discussed in the main text. Finally, instead of minimizing this function, we have to maximize it. Intuitively, by doing so, we ask what is the largest set of features that can change values, without altering the model's decision.

Let us assume that the solution to this problem dictates that variables $X_{i_1}, X_{i_2}, \dots, X_{i_k}$, should change values. while the variables in $\mathbf{Z} = \{X_1, X_2, \dots, X_n\} \setminus \{X_{i_1}, X_{i_2}, \dots, X_{i_k}\}$ should not. Then $\mathbf{Z}$ is equal to the prime implicants. To see this, let us assume that $\mathbf{Z}$ contains $m$ elements, and that the number of prime implicants of $\mathbf{X}$ are $l < m$. Then, this means that as long as these $l$ variables maintain their values, the model will classify the datapoint as 1. In turn, this means that all the remaining $n - l$ variables can switch values, and that this would be a feasible solution to the optimization problem of the previous paragraph. Now this leads to a contradiction, since by assumption the solution alters the values of $n - m$ variables, meaning that the inequality $n - m \geq n - l \Rightarrow l \geq m$ should hold, which is not possible. A similar argument makes sure that conditional prime implicants can be generated by just incorporating the constraint that the conditioning variable maintains its value. This can be readily done, since it exactly corresponds to adding a diversity constraint.

### 8.3 The non-binary case

So far we have assumed that all variables (or rules) are binary, but it is possible to extend our framework to the non-binary case, by utilizing a simple transformation. In general, let $X$ be a variable taking values in $\{0, 1, \dots, k\}$. We can now introduce $k$ new binary variables, $X_0, X_1, \dots, X_k$, such that $X_i = 1 \Leftrightarrow X = i$. Furthermore, we need to add the constraint $\sum_{i=0}^{k} X_i = 1$ to enforce that $X$ takes exactly one value. Employing this trick it is immediate to handle the non-binary case.

### 8.4 Diversity

One of the benefits of our proposal is that it is seamless to generate diverse counterfactuals. This is not true for many of the existing techniques, but it is a benefit of employing ILP, as recognized by Russell (2019). In the BNC case it is as simple as just setting the variables to their desired values, leaving everything else intact. In the DT and RF cases, since variables variables may be continuous, a user could ask for counterfactuals that satisfy conditions such as $a \leq X \leq b$, for a variable $X$ (or a set of variables). This is again easy to handle, since the condition $a \leq X \leq b$, is enough to decide the values of some of the constraints

in $F_x$. Then, it is just a matter of plugging these values into the optimization problem and proceeding as normal, leaving the rest unchanged.

## 9 Experiments

In this section we will demonstrate some of the advantages of utilizing the proposed framework. To this end, we will examine three different case studies, based on the COMPAS, LSAT, and Congressional Voting Records datasets. For the first two, we are going to employ a DT and a RF, respectively, while for the last one we use a Naive Bayes Classifier, although any BNC can be used. In Tables 1, 2, counterfactual conditions are in bold, while diversity conditions are underlined, inside a parenthesis.

*COMPAS* is a popular algorithm for assessing the likelihood that a person will reoffend (recidivate) within two years from being released from prison. It has drawn significant attention within the fairness in AI community, due to the number of biases it exhibits, such as favoring white inmates against black ones (Dressel & Farid, 2018). The dataset contains the COMPAS training variables (age, race, sex, number of prior crimes, number of juvenile felonies), whether the inmate actually reoffended within a 2-year period (2 year residivism), as well as the final score generated by the algorithm. A DT was trained on this dataset, predicting the risk of reoffending.

Table 1 shows the records of 5 inmates, where the first row represents the factual datapoint, the second an unconstrained counterfactual, and the third one is making use of the

**Table 1** COMPAS dataset instances

| | Sex | Age | Race | Juvenile felonies | Prior crimes | 2 year residivism | Outcome |
|---|---|---|---|---|---|---|---|
| Factual | Male | 33 | Caucasian | 0 | 2 | Yes | Low score |
| Counterfactual | Male | 33 | Caucasian | **>0** | 2 | Yes | High score |
| Diverse counterfactual | **Female** | 33 | Caucasian | (=0) 0 | 2 | Yes | High score |
| Factual | Male | 21 | Black | 0 | 0 | Yes | High score |
| Counterfactual | Male | **≤ 19.5** | Black | 0 | 0 | Yes | Low score |
| Diverse counterfactual | Male | (>20) **> 21.5** | Black | 0 | 0 | Yes | Low score |
| Factual | Male | 27 | Black | 0 | 0 | No | Low score |
| Counterfactual | Male | 27 | Black | 0 | **>4.5** | No | High score |
| Diverse counterfactual | Male | **<21.5** | Black | 0 | (=0) 0 | No | High score |
| Factual | Male | 32 | Black | 0 | 0 | No | Low score |
| Counterfactual | Male | 27 | Black | 0 | **>1.5** | No | High score |
| Diverse counterfactual | Male | 32 | **Caucasian** | **≥ 1** | (≤ 1) 0 | No | High score |
| Factual | Male | 43 | Caucasian | 0 | 2 | No | Low score |
| Counterfactual | **Female** | 43 | Caucasian | 0 | 2 | No | High score |
| Diverse counterfactual | (Male) | 43 | Caucasian | 0 | **>4** | No | High score |

**Table 2** LSAT dataset instances

| | Sex | LSAT | Race | UGPA | Outcome |
|---|---|---|---|---|---|
| Factual | Male | 34 | White | 3 | Pass |
| Counterfactual | Male | ≤ **19.25** | White | 3 | Fail |
| Diverse counterfactual | Male | (≥25) 34 | **Black** | < **1.95** | Fail |
| Factual | Male | 36.5 | White | 3.2 | Pass |
| Counterfactual | Male | ≤ **20.75** | **Black** | ≤**1.95** | Fail |
| Diverse counterfactual | Male | ≤ **19.25** | (White) White | ≤ **2.15** | Fail |
| Factual | Female | 43 | White | 2.8 | Pass |
| Counterfactual | Female | ≤ **26.75** | White | 2.8 | Fail |
| Diverse counterfactual | Female | (<20)≤ **19.25** | White | ≤ **2.15** | Fail |
| Factual | Male | 35 | White | 2.7 | Pass |
| Counterfactual | Male | 35 | **Black** | ≤ **1.95** | Fail |
| Diverse counterfactual | Male | ≤ **19.25** | (White) White | 2.7 | Fail |
| Factual | Male | 33 | White | 3 | Pass |
| Counterfactual | Male | 33 | **Black** | ≤ **1.85** | Fail |
| Diverse counterfactual | Male | ≤**19.25** | (White) White | 3 | Fail |

diversity constraints. Looking at the first instance, we see that the unconstrained counterfactual is in fact an infinite counterfactual set, since any instance satisfying "juvenile felonies > 0" is a valid counterfactual. In contrast, all of the methods in Kanamori et al. (2020, 2021); Karimi et al. (2020); Mohammadi et al. (2021); Verma et al. (2022) generate single instances, so it would be impossible to attain an infinite set, while methods that may yield infinite counterfactual sets, such as Cui et al. (2015); Tolomei et al. (2017), do not support diverse counterfactuals. Our approach supports both of these features which enables acquiring more in-depth model insights, since although having just a single counterfactual would still be useful, uncovering the underlying condition that needs to change ("juvenile felonies > 0") is more informative than providing specific realizations of this change (e.g. "juvenile felonies = 1"). The same observation holds for all of the instances in Table 1, since each one is associated with an infinite counterfactual set. This feature of our approach was enabled by representing the underlying DT as a collection of propositional rules, and formulating the ILP problem in terms of the obtained rules, instead of the plain input variables (in this case, age, race, sex, number of prior crimes, number of juvenile felonies).

At this point, focusing on the first instance, the unconstrained counterfactual alone does not provide any indications that the underlying model exhibits biased behaviour. This presents a great opportunity to demonstrate the advantage of generating diverse counterfactuals, since by enforcing the constraint "juvenile felonies = 0", the resulting counterfactual suggests that had the inmate been female, the model would have predicted a high score of reoffending, thus uncovering the hidden bias exhibited by the model. The same holds for the fourth instance in Table 1, which highlights how diversity constraints can be used to guide an analysis that aims at exposing biased models (more details in the following experiment). Compared to other approaches that support generating diverse counterfactual, such as the ones in Shih et al. (2018); Shi et al. (2020); Choi et al. (2020); Karimi et al. (2020); Mohammadi et al. (2021), our framework is the only one that allows for infinite sets of diverse counterfactuals (as shown for example in the second instance in Table 1). Finally, in contrast to the aforementioned approaches, our method is not based on the Hamming

distance, so it is possible to incorporate the relative importance of each variable in the objective function, resulting in counterfactuals that are closer to the data manifold.

*LSAT* LSAT is another popular dataset in the fairness literature, since it exhibits a strong bias against black people, too. In this setting, the model has to predict whether students will pass the bar, based on their sex, age, law school admission test (lsat), and undergraduate gpa (ugpa). Table 2 shows 5 student records, along with the model's prediction. All of the instances are associated with infinite counterfactual sets, which, as discussed in the previous experiment, facilitates gaining a more thorough understanding of the underlying model.

For example, focusing on the first instance, making sure that lsat is less than 19.25 is enough to alter the model's prediction. While looking at this counterfactual does not reveal any biases, the relative discrepancy between the factual value of lsat and the counterfactual condition (about 15 points), should be an indicator that constraining the lsat value closer to its factual value, could expose biased behavior. While incorporating inequality constraints is in general very challenging, our framework can handle them seamlessly, since it reduces to assigning specific values to some of the indicator variables. In contrast, although the approaches in Shih et al. (2018); Shi et al. (2020); Choi et al. (2020); Karimi et al. (2020); Mohammadi et al. (2021) accommodate for generating diverse counterfactuals, they only allow for constraints of the form $x_i = \alpha$, so it would not be possible to constrain the range of a variable and conduct the same kind of analysis. Going back to the instance under consideration, enforcing that lsat is greater than 25 leads to a counterfactual that clearly showcases the bias in the model, since the student's race is a factor that can alter the model's prediction. This case captures the advantages that come with diversity and inequality constraints, when interrogating a model for biased behaviour.

Moreover, the obtained insights can help guide a more targeted analysis and inspect the dataset for the reasons behind the observed bias. In our case, looking at the dataset we see that 96.7% of male, white students passed the bar, while the same percentage for male, black students was 77.8%. Furthermore, the number of white students in the dataset was about 21 times that of black ones. This shows that black, male students are severely underrepresented, while the imbalance between successful/unsuccessful students in the two groups may lead the model to assign significant predictive power to a student's race.

Next, taking into account the generated counterfactuals it is possible to look for imbalances that are not as apparent. To this end, we inspected the dataset for black, male students with $lsat < 19.5$ (based on the counterfactual condition) and $gpa = 3$, only to find out that all such students failed to pass the bar. However, for white, male students, with the same characteristics, half of them passed the bar. On top of this discrepancy, even the specific instances prompted biased behavior, since, for example, a black student with $lsat = 19$, $gpa = 3$, failed, while a white one with $lsat = 17.5$, $gpa = 3$, succeeded, encouraging the model to take racial information into account.

Following this analysis, it should come as no surprise that the RF picked up a corresponding bias, since by looking at the individual DTs we found out that there are 6 different paths that lead to a positive outcome for white, male students with $gpa < 3$, as opposed to only 1 for black students. This means that the RF is more "forgiving" towards white students with low gpa, in contrast to black ones. Targeting these two specific subgroups was guided by the insights obtained by combining infinite counterfactual sets with diversity and inequality constraints, which allowed for generating multiple counterfactuals that led to the discovery of significant information regarding both the dataset and the model.

*Congressional Voting Records* This dataset contains the votes of the U.S. House of Representatives Congressmen on 16 key votes. This time, the problem is to predict whether

**Table 3** Congressional Voting Records dataset instances

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | Outcome |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Factual | + | + | + | − | − | − | + | + | + | − | + | − | − | − | + | + | Democrat |
| Prime implicants | | | ✓ | ✓ | ✓ | | | | | | | | | ✓ | | | Democrat |
| Conditional prime implicants | ✓ | | | ✓ | ✓ | | | | | | ✓ | | | ✓ | | | Democrat |
| Factual | + | + | − | + | + | + | − | − | − | + | − | + | − | + | − | + | Republican |
| Prime implicants | | | | ✓ | ✓ | | | | | | | ✓ | ✓ | ✓ | | | Republican |
| Conditional prime implicants | ✓ | | ✓ | ✓ | ✓ | ✓ | | ✓ | | | | | | ✓ | | | Republican |
| Factual | + | − | + | − | − | − | − | + | + | + | − | − | − | − | + | + | Democrat |
| Prime implicants | | | | ✓ | ✓ | | | | | | | | | ✓ | | | Democrat |
| Conditional prime implicants | ✓ | | ✓ | ✓ | ✓ | | | | | | | | | | | | Democrat |
| Factual | − | − | − | + | + | + | − | − | − | − | − | + | + | + | − | + | Republican |
| Prime implicants | | | | ✓ | ✓ | | | | ✓ | | | | ✓ | ✓ | ✓ | | Republican |
| Conditional prime implicants | ✓ | | ✓ | ✓ | ✓ | | | | | | | | ✓ | ✓ | | | Republican |
| Factual | − | − | + | − | − | + | + | + | + | + | + | − | − | − | + | + | Democrat |
| Prime implicants | | | | ✓ | ✓ | | | | | | | | | ✓ | | | Democrat |
| Conditional prime implicants | ✓ | | ✓ | ✓ | ✓ | | | | | | | | | ✓ | | | Democrat |

a person is a Democrat or a Republican, based on these 16 votes. To this end, we trained a Naive Bayes classifier, however the same analysis can be performed for any BNC. Table 3 shows how 5 particular congressman voted (where + represents voting for, and − voting against). This time, instead of computing counterfactuals, we will present prime implicant explanations to demonstrate the flexibility of the proposed framework. In addition, we should note that of all the works discussed in Sect. 2, only those that are based on tractable architectures (Shih et al., 2018; Shi et al., 2020; Choi et al., 2020) allow for generating both counterfactual and prime implicant explanations, however, they do not support infinite counterfactual sets or any other distance function apart from the Hamming distance. Moreover, as shown in Sect. 8.1, counterfactuals generated by the aforementioned approaches can be retrieved by our framework by adjusting the parameters of the distance function.

Focusing on the first instance, the unconditional prime implicants (shown with ✓) form a set of 4 elements, meaning that as long as the votes regarding topics 3, 4, 5, 14 remain the same, the model will always classify that person as a Democrat. Moreover, to further inspect the model, it is possible to compute conditional prime implicants. For example, requiring that the first vote remains the same, we see that the resulting explanation now has 5 elements, some of them not present in the unconditional explanation. This result indicates there is some relationship among these variables, which could in turn motivate additional analysis. This example shows how prime implicants and counterfactuals can be used to gain complementary insights, identifying conditions that can alter or maintain a model's predictions. This observation highlights the overall strengths of our approach, since it accommodates for generating multiple explanation types, while also combining various useful properties (infinite counterfactuals, diversity, flexible distance functions) to enable an in-depth model inspection. In contrast, most approaches are either tailored to counterfactuals (Verma et al., 2020), support a subset of the aforementioned properties, or are limited by the expressiveness of the distance function.

## 10 Future work and conclusions

In this work we present a framework for generating counterfactual explanations for (ensembles of) multilinear models. This way we extend the methodology in Russell (2019), as well as generalize some of the results in Shih et al. (2018). We show how to apply our results to DTs, RFs, and BNCs, but any multilinear model can be utilized, instead. This is in contrast to methods like, Fernández et al. (2020), since this is based on a modification of the CART algorithm, so it is only applicable to DTs and RFs. Analogously, for BNCs, we show how our framework permits more expressive distance functions, that incorporate the relative importance of each term, instead of treating all feature changes as equally important or feasible. Moreover, we demonstrate how diversity constraints can facilitate inspecting a model for biased behaviour, in cases where unconstrained counterfactuals do not initially reveal such information, guiding the discovery of certain underrepresented groups of the population in the dataset.

In our opinion there are a lot of interesting research directions to go from here. A first remark is that as can be seen from the complexity results, the worst case scenario is exponential, so there are cases where encoding a DP can be impractical. These situations highlight the importance of developing approximate representations of DPs, that correctly classify instances with high probability. This seems like a natural next step, especially considering the long-standing research line of approximate reasoning

in BNs, as well as some recent attempts at approximate reasoning with DTs and RFs (Deng, 2014). An alternative way to address this issue, without resorting to approximations, could be to embed it into optimization frameworks, such as *column generation* (Michele Conforti, 2014), that can effectively handle large problems. Other interesting directions include defining probabilistic versions of DPs, reflecting how probable an assignment is, since currently all assignments are treated as equally probable. Advances in these areas could facilitate generating out-of-the-box counterfactuals, leading to their wider adaptation in practical applications.

**Author Contributions** IP conceived the presented idea, developed the theory and run the experiments. VB supervised the project. Both authors contributed to the final version of the manuscript.

**Funding** Vaishak Belle was partly supported by a Royal Society University Research Fellowship. Ioannis Papantonis was partly supported by the EPSRC grant Towards Explainable and Robust Statistical AI: A Symbolic Approach.

**Availability of data and material** The data that support the findings of this work were derived from the following resources available in the public domain: https://github.com/propublica/compas-analysis/blob/master/compas-scores-two-years.csv (Compass dataset), https://storage.googleapis.com/lawschool_dataset/bar_pass_prediction.csv (Lsat dataset), https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records (Congressional voting records dataset).

## Declarations

**Conflict of interest** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

**Code availability** The code will be made publicly available online, upon acceptance of this work. https://github.com/GiannisPapantonis/DiverseCounterfactuals.

**Consent to participate** Not applicable.

**Consent for publication** Not applicable.

**Ethics approval** Not applicable.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

## References

Belle, V., & Papantonis, I. (2020). Principles and practice of explainable machine learning. CoRR,abs/2009.11698.

Binns, R., Van Kleek, M., Veale, M., Lyngs, U., Zhao, J., & Shadbolt, N. (2018). 'It's reducing a human being to a percentage': Perceptions of justice in algorithmic decisions. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI'18, Association for Computing Machinery*, (pp. 1–14), New York, NY, USA.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer.

Boutilier, C., Friedman, N., Goldszmidt, M., & Koller, D. (1996). Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence, UAI'96*, (pp. 115–123), San Francisco, CA: Morgan Kaufmann Publishers Inc.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123–140.

Bryant, R. E. (1992). Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys, 24*(3), 293–318.

Choi, A., Shih, A., Goyanka, A., & Darwiche, A. (2020). On symbolically encoding the behavior of random forests. CoRR,abs/2007.01493.

Choi, M. J., Tan, V. Y. F., Anandkumar, A., & Willsky, A. S. (2011). Learning latent tree graphical models. *Journal of Machine Learning Research, 12*, 1771–1812.

Conforti, M., Cornuéjols, G. Z. (2014). Integer programming. In *Graduate Texts in Mathematics*, (p. 271). Springer: Cham.

Cui, Z., Chen, W., He, Y., & Chen, Y. (2015). Optimal action extraction for random forests and boosted trees. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (pp. 179–188).

Darwiche, A. (2003). A differential approach to inference in Bayesian networks. *Journal of the ACM, 50*(3), 280–305.

Deng, H. (2014). Interpreting tree ensembles with in trees. arXiv:1408.5456.

Dhar, V. (1998). Data mining in finance: Using counterfactuals to generate knowledge from organizational information systems. *Information Systems, 23*(7), 423–437.

Díaz, Ó., Dalton, J. A., & Giraldo, J. (2019). Artificial intelligence: A novel approach for drug discovery. *Trends in Pharmacological Sciences, 40*(8), 550–551.

Dressel, J. & Farid, H. (2018). The accuracy, fairness, and limits of predicting recidivism. *Science Advances*, *4*(1), eaao5580.

Fatum, R., & Hutchison, M. M. (2010). Evaluating foreign exchange market intervention: Self-selection, counterfactuals and average treatment effects. *Journal of International Money and Finance, 29*(3), 570–584.

Fernández, R. R., Martín de Diego, I., Aceña, V., Fernández-Isabel, A., & Moguerza, J. M. (2020). Random forest explainability using counterfactual sets. *Information Fusion, 63*, 196–207.

Friedman, N., Geiger, D., Goldszmidt, M., Provan, G., Langley, P., & Smyth, P. (1997). Bayesian network classifiers. *Machine Learning, 29*, 131–163.

Gens, R. & Domingos, P. (2012). Discriminative learning of sum-product networks. *Advances in Neural Information Processing Systems*, 25.

Kanamori, K., Takagi, T., Kobayashi, K., & Arimura, H. (2020). Dace: Distribution-aware counterfactual explanation by mixed-integer linear optimization. In *IJCAI*, (pp. 2855–2862).

Kanamori, K., Takagi, T., Kobayashi, K., Ike, Y., Uemura, K., & Arimura, H. (2021). Ordered counterfactual explanation by mixed-integer linear optimization. *In Proceedings of the AAAI Conference on Artificial Intelligence, 35*, 11564–11574.

Karimi, A. -H., Barthe, G., Balle, B., & Valera, I. (2020). Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics*, (pp. 895–905). PMLR.

Lewis, D. (1974). Tijdschrift Voor Filosofie. *Counterfactuals, 36*(3), 602–605.

Lv, J., Dong, B., Lei, H., Shi, G., Wang, H., Zhu, F., Wen, C., Zhang, Q., Fu, L., Gu, X., et al. (2021). Artificial intelligence-assisted auscultation in detecting congenital heart disease. *European Heart Journal-Digital Health, 2*(1), 119–124.

Mishler, A., Kennedy, E. H., & Chouldechova, A. (2021). Fairness in risk assessment instruments: Post-processing to achieve counterfactual equalized odds. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, (pp. 386–400).

Mohammadi, K., Karimi, A.-H., Barthe, G., & Valera, I. (2021). Scaling guarantees for nearest counterfactual explanations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, (pp. 177–187).

Moore D. H. (1987). Classification and regression trees, by leo breiman, jerome h. friedman, richard a. olshen, and charles j. stone. brooks/cole publishing, monterey, 1984, 358 pages, \$27.95. *Cytometry*, *8*(5):534–535.

Mothilal, R. K., Sharma, A., & Tan, C. (2020). Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency. Association for Computing Machinery*.

Pearl, J. (2009). *Causality: Models, Reasoning & Inference* (2nd ed.). Cambridge: Cambridge University Press.

Prosperi, M., Guo, Y., Sperrin, M., Koopman, J. S., Min, J. S., He, X., Rich, S., Wang, M., Buchan, I. E., & Bian, J. (2020). Causal inference and counterfactual prediction in machine learning for actionable healthcare. *Nature Machine Intelligence, 2*(7), 369–375.

Quinlan, J. R. (1987). Simplifying decision trees. *International Journal of Man–Machine Studies, 27*(3), 221–234.

Rooshenas, A. & Lowd, D. (2014). Learning sum-product networks with direct and indirect variable interactions. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, (vol. 32, ICML'14, pp. I-710–I-718). JMLR.org.

Ruben, D. (1990). *Explaining Explanation*. New York: Routledge.

Russell, C. (2019). Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* '19*, (pp. 20–28), New York, NY, USA.

Sampson, R. J., Laub, J. H., & Wimer, C. (2006). Does marriage reduce crime? A counterfactual approach to within-individual causal effects. *Criminology, 44*(3), 465–508.

Shi, W., Shih, A., Darwiche, A., & Choi, A. (2020). On tractable representations of binary neural networks.

Shih, A., Choi, A., & Darwiche, A. (2018). A symbolic approach to explaining Bayesian network classifiers. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization*, (pp. 5103–5111).

Tolomei, G., Silvestri, F., Haines, A., & Lalmas, M. (2017). Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (pp. 465–474).

Vaishya, R., Javaid, M., Khan, I. H., & Haleem, A. (2020). Artificial intelligence (AI) applications for Covid-19 pandemic. *Diabetes & Metabolic Syndrome: Clinical Research & Reviews, 14*(4), 337–339.

VanderWeele, T. J. (2020). Invited commentary: Counterfactuals in social epidemiology-thinking outside of "the box''. *American Journal of Epidemiology, 189*(3), 175–178.

Verma, S., Boonsanong, V., Hoang, M., Hines, K. E., Dickerson, J. P., & Shah, C. (2020). Counterfactual explanations and algorithmic recourses for machine learning: A review. arXiv preprint arXiv:2010.10596.

Verma, S., Hines, K., & Dickerson, J. P. (2022). Amortized generation of sequential algorithmic recourses for black-box models. *In Proceedings of the AAAI Conference on Artificial Intelligence, 36*, 8512–8519.

Wachter, S., Mittelstadt, B., & Russell, C. (2017). *Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR*. European Economics: Microeconomics & Industrial Organization eJournal.

Weissler, E. H., Naumann, T., Andersson, T., Ranganath, R., Elemento, O., Luo, Y., Freitag, D. F., Benoit, J., Hughes, M. C., Khan, F., et al. (2021). The role of machine learning in clinical research: transforming the future of evidence generation. *Trials, 22*(1), 1–15.

Zhao, H., Melibari, M., & Poupart, P. (2015). On the relationship between sum-product networks and Bayesian networks. In *Proceedings of the 32nd International Conference on Machine Learning*.

Zhao, H., Poupart, P., & Gordon, G. J. (2016). A unified approach for learning the parameters of sum-product networks. In *Advances in Neural Information Processing Systems*.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.