Causality-based Counterfactual Explanation for Classification Models

Tri Dung Duong[a], Qian Li[b,*], Guandong Xu[a,*]

[a]*Faculty of Engineering and Information Technology, University of Technology Sydney, NSW, Australia*
[b]*School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University, WA, Australia*

Abstract

Counterfactual explanation is one branch of interpretable machine learning that produces a perturbation sample to change the model's original decision. The generated samples can act as a recommendation for end-users to achieve their desired outputs. Most of the current counterfactual explanation approaches are the gradient-based method, which can only optimize the differentiable loss functions with continuous variables. Accordingly, the gradient-free methods are proposed to handle the categorical variables, which however have several major limitations: 1) causal relationships among features are typically ignored when generating the counterfactuals, possibly resulting in impractical guidelines for decision-makers; 2) the counterfactual explanation algorithm requires a great deal of effort into parameter tuning for dertermining the optimal weight for each loss functions which must be conducted repeatedly for different datasets and settings. In this work, to address the above limitations, we propose a prototype-based counterfactual explanation framework (ProCE). ProCE is capable of preserving the causal relationship underlying the features of the counterfactual data. In addition, we design a novel gradient-free optimization based on the multi-objective genetic algorithm that generates the counterfactual explanations for the mixed-type of continuous and categorical features. Numerical experiments demonstrate

---

*Corresponding author
Email addresses:* `TriDung.Duong@student.uts.edu.au` (Tri Dung Duong),
`qli@curtin.edu.au` (Qian Li), `Guandong.Xu@uts.edu.au` (Guandong Xu)

that our method compares favorably with state-of-the-art methods and therefore is applicable to existing prediction models. All the source codes and data are available at `https://github.com/tridungduong16/multiobj-scm-cf`.

Keywords: counterfactual explanation, interpretable machine learning, structural causal model

---

1. Introduction

Machine learning (ML) is increasingly recognized as an effective approach for large-scale automated decisions in several domains. However, when an ML model is deployed in critical decision-making scenarios such as criminal justice [1, 2] or credit assessment [3], many people are skeptical about its accountability and reliability. Hence, interpretablity is vital to make machine learning models transparent and understandable by humans. Recent years witness an increasing number of studies that have explored ML mechanisms under the causal perspective [4, 5, 6]. Among these studies, counterfactual explanation (CE) is the prominent example-based method that focuses on generating counterfactual samples for interpreting model decisions. For example, consider a customer `A` whose loan application has been rejected by the ML model of a bank. Counterfactual explanations can generate a "what-if" scenario of this person, e.g., "your loan would have been approved if your income was $51,000 more". Namely, the goal of counterfactual explanation is to generate perturbations of an input that leads to a different outcome from the ML model. By allowing users to explore such "what-if" scenarios, counterfactual examples are interpretable and are easily understandable by humans.

Despite recent interests in counterfactual explanations, existing methods suffer three limitations. First, the counterfactual methods neglect the causal relationship among features, leading to the infeasible counterfactual samples for decision makers [7, 8]. In fact, a counterfactual sample is considered as feasible if the changes satisfy conditions restricted by the causal relations. For example, since education causes the choice of the occupation, changing the occupation

2

without changing the education is infeasible for the loan applicant in the real world. Namely, the generated counterfactuals need to preserve the causal relations between features in order to be realistic and actionable. Second, on the algorithm level, most counterfactual methods use the gradient-free optimization algorithm to deal with various data and model types [9, 8, 10, 11, 12]. These gradient-free optimizations rely on the heuristic search, which however suffers from inefficiency due to the large heuristic search space. In addition, optimizing the trade-off among different loss terms in the objective function is difficult, which often leads to sub-optimal counterfactual samples [13, 14, 11].

To address the above limitations, we propose a prototype-based counterfactual explanation framework (ProCE) in this paper. ProCE is a model-agnostic method and is capable of explaining the classification in the mixed feature space. It should be emphasized that the proposed method focuses on maintaining the causal relationships among the features in dataset instead of the causal relationship between features and target variable [15]. Overall, our contributions are summarized as follows:

- By integrate causal discovery framework and causal loss function, our proposed method can produce the counterfactual samples that satisfy the causal constraints among features.

- We utilize the auto-encoder model and class prototype to guide the search progress and speed up the searching speed of counterfactual samples.

- We design a novel multi-objective optimization that can find the optimal trade-off between the objectives while maintaining diversity in counterfactual explanations' feature space.

2. Background

2.1. Preliminary

Throughout the paper, lower-cased letters x and $\boldsymbol{x}$ denote the deterministic scalars and vectors, respectively. We consider a dataset $\mathcal{D} = \{\boldsymbol{x}_i, c_i\}_{i=1}^n$ con-

3

sisting of n instances, where $\boldsymbol{x}_i \in \mathcal{X}$ is a sample, $c_i \in \mathcal{C} = \{0, 1\}$ is the class of individuals $\boldsymbol{x}_i$, and $\boldsymbol{x}_i^j$ is the j-th feature of $\boldsymbol{x}_i$. Also, we consider a classifier $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$ that has the input of feature space $\mathcal{X}$ and the output as $\mathcal{Y} = \{0, 1\}$. We denote $Q_\phi(.)$ as an encoder model parameterized by ϕ. Finally, $\text{proto}_*(\boldsymbol{x})$ and $\mathcal{K}(\boldsymbol{x})$ are the prototype and the set of K-nearest instances of an instance $\boldsymbol{x}$, respectively.

Definition 2.1 (Counterfactual Explanation). *With the original sample $\boldsymbol{x}_{org} \in \mathcal{X}$, and original prediction $y_{org} \in \mathcal{Y}$, the counterfactual explanation aims to find the nearest counterfactual sample $\boldsymbol{x}_{cf}$ such that the outcome of classifier for $\boldsymbol{x}_{cf}$ changes to desired output class y_{cf}. In general, the counterfactual explanation $\boldsymbol{x}_{cf}$ for the individual $\boldsymbol{x}_{org}$ is the solution of the following optimization problem:*

$$\boldsymbol{x}_{cf}^* = \arg\min_{\boldsymbol{x}_{cf} \in \mathcal{X}} f(\boldsymbol{x}_{cf}) \quad \text{subject to} \quad \mathcal{H}(\boldsymbol{x}_{cf}^*) = y_{cf} \tag{1}$$

where $f(\boldsymbol{x}_{\text{cf}})$ is the function measuring the distance between $\boldsymbol{x}_{\text{org}}$ and $\boldsymbol{x}_{\text{cf}}$. Eq (1) demonstrates the optimization objective that minimizes the similarity of the counterfactual and original samples, as well as ensures the classifier to change its decision output. For such explanations to be plausible, they should only suggest small changes in a few features.

To make it clear, we consider a simple scenario that a person with a set of features {income: \$50k, CreditScore: "good", education: "bachelor", age: 52} applies for a loan in a financial organization and receives the reject decision from a predictive model. In this case, the company can utilize the counterfactual explanation (CF) as an advisor that provides constructive advice for this customer. To allow this customer successfully get the loan, CF can give an advice that how to change the customer's profile such as increasing his/her income to \$51k, or enhancing the education degree to "Master". This toy example illustrates that CF is capable of providing interpretable advice that how to makes the least changes for the sample to achieve the desired outcome.

2.2. Related Work

Recently, there has been an increasing number of studies in this field. The existing counterfactual explanation methods can be categorized into gradient-based methods[16, 17, 14], auto-encoder model [18, 13], heuristic search based methods [8, 9] and integer linear optimization [19, 20].

Gradient-based methods: Counterfactual explanation is first proposed by the study [17] as the example-based method to interpret machine learning models' decision. In this study, the authors construct the cross-entropy loss between the desired class and counterfactual samples' prediction with the purpose of changing the model output. Thereafter, some gradient-descent optimization algorithms would be used to minimize the constructed loss. This approach draws much attention with a plethora of studies [11, 18, 14, 14] that aim to customize the loss function to enhance the properties of counterfactual generation. For example, the study [21] extends the distance functions in Eq (1) by using a weight vector (Θ) to emphasize the importance of each feature. Some algorithms such as k-nearest neighbors or global feature evaluation can be deployed to find this vector (Θ). Another framework called DiCE [14] proposes using the diversity score to produce the number of generated samples that allows users to have more options. They thereafter use the weighted sum to combine different loss functions together and also adopt the gradient-descent algorithm to approximately find the optimal solution. The research [22] utilizes the class prototype to guide the search progress to fall into the distribution of the expected class. This method however does not consider the causal relationship among features. The differentiable methods are the prominent approach in counterfactual explanation that allows to optimize easily and control the loss functions, but are only restricted to the differentiable models, and finds it hard to deal with the non-continuous values in tabular data.

Auto-encoder model: Other recent studies based on the variational auto-encoder (VAE) model utilizes the properties of generative models to generate new counterfactual samples. In the study [23], the authors first construct an encoder-decoder architecture. Thereafter, they generate the latent representa-

tion from the encoder, and make some perturbation into the latent representation, and go through the decoder until the prediction models achieve the desired class. Meanwhile, another line of recent work [13] proposes the conditional auto-encoder model by combining different loss functions including prediction loss and proximity loss. They thereafter generate multiple counterfactual samples for all input data points by conditioning on the target class. These studies heavily rely on gradient-descent optimization which can face difficulties when handling categorical features. In addition, VAE models that maximize the lower bound of the log-likelihood rather than measuring the exact log-likelihood can give unstable and inconsistent results.

Heuristic search methods: There is an increasing number of counterfactual explanation methods for non-differentiable models, which makes the previous gradient-based approach not applicable. They utilizes heuristic search for the optimization problem such as Nelder-Mead [11], growing spheres [24], FISTA [10, 22], or genetic algorithms [25, 12, 9]. The main idea of these approaches adopts evolutionary algorithms to effectively finds the optimal counterfactual samples based on the defined cost functions. For example, CERTIFAI [9] customizes the genetic algorithm for the counterfactuals search progress. CERTIFAI adopts the indicator functions (1 for different values, else 0) and mean squared error for categorical and continuous features, respectively. Apart from that, the study [8] introduces a method called FACE that adopts Dijsstra's algorithm to generate counterfactual samples by finding the shortest path of the original input and the existing data points. The main advantage of FACE is that the produced path from Dijsstra's algorithm provides an insight into the step-by-step and feasible actions that users can take to achieve their goals. The generated samples of this method are limited to the input space without generating new data.

Integer linear optimization The studies [7, 19] propose to adopt integer linear optimization (ILO) solver for linear models utilizing linear costs to generate the actionable changes. Specifically, they formulate the problem of finding counterfactual samples according to the cost function as a mixed-integer lin-

ear optimization problem and then utilize some existing solvers [26] to obtain the optimal solution. To speed up the counterfactual samples search process, the study [27] introduces convex constraints to bound the solutions in a region of data space locally. Although these approaches seem promising when dealing with non-continuous features and non-differentiable functions, they can be applied to linear models only.

Our method extends the line of studies [22, 13] by integrating both structural causal model and class prototype. We also formulate the problem as the multi-objective optimization problem and propose an algorithm to find the counterfactual samples effectively.

3. Methodology

In this section, we firstly present different objective functions corresponding to different properties of counterfactual samples. The structural causal model and causal distance are also investigated to exploit the underlying causal relationship among features. Then, we formulate the counterfactual sample generation as a multi-objective optimization problem and propose an algorithm based on the non-dominated sorting genetic algorithm (NSGA-II) to obtain the optimal solutions. Figure 1 generally describes the overall architecture of our proposed framework containing four main different loss functions: 1) prediction loss that ensures the valid counterfactual samples, 2) proximity loss encourages that only small changes would be performed in the counterfactual samples from the original one, 3) prototype-based loss that guides the search progress, and finally 4) causality-preserving loss that maintains the causal relationships. Moreover, there are three models in the framework: provided prediction model (h), auto-encoder model (Q_ϕ), and structural causal model ($\mathcal{M}$).

3.1. Prototype-based Causal Model

Counterfactuals provide these explanations in the form of "how to assign these features with different values, your credit application would have been accepted". This indicates that counterfactual samples should be constrained under
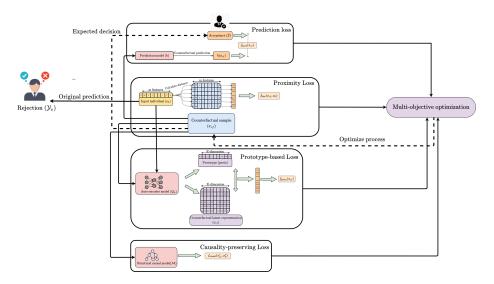
7

Figure 1: The overall framework for the proposed ProCE. The counterfactual samples are first initialized randomly.

several particular conditions. We first provide definitions of each constraint condition and further tie them together as a multi-objective optimization problem to find an optimal counterfactual explanation. For clarity, we first introduce each constrain condition as loss function as follows.

3.1.1. Prediction Loss

We firstly consider the prediction loss which is the prominent loss function for counterfactual explanation. In order to achieve the desired outcome, prediction loss aims to calculate the distance between the counterfactual and expected/desired predictions. This loss function encourages the predictive models to change their predictions of counterfactual samples towards the desired outcomes. Particularly, for the classification scenario, we use the cross-entropy loss to minimize the counterfactual and expected outcome. The prediction loss is defined as follows:

$$f_{\text{pred}}(\boldsymbol{x}_{\text{cf}}) = -y_{\text{cf}}\log(\mathcal{H}(\boldsymbol{x}_{\text{cf}})) - (1 - y_{\text{cf}})\log(1 - \mathcal{H}(\boldsymbol{x}_{\text{cf}})) \tag{2}$$

8

Cross-entropy loss [28] normally measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss is considered in this case to increases as the predicted probability of counterfactual samples $\mathcal{H}(\boldsymbol{x}_{\text{cf}})$ diverges from desired outcome y_{cf}.

3.1.2. Prototype-based Loss

In practice, the search space of counterfactuals might be incredibly large which thus results in slow optimization. Inspired by the work [22], we utilize the class prototype to guide the search progress with the aim of improving the efficiency of finding the counterfactual solutions. Class prototype is first defined as the mean encoding of the instances belonging to the same class [29]. Therefore, in our work, we construct an auto-encoder model to obtain the latent space which allows us to learn a better representation of these instances.

We resort to an encoder function denoted by $Q_\phi : \mathcal{X} \to \mathbb{R}^E$ which projects the input feature $\mathcal{X}$ to the E-dimensional latent space. We denote $\mathcal{K}(\boldsymbol{x}_{\text{org}}) = \{\boldsymbol{x}_k, c_k\}_{k=1}^K$ as a set of K-nearest instances of $\boldsymbol{x}_{\text{org}}$ by estimating the latent distance $||Q_\phi(\boldsymbol{x}_k) - Q_\phi(\boldsymbol{x}_{\text{org}})||_2^2$. Moreover, the classes of these K instances, i.e., $\{c_k\}_{k=1}^K$, are different from the original prediction y_{org} meaning that $c_k \neq y_{\text{org}}$. Formally, $\mathcal{K}(\boldsymbol{x}_{\text{org}})$ is defined as:

$$\mathcal{K}(\boldsymbol{x}_{\text{org}}) = \{\boldsymbol{x}_k, c_k\}_{k=1}^K \subset \mathcal{D} \tag{3}$$

such that

$$\begin{cases} c_k \neq y_{\text{org}} \\ ||Q_\phi(\boldsymbol{x}_r) - Q_\phi(\boldsymbol{x}_{\text{org}})||_2^2 \geq ||Q_\phi(\boldsymbol{x}_j) - Q_\phi(\boldsymbol{x}_{\text{org}})||_2^2 \quad \forall \boldsymbol{x}_r \in \{\mathcal{D} \backslash \mathcal{K}(\boldsymbol{x}_{\text{org}})\} \end{cases} \tag{4}$$

Therefore, a prototype of an original instance $\boldsymbol{x}_{\text{org}}$ is computed by the mean of its nearest neighbors in the latent space:

$$\text{proto}_*(\boldsymbol{x}_{\text{org}}) = \frac{1}{K} \sum_{\boldsymbol{x}_k \in \mathcal{K}(\boldsymbol{x}_{\text{org}})} Q_\phi(\boldsymbol{x}_k) \tag{5}$$

The definition of proto_* in Eq. 5 indicates that the prototype is in fact the representatives of the samples belonging to counterfactual class. We thus define the prototype loss function as L_2-norm distance between the representation of the counterfactual samples $\boldsymbol{x}_{\text{cf}}$ in the latent space and the obtained prototypes:

$$f_{\text{proto}}(\boldsymbol{x}_{\text{cf}}) = \|Q_\phi(\boldsymbol{x}_{\text{cf}}) - \text{proto}_*\|_2^2 \qquad (6)$$

3.1.3. Features cost

One of the main obstacles of generating counterfactual samples is to compute the feature cost which captures the effort required for changing from original instance $\boldsymbol{x}_{\text{org}}$ to counterfactual ones $\boldsymbol{x}_{\text{cf}}$. From the fundamental principles of counterfactual explanation, the generated samples should be as close as to the original one. The smallest changes mean that the least efforts are made for decision-makers to take to achieve their desired goals. However, even experts would find it hard to put the precise cost to demonstrate how unactionable the feature is. Moreover, when it comes to the mixed-type tabular data that contains both the categorical and continuous features, it is challenging to define the distance loss function [30, 31, 32, 33]. The previous studies [9, 14, 25] normally apply the indicator function that returns 1 when two categorical values match and returns 0 otherwise, and adopts L_2-norm distance for comparing continuous features. However, the indicator function which only returns 0 and 1 fails to measure the degree of similarity of two categories. In this study, we use the encoder model Q_ϕ to map the categorical features into the latent space before estimating their distance. The main advantage of this approach is that the encoder model has the capability to capture the underlying relationship and pattern between each categorical value. This means that manual feature engineering such as assigning weight for each category is not necessary, thus saving a great deal of time and effort. Thus, we come up with the distance between two samples is defined as below:

$$f_{\text{dist}}(\boldsymbol{x}_{\text{cf}}, \boldsymbol{x}_{\text{org}}) = \begin{cases} \left\| \boldsymbol{x}_{\text{cf}}^j - \boldsymbol{x}_{\text{org}}^j \right\|_2^2, & \text{if } \boldsymbol{x}^j \text{ is } j\text{-th continuous feature} \\ \left\| Q_\phi(\boldsymbol{x}_{\text{cf}}^j) - Q_\phi(\boldsymbol{x}_{\text{org}}^j) \right\|_2^2, & \text{if } \boldsymbol{x}^j \text{ is } j\text{-th categorical feature} \end{cases} \tag{7}$$

3.1.4. Causality-preserving Loss

Although the distance function in Eq. (7) demonstrates the similarity of two samples, it fails to capture the causal relationship between each feature. To deal with this problem, we integrate the structural causal model, and thus construct the causal loss function to ensure the features' causal relationships in generated samples. We provide some fundamental definitions about causality and thereafter define the corresponding causal loss. In general, a structural causal model $\mathcal{M} = \{\mathbf{U}, \mathbf{V}, \mathbf{F}\}$ [34] consists of three main components defined as below:

- $\mathbf{U}$ is the set of exogenous nodes which has no parents in the causal graph.

- $\mathbf{V}$ is the set of random variables which are endogenous nodes whose causal mechanisms we are modeling. These variables have parents in the causal graph.

- $\mathbf{F}$ is the set of structural causal functions describing the causal relationships among the unobserved and observed variables. Specifically, for each node $\boldsymbol{X} \in V$, a function $f_X \in F$ such that $X = f_X(\text{Pa}(X), \mathbf{U}_X)$ where $\text{Pa}(X)$ is the parent nodes of X.

A causal graph indicates a probabilistic graphical model that represents the assumptions about data-generating mechanism. A causal graph consists of a set of nodes and edges where each node represents a random variable, and each edge illustrates the causal relationship. The causal effect in causal model is facilitated by *do-operator* or intervention [35] that assigns value $\boldsymbol{x}$ to a random variable X denoted by $do(\boldsymbol{x})$. The symbol $do(x)$ is a model manipulation on a causal graph

11

$\mathcal{M}$, which is defined as substitution of causal equation $X = f_X(\text{Pa}(X)_{\mathcal{G}}, \mathbf{U}_X)$ with $X = \boldsymbol{x}$.

For each endogenous node $v \in V$, and its parent nodes $(v_{p_1}, v_{p_2}, \ldots, v_{p_k})$, we estimate each node v as $v = g(v_{p_1}, v_{p_2}, \ldots, v_{p_k})$ to represent their causal relationship with $g(*)$ is the structural causal equation constructed by linear regression model. Since having the full causal graph is often impractical in real-world setting, it is quite challenging to estimate structural causal equation $g(*)$. In this work, we utilize LiNGAM [36] which is a novel estimation technique based on the non-Gaussianity of the data to determine the function $g(*)$. During the counterfactuals generation progress, we firstly produce the predicted value of endogenous node $\boldsymbol{x}^v$ based on their parents before estimating the distance, which is measured as:

$$
\begin{aligned}
f_{\text{causal}}(\boldsymbol{x}_{\text{cf}}^v, \boldsymbol{x}_{\text{org}}^v) &= \left\| \boldsymbol{x}_{\text{cf}}^v - \boldsymbol{x}_{\text{org}}^v \right\|_2^2 \\
&= \left\| g(\boldsymbol{x}_{\text{cf}}^{v_{p_1}}, \boldsymbol{x}_{\text{cf}}^{v_{p_2}}, \ldots, \boldsymbol{x}_{\text{cf}}^{v_{p_k}}) - \boldsymbol{x}_{\text{org}}^v \right\|_2^2
\end{aligned} \tag{8}
$$

With a set of observed variables containing the endogenous and exogenous ones $\boldsymbol{X} = \{\mathbf{U}, \mathbf{V}\}$, we can re-write the general distance between the original and counterfactual sample is the sum of distance of both of them. For the exogenous nodes $\mathbf{U}$ (nodes without any parents in the causal network), we still utilize the Eq. (7) which computingthe distance between two instances, while the causal distance in Eq. (8) is employed for exogenous variables $\mathbf{V}$ (the remaining features).

$$
f_{\text{final\_dist}}(\boldsymbol{x}_{\text{cf}}) = \sum_u^{\mathbf{U}} f_{\text{dist}}(\boldsymbol{x}_{\text{cf}}^u, \boldsymbol{x}_{\text{org}}^u) + \sum_v^{\mathbf{V}} f_{\text{causal}}(\boldsymbol{x}_{\text{cf}}^v, \boldsymbol{x}_{\text{org}}^v) \tag{9}
$$

3.2. Multi-objective Optimization

In this section, we aim to describe the proposed algorithm which is used for optimization process. With the loss functions presented in Sections 3.1 including f_{pred}, f_{proto}, $f_{\text{final\_dist}}$, we come up with the general objective functions Eq (10).

These loss functions illustrates different properties that counterfactual samples should adhere to. The general loss functions containing three different losses is:

$$\mathcal{L}(\boldsymbol{x}_{\text{cf}}) = \{f_{\text{pred}}(\boldsymbol{x}_{\text{cf}}), f_{\text{proto}}(\boldsymbol{x}_{\text{cf}}), f_{\text{final\_dist}}(\boldsymbol{x}_{\text{cf}})\} \tag{10}$$

Therefore, the optimal solutions can be re-written as follows:

$$\boldsymbol{x}_{\text{cf}}^{*} = \underset{\boldsymbol{x}_{\text{cf}} \in \mathcal{X}}{\arg\min} \, \mathcal{L}(\boldsymbol{x}_{\text{cf}}) \tag{11}$$

In order to obtain the optimal solutions, the majority of existing studies [13, 14, 11] uses the trade-off parameter sum assigning each loss function a weight, and combines them together. This approach seems to be reasonable; however, it is very challenging to balance the weights for each loss, resulting in a great deal of efforts and time into hyperparameter tuning. To address this issue, we propose to formulate the counterfactual explanation search as the multi-objective problem (MOP). In this study, we modify the elitist non-dominated sorting genetic algorithm (NSGA-II) [37] to deal with this optimization problem. Its main superiority is to optimize each loss function simultaneously as well as provide the solutions presenting the trade-offs among objective functions. To make it clear, we first present some related definitions. Given a set of n candidate solutions $\mathcal{P} = \{\boldsymbol{x}_i\}_{i=1}^n$, we have the following ones:

Definition 3.1 (Dominance in the objective space). *In the multi-objective optimization problem, the goodness of a solution is evaluated by the dominance[38]. Given two solutions $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ along with a number of p objective functions f_i, we have:*

1. *$\boldsymbol{x}$ weakly dominates $\hat{\boldsymbol{x}}$ ($\boldsymbol{x} \succeq \hat{\boldsymbol{x}}$) iff $f_i(\boldsymbol{x}) \geq f_i(\hat{\boldsymbol{x}}) \; \forall i \in \{1, \ldots, p\}$.*

2. *$\boldsymbol{x}$ dominates $\hat{\boldsymbol{x}}$ ($\boldsymbol{x} \succ \hat{\boldsymbol{x}}$) iff $\boldsymbol{x} \succeq \hat{\boldsymbol{x}}$ and $\boldsymbol{x} \neq \hat{\boldsymbol{x}}$.*

Definition 3.2 (Pareto front). *Pareto front is a set of m solutions denoted by $F_* = \{\boldsymbol{x}_j\}_{j=1}^m \subset \mathcal{P}$ such that $\boldsymbol{x}_j$ dominates all remaining solutions $\boldsymbol{x}_r \in \{\mathcal{P} \backslash F_*\}$ with all objective functions. It means that $f_i(\boldsymbol{x}_j) \geq f_i(\boldsymbol{x}_r) \; \forall i \in \{1, \ldots, p\}$.*

The main goal of non-dominated solutions is to provide a reasonable compromise between all the objective functions that enhance one function's performance but not degrade others.

Definition 3.3 (Non-dominated sorting procedure). *Non-dominated sorting step is mainly used to sort the solutions in population according to the Pareto dominance principle, which plays a central role in the selection procedure. In fact, the set of candidate solutions $\mathcal{P}$ can be divided into a set of H disjoint Pareto front as $\mathcal{F} = \{F_1, F_2, \ldots, F_H\}$ where H is the maximum number of fronts. Non-dominated sorting is a procedure for finding them. Particularly, in the non-dominated sorting step, all the non-dominated solutions from Definition 3.2 are selected from the population and are constructed as the Pareto front F_1. After that, the non-dominated solutions are chosen from the remaining population. The process is repeated until all the solutions are assigned to a front F_H.*

Definition 3.4 (Crowding distance). *One of the vital characteristics of a population solution is diversity. In order to encourage the diversity of candidate solutions, the simplest approach is to choose the individuals having a low density. Particularly, to measure this characteristic, the crowding distance [39, 40] is used to rank each candidate solution. Specficially, the crowding distance of an instance $\boldsymbol{x}$ is calculated as follows:*

$$d(\boldsymbol{x}) = \sqrt{\sum_{i=1}^{p} \left(\frac{f_i(\boldsymbol{x}_a) - f_i(\boldsymbol{x}_b)}{f_i^{min} - f_i^{max}} \right)^2} \tag{12}$$

where p is the number of objective functions, $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ are two nearest instances of $\boldsymbol{x}$ by calculating the Euclidean distance, f_i is the i-th objective function, f_i^{min} and f_i^{max} are its minimum or maximum value, respectively. The fundamental concept behind crowding distance is to compute the Euclidean distance between each candidate solution $\{\boldsymbol{x}_j\}_{j=1}^{m}$ in a front F_* by using p objective functions corresponding to p-dimensional hyper space.

The optimization process for objective function (10) is given by Algorithm 1. The main idea behinds our approach is that for each generation, the algorithm chooses the Pareto Front for each objective function and evolves to the better ones. We firstly find the nearest class prototype of the original sample $\boldsymbol{x}_{\mathrm{org}}$, which is used to measure the prototype loss function later. For the optimal counterfactual $\boldsymbol{x}_{\mathrm{cf}}^*$ finding progress, each candidate solution is represented by the D-dimensional feature as the genes. A random candidate population is initialized with the Gaussian distribution. Thereafter, the objective functions including f_{pred}, f_{proto}, $f_{\mathrm{final\_dist}}$ are calculated for each candidate. Non-dominated sorting procedure illustrated in Definition 3.3 is then performed to obtain a set of Pareto fronts $\mathcal{F} = \{F_i\}_{h=1}^{H}$.

The crowding distance function illustrated in Definition 3.4 and Eq. (12) then is adopted as the score to assign to each individual in the current population. The algorithm only keeps the candidate solutions having the greatest ranking score, which illustrates that these solutions have low density. The cross-over and mutation procedures [41] are finally performed to generate the next population. Particularly, the cross-over of two parents generates the new candidate solutions by randomly swapping parts of genes. Meanwhile, the mutation procedure randomly alters some genes in the candidate solutions to encourage diversity and avoid local minimums. We repeat this process through many generations to find the optimal counterfactual solution.

Algorithm 1 Multi-objective Optimization for Prototype-based Counterfactual Explanation (ProCE)

Input: An original sample $\boldsymbol{x}_{\text{org}}$ with its prediction y_{org}, desired class y_{cf}, a provided machine learning classifier $\mathcal{H}$ and encoder model Q_ϕ.

1: Compute prototype proto_* by Eq. (5).
2: Initialize a batch of initial population with n candidate solutions $\mathcal{P} = \{\boldsymbol{\Delta}_i\}_{i=1}^n$ with $\boldsymbol{\Delta}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\nu})$.
3: $\mathcal{Q} = \emptyset$
4: **for** $g = 1$ to G generation **do**
5: $\quad$ $\mathcal{P} = \mathcal{P} \cup \mathcal{Q}$
6: $\quad$ **for** each candidate solution $\boldsymbol{\Delta}_i$ in $\mathcal{P}$ **do**
7: $\quad\quad$ Compute $f_{\text{pred}}(\boldsymbol{\Delta}_i)$ based on Eq. (2).
8: $\quad\quad$ Use proto_* to compute $f_{\text{proto}}(\boldsymbol{\Delta}_i)$ based on Eq. (6).
9: $\quad\quad$ Compute $f_{\text{final\_dist}}(\boldsymbol{\Delta}_i)$ based on Eq. (9).
10: $\quad$ **end for**
11: $\quad$ Obtain $\mathcal{F} = \{F_h\}_{h=1}^H$ by using non-dominated sorting procedure in Definition 3.3.
12: $\quad$ $\mathcal{P} = \emptyset$
13: $\quad$ $h = 0$
14: $\quad$ **while** $|\mathcal{P}| + |F_h| < n$ **do**
15: $\quad\quad$ $\mathcal{P} = \mathcal{P} \cup F_h$
16: $\quad\quad$ $h = h + 1$
17: $\quad$ **end while**
18: $\quad$ Compute the crowding distance as the ranking score for each solution in $\mathcal{P}$ based on Eq. (12).
19: $\quad$ Keep n individuals in $\mathcal{P}$ based on ranking score.
20: $\quad$ Randomly pair $\lceil n/2 \rceil$ $\{\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2\} \in \mathcal{P}$
21: $\quad$ **for** each pair $\{\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2\}$ **do**
22: $\quad\quad$ Perform crossover$(\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2) \rightarrow \boldsymbol{\Delta}_1', \boldsymbol{\Delta}_2'$
23: $\quad\quad$ Perform mutation $\boldsymbol{\Delta}_1' \rightarrow \tilde{\boldsymbol{\Delta}}_1, \boldsymbol{\Delta}_2' \rightarrow \tilde{\boldsymbol{\Delta}}_2$
24: $\quad\quad$ $\mathcal{Q} = \mathcal{Q} \cup \{\tilde{\boldsymbol{\Delta}}_1, \tilde{\boldsymbol{\Delta}}_2\}$
25: $\quad$ **end for**
26: **end for**
27: $\boldsymbol{\Delta}^* \leftarrow \mathcal{P}[0]$
Output: $\boldsymbol{x}_{\text{cf}} = \boldsymbol{\Delta}^*$

---

4. Experiments

We conduct experiments on four datasets to demonstrate the superior performance of our method when compared with state-of-the-art methods. All implementations are conducted in Python 3.7.7 with 64-bit Red Hat, Intel(R) Xeon(R) Gold 6150 CPU @ 2.70GHz. For our method, we construct the multi-objective optimization algorithm with the support of library Pymoo[1] [42]. More details of implementation settings can be found in our code repository.

4.1. Datasets

This section provides information about the datasets, on which we perform the comparison experiments. Our method is capable of generating counterfactual samples while maintaining the causal relationship. To validate this claim, we consider some feature conditions that restrict the generated counterfactual samples for each dataset. For simplicity, we denote $a \propto b$ for the condition that (a increase $\Rightarrow b$ increase) AND (a decrease $\Rightarrow b$ decrease). We use four datasets including `Simple-BN`, `Sangiovese`, `Adult` and `Law`.

`Simple-BN` [13] is a synthetic dataset containing 10,000 records with three features (a_1,a_2,a_3) and a binary output (y). The data is generated based on the followed causal mechanism:

$$a_1 \sim \mathcal{N}(\mu_1, \sigma_1)$$
$$a_2 \sim \mathcal{N}(\mu_2, \sigma_2)$$
$$a_3|a_1, a_2 \sim \mathcal{N}(k_3 * (a_1 + a_2)^2 + b_3, \sigma_3)$$
$$y|a_1, a_2, a_3 \sim \text{Ber}(\sigma(k_y * (a_1 * a_2) + b_y - a_3)) \tag{13}$$

As illustrated by structural causal equations in Eq (13), two random variables a_1 and a_2 follow the corresponding normal distribution $\mathcal{N}(\mu_1, \sigma_1)$ and $\mathcal{N}(\mu_2, \sigma_2)$, while a_3 follows the normal distribution with mean value determined by the function of a_1 and a_2. Additionally, target variable y follows the Bernoulli distribution with the function of a_1, a_2 and a_3. Based on these generating

---

[1]https://pymoo.org/algorithms/nsga2.html

17

mechanism, we consider the following causal relationship between a_1, a_2 and a_3:

$$(a_1, a_2) \propto a_3 \qquad (14)$$

The condition in Eq (13) means that a_3 monotonically increase and decrease by a function of two random variables a_1 and a_2.

Sangiovese[2][43] dataset evaluates the impact of several agronomic settings on the quality of the Tuscan grapes. This dataset provides information about 14 continuous features along with the binary output. We consider the task of determining whether the grapes' quality is good or not. Based on the conditional linear Bayesian network provided with the dataset, we consider a causal relationship between two features including mean number of sprouts (SproutN) and mean number of bunches (BunchN) that is:

$$\text{BunchN} \propto \text{SproutN} \qquad (15)$$

Adult[3][44] is the real-world dataset providing information of loan applicants in the financial organization. It is a mixed-type dataset that consists of instances having both continuous features and categorical features. For this dataset, we consider the task of determining whether the annual income of a person exceeds \$50k dollars. Similar to the study [13], with $\boldsymbol{x}_*^{\text{age}}$ and $\boldsymbol{x}_*^{\text{education}}$ referring to the feature age and education of an individual, we consider two conditions as below:

$$\boldsymbol{x}_{\text{cf}}^{\text{age}} \geq \boldsymbol{x}_{\text{org}}^{\text{age}} \qquad (16)$$

$$\boldsymbol{x}_{\text{cf}}^{\text{education}} \propto \boldsymbol{x}_{\text{org}}^{\text{age}} \qquad (17)$$

Regarding the first condition ($\boldsymbol{x}_{\text{cf}}^{\text{age}} \geq \boldsymbol{x}_{\text{org}}^{\text{age}}$), counterfactual algorithms should not suggest decreasing individuals' ages since it violates the natural constraint that human age increases over time. Meanwhile, the second condition ($\boldsymbol{x}_{\text{cf}}^{\text{education}} \propto$

---

[2]https://www.bnlearn.com/bnrepository/clgaussian-small.html
[3]https://archive.ics.uci.edu/ml/datasets/adult

$\boldsymbol{x}_{\text{org}}^{\text{age}}$) demonstrates the education-age causal relationship that obtaining a higher degree of education such as from "Bachelor" to "PhD" requires years to complete, thus causing age to increase. As a result, any counterfactual sample increasing education-level without increasing age is infeasible.

Law[4][45] dataset provides information of students with their features: sex, race and their entrance exam scores (LSAT), grade-point average (GPA) and first year average grade (FYA). The main task is to determine which applicants will be accepted to the law program. We consider a causal relationship:

$$(\text{LSAT}, \text{GPA}) \propto \text{FYA} \tag{18}$$

In order to evaluate the models' effectiveness, we randomly split each dataset into 80% training and 20% test set. We conduct 100 repeated experiments, then evaluate performance on the test set and finally report the average statistics.

4.2. Evaluation Metrics

In this section, we briefly describe six quantitative metrics that are used to evaluate the performance of our proposed method and baselines. We sample a number of n factual samples and generate the counterfactual samples for them. Meanwhile n_{cat} and n_{con} are the corresponding number of categorical and continuous features. $\mathbb{1}(.)$ is the indicator function that returns 1 when the conditions are satisfied, otherwise returns 0.

Target-class validity (%Tcv) [13, 8] evaluates how well the algorithm can produce valid samples. Particularly, %Tcv is calculated as the ratio of the number of samples belonging to the desired class and the number of factual samples. Higher target-class validity is favorable, demonstrating that the algorithm can generate greater numbers of counterfactual samples towards the desirable target variable.

---

[4]http://www.seaphe.org/databases.php

$$\%\mathrm{Tcv} = \sum_{i=0}^{n} \frac{\mathbb{1}(h(\boldsymbol{x}_{\mathrm{cf}}) = y_{\mathrm{cf}})}{n} \tag{19}$$

Causal-constraint validity (%Ccv) measures the percentage of counterfactual samples satisfying the pre-defined causal conditions. With this metric, the main aim is to evaluate how well our algorithm can generate feasible counterfactual samples that do not violate the causal relationship among features [13]. With the causal conditions defined in the Section 4.1, using n_s as the number samples satisfying causal conditions, the causal-constraint validity is defined in Eq (20). Higher causal-constraint validity is preferable, illustrating the greater number of satisfied counterfactual samples.

$$\%\mathrm{Ccv} = \frac{n_s}{n} \tag{20}$$

Categorical proximity measures the proximity for categorical features representing the total number of matches on the values of each category between $\boldsymbol{x}_{\mathrm{cf}}$ and $\boldsymbol{x}_{\mathrm{org}}$. Higher categorical proximity is better, implying that the counterfactual sample preserves the minimal changes from the original [14].

$$\mathrm{Cat\_proximity} = 1 - \sum_{i=0}^{n} \sum_{j=0}^{n_{cat}} \mathbb{1}(\boldsymbol{x}_{\mathrm{cf}}^{j} \neq \boldsymbol{x}_{\mathrm{org}}^{j}) \tag{21}$$

Continuous proximity illustrates the proximity of the continuous features, which is calculated as the negative of L_2-norm distance between the continuous features in $\boldsymbol{x}_{\mathrm{cf}}$ and $\boldsymbol{x}_{\mathrm{org}}$. Higher continuous proximity is preferable, implying that the distance between the continuous features of $\boldsymbol{x}_{\mathrm{org}}$ and $\boldsymbol{x}_{\mathrm{cf}}$ should be as small as possible [14].

$$\mathrm{Con\_proximity} = - \sum_{i=0}^{n} \sum_{j=0}^{n_{con}} \left\| \boldsymbol{x}_{cf}^{j} - \boldsymbol{x}_{0}^{j} \right\|_{2}^{2} \tag{22}$$

IM1 and IM2 are two interpretability metrics (IM) proposed in [22]. Let Q_{ϕ}^{org}, Q_{ϕ}^{cf} and Q_{ϕ}^{full} be the auto-encoder models trained specifically on samples of class y_{org}, samples of class y_{cf} and the full dataset, respectively, we first provide the general idea behind these two metrics. On the one hand, **IM1** measures the

ratio of reconstruction errors of counterfactual sample $\boldsymbol{x}_{\mathrm{cf}}$ using Q_ϕ^{cf} and Q_ϕ^{org}. A smaller value for **IM1** indicates that $\boldsymbol{x}_{\mathrm{cf}}$ can be reconstructed more accurately by the autoencoder trained only on instances of the counterfactual class y_{cf} than by the autoencoder trained on the original class y_{org}. This therefore demonstrate that the counterfactual sample $\boldsymbol{x}_{\mathrm{cf}}$ lies closer to the data manifold of counterfactual class y_{cf}, which is considered to be more interpretable. On the other hand, **IM2** evaluates the similarity of counterfactual sample $\boldsymbol{x}_{\mathrm{cf}}$ produced by Q_ϕ^{cf} and Q_ϕ. A low value of IM2 means that the reconstructed instances of $\boldsymbol{x}_{\mathrm{cf}}$ are very similar when using either Q_ϕ^{cf} or Q_ϕ^{full}. Therefore, the data distribution of the counterfactual class y_{cf} describes x_{cf} as close as the distribution of all classes. Particularly, **IM1** and **IM2** are defined as follows:

$$\mathrm{IM1}(Q_\phi^{\mathrm{cf}}, Q_\phi^{\mathrm{org}}, \boldsymbol{x}_{\mathrm{cf}}) = \sum_{i=0}^{\mathrm{n}} \frac{\left\| \boldsymbol{x}_{\mathrm{cf}} - Q_\phi^{\mathrm{cf}}(\boldsymbol{x}_{\mathrm{cf}}) \right\|_2^2}{\left\| \boldsymbol{x}_{\mathrm{cf}} - Q_\phi^{\mathrm{org}}(\boldsymbol{x}_{\mathrm{cf}}) \right\|_2^2 + \epsilon} \tag{23}$$

$$\mathrm{IM2}(Q_\phi^{\mathrm{cf}}, Q_\phi^{\mathrm{full}}, \boldsymbol{x}_{\mathrm{cf}}) = \sum_{i=0}^{\mathrm{n}} \frac{\left\| Q_\phi^{\mathrm{cf}}(\boldsymbol{x}_{\mathrm{cf}}) - Q_\phi^{\mathrm{full}}(\boldsymbol{x}_{\mathrm{cf}}) \right\| r_2}{\| \boldsymbol{x}_{\mathrm{cf}} \|_2^2 + \epsilon} \tag{24}$$

4.3. Baseline Methods

We compare our proposed method (ProCE) with several baselines including Wachter (AR), Growing Sphere (GS), CERTIFAI, CCHVAE and FACE. All of them are the recent approaches in the counterfactual explanation with available source codes and framework. The brief description of these baselines are illustrated as follows:

1. **Wachter (Wach)** [17] which is a fundamental approach that generates counterfactual explanations by minimizing L_1-norm by using gradient descent to find counterfactuals x_{cf} as close as to original instance x_{org}.

2. **Growing Sphere (GS)** [46] is a random search algorithm, which generates samples around the factual input point until a point with a corresponding counterfactual class label was found. Growing hyperspheres are utilized to create the random samples around the original instance.

This approach deals with immutable features by excluding them from the search procedure.

3. **CERTIFAI** [47] CERTIFAI is an approach that utilizes genetic algorithm to finds the counterfactual samples more effectively. The source code for this method is not avaibale; therefore, we implement the CERTIFAI with the support from Python library PyGAD[5].

4. **DiCE** [14]. DiCE is one of the most prominent counterfactual explanation framework. This construct the weighted sum of different loss functions including proximity, diversity and sparsity together, and optimize the combined loss via the gradient-descent algorithm. For implementation, we utilize the source code[6] with default settings.

5. **FACE** [8] produces a feasible and actionable set of counterfactual actions based on the shortest path lengths as determined by density-weighted metrics. The generated counterfactuals by this method that are plausible and coherent with the underlying data distribution.

For all the experiments, we build two predictions model namely 1st **classifier** and 2nd **classifier**. The first classifier is a neural network with three hidden layers, while the second one has five hidden layers with the following architecture:

1st **classifier**

- hidden Layer 1(Number of features, 64), batch normalization layer, dropout(0.1), activation function ReLU

- hidden Layer 2(64, 32), batch normalization layer, Dropout(0.1), activation function ReLU

- hidden Layer 3(32, 16), batch normalization layer, Dropout(0.1), activation function ReLU

---

[5]`https://github.com/ahmedfgad/GeneticAlgorithmPython`
[6]`https://github.com/divyat09/cf-feasibility`

- last Layer (16, Data size), activation function sigmoid

2nd classifier

- hidden layer 1(Number of features, 256), batch normalization layer, Dropout(0.1), activation function ReLU

- hidden layer 2(Number of features, 128), batch normalization layer, Dropout(0.1), activation function ReLU

- hidden layer 3(Number of features, 64), batch normalization layer, Dropout(0.1), activation function ReLU

- hidden layer 4(64, 32), batch normalization layer, Dropout(0.1), activation function ReLU

- hidden layer 5(32, 16), batch normalization layer, Dropout(0.1), activation function ReLU

- last hidden layer (16, Data size), activation function sigmoid

The continuous features in datasets are in different value ranges; therefore, following the common practice in feature engineering [48, 49, 50], we normalize the continuous feature to range (0,1). Moreover, regarding the categorical features, we transform them into numeric forms by using a label encoder.

4.4. Results and Discussions

The performance of different metrics on 1st and 2nd classifier are illustrated in Table 1 and 2, respectively. Regarding to the 1st classifier from Table 1, all three methods achieve the competitive target-class validity, except the Watch performance in all datasets with around 90% of samples belonging to the target class. Regarding the percentage of samples satisfying the causal constraints, by far the greatest performance is achieved by ProCE with 85.91%, 91.84%, 95.64% and 90.43% for `Simple-BN`, `Sangiovese`, `Adult` and `Law` datasets, respectively. FACE also produces a competitive performance across four datasets in terms of this metric, standing at 81.49%, 88.65%, 92.49% and 86.71% while the majority

of generated samples from Watch violate the causal constraints (63.61%, 58.1%, 70.40% and 76.71%). The performance of %Ccv cannot be achieved to 100% for all the methods which demonstrates that it is quite challenging to maintain the causal constraints in counterfactual samples. Moreover, these results indicate that by integrating the structural causal model, our proposed method can effectively produce the counterfactual samples preserving the features' causal relationships. Regarding interpretability scores, our proposed method achieved the best IM1 and IM2 on four datasets. DiCE is ranked second recorded with competitive result in `Adult` dataset (0.0809 for IM1 and 0.2679 for IM2) and `Law` dataset (0.0423 for IM1 and 0.0427 for IM2). The performance of all metrics on the 2nd classifier in Table 2 also demonstrates the competitive performance of our proposed method across all metrics. We also notice that although the 2nd has a more complicated architecture than the 1st classifier, there is a small variation on the performance of counterfactual explanation algorithm. Finally, as expected, by using prototype as a guideline of the counterfactual search process, ProCE produces more interpretable counterfactual instances recorded with good performance in IM1 and IM2. By contrast, it is challenging for other approaches to reconstruct the counterfactual samples, leading to high interpretability scores (IM1 and IM2).

On the other hand, to better comprehend the effectiveness of our proposed method in producing counterfactual samples compared with other approaches, we also perform a statistical significance test (paired t-test) between our approach (ProCE) and other methods on each dataset and each metric with the obtained results on 100 randomly repeated experiments and report the result of p-value in Table 1 and 2. We find that our model is statistically significant with $p < 0.05$, thus demonstrating the effectiveness of ProCE in counterfactual samples generation task.

Figure 2 provides information about the categorical proximity in the Adult dataset and continuous proximity in four datasets. For the categorical proximity on both 1st and 2nd classifier, ProCE consistently achieves an average of 5 out of the total 6 categories in the dataset meaning that the counterfactual sample

Method	Dataset	Performance				p-value			
		%Tcv	%Ccv	IM1	IM2 (x10)	%Tcv	%Ccv	IM1	IM2
Wach	Simple-BN	91.00	63.61	0.0379 ± 0.0741	0.0769 ± 0.1385	0.0129	0.0289	0.0393	0.0446
GS	Simple-BN	100.00	79.72	0.0453 ± 0.0835	0.0792 ± 0.0202	0.0340	0.0480	0.0223	0.0483
CERTIFAI	Simple-BN	100.00	77.44	0.0489 ± 0.1353	0.0271 ± 0.0711	0.0098	0.0226	0.0365	0.0218
DiCE	Simple-BN	100.00	73.61	0.0376 ± 0.1345	0.0815 ± 0.1762	0.0227	0.031	0.0135	0.0427
FACE	Simple-BN	100.00	81.49	0.0365 ± 0.0583	0.0429 ± 0.1614	0.0256	0.0197	0.0444	0.0468
ProCE	Simple-BN	**100.00**	**85.91**	$\mathbf{0.0322 \pm 0.1014}$	$\mathbf{0.0211 \pm 0.0845}$	-	-	-	-
Wach	Sangiovese	92.03	58.10	0.2513 ± 0.1452	0.0533 ± 0.0132	0.0260	0.0365	0.0447	0.0358
GS	Sangiovese	100.00	89.60	0.2295 ± 0.0584	0.0425 ± 0.1502	0.0131	0.0469	0.014	0.0162
CERTIFAI	Sangiovese	100.00	74.29	0.2915 ± 0.1920	0.0721 ± 0.1366	0.0410	0.0389	0.0215	0.0212
DiCE	Sangiovese	100.00	78.10	0.2447 ± 0.0759	0.0374 ± 0.1657	0.0297	0.0306	0.0388	0.0102
FACE	Sangiovese	100.00	88.65	0.2424 ± 0.0962	0.0873 ± 0.0495	0.0471	0.0148	0.0140	0.0119
ProCE	Sangiovese	**100.00**	**91.84**	$\mathbf{0.2152 \pm 0.1686}$	$\mathbf{0.0370 \pm 0.0574}$	-	-	-	-
Wach	Adult	93.95	70.40	0.0709 ± 0.1582	0.3063 ± 0.1382	0.048	0.0285	0.0242	0.0407
GS	Adult	100.00	70.13	0.2241 ± 0.0396	0.3343 ± 0.0564	0.0144	0.0274	0.0114	0.0468
CERTIFAI	Adult	100.00	91.99	0.0939 ± 0.0834	0.3735 ± 0.1150	0.0320	0.0348	0.0310	0.0222
DiCE	Adult	100.00	80.40	0.0809 ± 0.1538	0.2679 ± 0.1661	0.0318	0.0169	0.0275	0.0415
FACE	Adult	100.00	92.49	0.1283 ± 0.0336	0.3245 ± 0.1881	0.0215	0.0346	0.019	0.0242
ProCE	Adult	**100.00**	**95.64**	$\mathbf{0.0675 \pm 0.1908}$	$\mathbf{0.2171 \pm 0.0546}$	-	-	-	-
Wach	Law	92.45	76.71	0.0536 ± 0.1312	0.0470 ± 0.0800	0.0159	0.026	0.0115	0.0378
GS	Law	100.00	86.23	0.0484 ± 0.1173	0.0487 ± 0.0858	0.0481	0.0392	0.0314	0.0315
CERTIFAI	Law	100.00	82.72	0.0567 ± 0.1427	0.0461 ± 0.1797	0.0102	0.0425	0.0191	0.0340
DiCE	Law	100.00	85.75	0.0423 ± 0.1902	0.0427 ± 0.0801	0.0138	0.0206	0.0122	0.0487
FACE	Law	100.00	86.71	0.0418 ± 0.0125	0.0435 ± 0.1160	0.0125	0.0315	0.0333	0.0450
ProCE	Law	**100.00**	**90.43**	$\mathbf{0.0410 \pm 0.1268}$	$\mathbf{0.0421 \pm 0.1907}$	-	-	-	-

Table 1: Performance of all methods on 1st classifier. We compute p-value by conducting a paired t-test between our approach (ProCE) and baselines with 100 repeated experiments for each metric.

generated from ProCE preserves an average of 5 categorical features from the original instances. CERTIFAI and FACE also yield competitive results for categorical proximity, whereas the lowest result is recorded in the GS algorithm (1.7 to 3.5 categories). These results illustrate that the gradient-free based approach including ProCE, CERTIFAI and FACE can achieve better performance in handling the non-continuous features in tabular data. When it comes to the continuous proximity, ProCE produces the counterfactual sample with the greatest similarity around over -0.02, -0.078, -0.1875 and -0.17 corresponding to Simple-BN, Sangiovese, Adult and Law dataset. Our proposed method pro-

Method	Dataset	Performance				p-value			
		%Tcv	%Ccv	IM1	IM2 (x10)	%Tcv	%Ccv	IM1	IM2
Wach	Simple-BN	93.33	70.96	0.0512 ± 0.0466	0.0262 ± 0.0507	0.0320	0.0096	0.0372	0.0487
GS	Simple-BN	100.00	79.46	0.0401 ± 0.1888	0.0354 ± 0.0352	0.0242	0.038	0.0274	0.0308
CERTIFAI	Simple-BN	100.00	83.68	0.0465 ± 0.0389	0.0824 ± 0.1345	0.0378	0.0138	0.031	0.0255
DiCE	Simple-BN	100.00	82.93	0.0342 ± 0.0790	0.0448 ± 0.0260	0.0376	0.0324	0.0497	0.0277
FACE	Simple-BN	100.00	82.03	0.0458 ± 0.1209	0.0435 ± 0.0123	0.0215	0.0086	0.0275	0.0437
ProCE	Simple-BN	**100.00**	**89.09**	**0.0318 ± 0.0104**	**0.0202 ± 0.0167**	-	-	-	-
Wach	Sangiovese	93.92	74.49	0.2731 ± 0.1090	0.0445 ± 0.0919	0.0255	0.0291	0.0474	0.0363
GS	Sangiovese	100.00	71.44	0.2654 ± 0.0394	0.0407 ± 0.0770	0.0319	0.0378	0.0294	0.0447
CERTIFAI	Sangiovese	100.00	80.95	0.2583 ± 0.1369	0.0798 ± 0.1898	0.0281	0.0304	0.0389	0.0297
DiCE	Sangiovese	100.00	92.25	0.2603 ± 0.1383	0.0880 ± 0.1144	0.0436	0.0323	0.0478	0.0381
FACE	Sangiovese	100.00	77.95	0.2302 ± 0.0029	0.0522 ± 0.0169	0.0464	0.0152	0.0351	0.0184
ProCE	Sangiovese	**100.00**	**86.25**	**0.2127 ± 0.0973**	**0.0360 ± 0.0388**	-	-	-	-
Wach	Adult	91.45	75.23	0.1731 ± 0.1270	0.3520 ± 0.1592	0.0127	0.0454	0.0407	0.0378
GS	Adult	100.00	75.82	0.1719 ± 0.1673	0.1565 ± 0.1634	0.0308	0.0099	0.0224	0.0447
CERTIFAI	Adult	100.00	80.56	0.1512 ± 0.0920	0.2326 ± 0.0686	0.0265	0.0351	0.0309	0.0341
DiCE	Adult	100.00	76.43	0.2371 ± 0.1801	0.3823 ± 0.0016	0.0154	0.0396	0.0427	0.0343
FACE	Adult	100.00	76.02	0.1649 ± 0.1448	0.3393 ± 0.0083	0.0254	0.0144	0.0105	0.0285
ProCE	Adult	**100.00**	**92.85**	**0.1467 ± 0.1096**	**0.1324 ± 0.1027**	-	-	-	-
Wach	Law	90.55	73.36	0.0437 ± 0.0913	0.0594 ± 0.1896	0.0375	0.0474	0.0462	0.0349
GS	Law	100.00	84.09	0.0532 ± 0.0988	0.0643 ± 0.0244	0.0269	0.0267	0.0402	0.0334
CERTIFAI	Law	100.00	80.88	0.0382 ± 0.0915	0.0592 ± 0.0566	0.0495	0.0172	0.0428	0.0286
DiCE	Law	100.00	87.54	0.0382 ± 0.0530	0.0461 ± 0.1928	0.0421	0.0489	0.0342	0.0373
FACE	Law	100.00	75.51	0.0422 ± 0.1875	0.0383 ± 0.0029	0.0476	0.0374	0.015	0.0304
ProCE	Law	**100.00**	**79.48**	**0.0317 ± 0.1073**	**0.0313 ± 0.1648**	-	-	-	-

Table 2: Performance of all methods on 2nd classifier. We compute p-value by conducting a paired t-test between our approach (ProCE) and baselines with 100 repeated experiments for each metric.

duces the least fluctuation in continuous proximity for `Sangiovese`, `Simple-BN`, `Adult`, while the biggest variation is witnessed in `Law`.

We also report the running time of different methods in Table 3. Overall, the shortest time is recorded with Watch method on `Simple-BN`, `Sangiovese`, and `Law` datasets. The possible reason is that Watch is the naive approach which optimizes the basic proximity loss functions using gradient descent. This therefore allows producing the counterfactual sample in a prominent time but demonstrates a poor performance in several metrics. Our approach (ProCE) also demonstrates competitive time performance on these three datasets. Regarding

Method	Dataset	Running time (s)	
		1st classifier	2nd classifier
Wach	Simple-BN	**3.030 ± 0.105**	**5.111 ± 0.135**
GS	Simple-BN	7.126 ± 0.153	6.541 ± 0.053
CERTIFAI	Simple-BN	6.213 ± 0.007	6.237 ± 0.088
DiCE	Simple-BN	6.522 ± 0.088	6.455 ± 0.016
FACE	Simple-BN	8.022 ± 0.014	6.599 ± 0.173
ProCE	Simple-BN	4.085 ± 0.055	6.017 ± 0.160
Wach	Sangiovese	**5.125 ± 0.097**	**5.768 ± 0.113**
GS	Sangiovese	8.048 ± 0.176	12.549 ± 0.086
CERTIFAI	Sangiovese	7.688 ± 0.131	8.906 ± 0.105
DiCE	Sangiovese	13.426 ± 0.158	11.775 ± 0.086
FACE	Sangiovese	7.810 ± 0.076	11.348 ± 0.200
ProCE	Sangiovese	6.809 ± 0.162	7.304 ± 0.101
Wach	Adult	7.046 ± 0.151	7.260 ± 0.058
GS	Adult	6.472 ± 0.021	6.464 ± 0.145
CERTIFAI	Adult	13.851 ± 0.001	9.457 ± 0.120
DiCE	Adult	7.943 ± 0.046	10.326 ± 0.016
FACE	Adult	10.821 ± 0.162	9.140 ± 0.149
ProCE	Adult	**4.837 ± 0.026**	**5.733 ± 0.019**
Wach	Law	**4.821 ± 0.068**	**4.957 ± 0.131**
GS	Law	12.126 ± 0.093	13.480 ± 0.152
CERTIFAI	Law	5.516 ± 0.009	6.337 ± 0.027
DiCE	Law	6.150 ± 0.038	8.103 ± 0.0410
FACE	Law	5.450 ± 0.184	6.661 ± 0.025
ProCE	Law	4.830 ± 0.130	5.001 ± 0.152

Table 3: We report running time of different methods on four datasets.

Adult dataset which contains both categorical and continuous features, our approach performs counterfactual sample generation in the outstanding time and also surpasses other non-gradient descent methods such as FACE, CERTIFAI and GS.

Figure 3a and 3b show the variation of our method's performance with the different numbers of K-nearest neighbors for class prototype and E-dimensional embedding sizes of the auto-encoder model, respectively. It is clear from Fig-
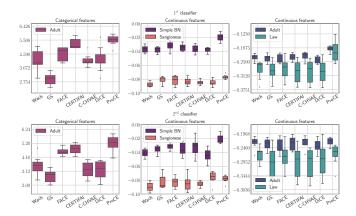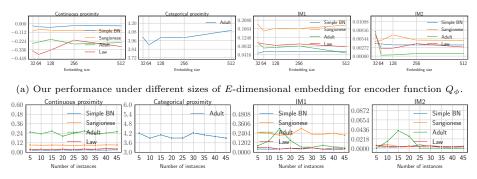
Figure 2: Baseline results in terms of **Continuous proximity** and **Categorical proximity**. Higher continuous and categorical proximity are better.

(a) Our performance under different sizes of E-dimensional embedding for encoder function Q_ϕ.

(b) Our performance under different numbers of K-nearest neighbors for class prototype

Figure 3: Sensitivity of hyperparameters.

ure 3a that the performance of continuous proximity for `Simple-BN`, `Sangiovese` and `Adult` datasets is nearly stable with different embedding sizes, while `Law` witnesses a quite significant variation, increasing from around -0.336 to -0.224 corresponding to embedding sizes of 32 to 256, followed by a slight decrease to -0.33 (embedding size 512). A similar pattern also is recorded for the remaining metrics including categorical proximity, IM1, and IM2 with the good and stable performance at an embedding size of 256. The slight small fluctuations possibly illustrate that the impact of embedding size on the model performance is not very significant. Moreover, 256 is the preferable embedding size, while the

sizes of 32 and 512 seem to be relatively small and large to sufficiently capture latent information for embedding vectors. Regarding categorical proximity, the performance declines slightly by 0.1 from 32 to 64, and thereafter varies slightly around 4.0 - 4.09 with embedding sizes of 128, 256, and 512. On the other hand, as can be seen from Figure 3b, IM1 and IM2 demonstrate a similar pattern illustrated by the worst performance when the number of instances of 15, followed by a stagnant performance from 25 to 45 instances. It is believed that the similar trend occurring in IM1 and IM2 is reasonable due to their similar properties illustrated in Section 4.2. Meanwhile, there is no significant variation in the performance of continuous and categorical proximity across four datasets. These results suggest that the performance of our proposed method witnesses a small variation in all evaluation metrics regarding two hyperparameters (embedding sizes and numbers of nearest neighbors), implying our model's stability and robustness.

5. Conclusion

This paper introduces a novel counterfactual explanation algorithm by integrating the structural causal model and the class prototype. We also proposed formulating the counterfactual generation as a multi-objective problem and construct an optimization algorithm to find the optimal counterfactual explanation in an effective manner. Our experiments validate that our method outperforms the state-of-the-art methods on many evaluation metrics. For future work, we plan to extend our framework to the imperfect structural causal model that is very commonplace in real-world scenarios. Meanwhile, other multi-objective optimization algorithms such as reinforcement learning and multi-task learning are also worthy of investigation.

References

[1] A. Završnik, Algorithmic justice: Algorithms and big data in criminal justice settings, European Journal of Criminology (2019) 1477370819876762.

[2] H. Kaur, H. Nori, S. Jenkins, R. Caruana, H. Wallach, J. Wortman Vaughan, Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning, in: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 2020, pp. 1–14.

[3] J. Galindo, P. Tamayo, Credit risk assessment using statistical and machine learning: basic methodology and risk modeling applications, Computational Economics 15 (1) (2000) 107–143.

[4] P. Schwab, W. Karlen, Cxplain: Causal explanations for model interpretation under uncertainty, arXiv preprint arXiv:1910.12336.

[5] J. J. Williams, J. Kim, A. Rafferty, S. Maldonado, K. Z. Gajos, W. S. Lasecki, N. Heffernan, Axis: Generating explanations at scale with learnersourcing and machine learning, in: Proceedings of the Third (2016) ACM Conference on Learning@ Scale, 2016, pp. 379–388.

[6] Q. Zhao, T. Hastie, Causal interpretations of black-box models, Journal of Business & Economic Statistics 39 (1) (2021) 272–281.

[7] B. Ustun, A. Spangher, Y. Liu, Actionable recourse in linear classification, in: Proceedings of the Conference on Fairness, Accountability, and Transparency, 2019, pp. 10–19.

[8] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, P. Flach, Face: feasible and actionable counterfactual explanations, in: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 2020, pp. 344–350.

[9] S. Sharma, J. Henderson, J. Ghosh, Certifai: A common framework to provide explanations and analyse the fairness and robustness of black-box models, in: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 2020, pp. 166–172.

[10] A. Dhurandhar, T. Pedapati, A. Balakrishnan, P.-Y. Chen, K. Shanmugam, R. Puri, Model agnostic contrastive explanations for structured data, arXiv preprint arXiv:1906.00117.

[11] R. M. Grath, L. Costabello, C. L. Van, P. Sweeney, F. Kamiab, Z. Shen, F. Lecue, Interpretable credit application predictions with counterfactual explanations, arXiv preprint arXiv:1811.05245.

[12] M. T. Lash, Q. Lin, N. Street, J. G. Robinson, J. Ohlmann, Generalized inverse classification, in: Proceedings of the 2017 SIAM International Conference on Data Mining, SIAM, 2017, pp. 162–170.

[13] D. Mahajan, C. Tan, A. Sharma, Preserving causal constraints in counterfactual explanations for machine learning classifiers, arXiv preprint arXiv:1912.03277.

[14] R. K. Mothilal, A. Sharma, C. Tan, Explaining machine learning classifiers through diverse counterfactual explanations, in: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, 2020, pp. 607–617.

[15] C. Fernández-Loría, F. Provost, X. Han, Explaining data-driven decisions made by ai systems: The counterfactual approach, arXiv preprint arXiv:2001.07417.

[16] J. Moore, N. Hammerla, C. Watkins, Explaining deep learning models with constrained adversarial examples, in: Pacific Rim International Conference on Artificial Intelligence, Springer, 2019, pp. 43–56.

[17] S. Wachter, B. Mittelstadt, C. Russell, Counterfactual explanations without opening the black box: Automated decisions and the gdpr, Harv. JL & Tech. 31 (2017) 841.

[18] A. Dhurandhar, P.-Y. Chen, R. Luss, C.-C. Tu, P. Ting, K. Shanmugam, P. Das, Explanations based on the missing: Towards contrastive explanations with pertinent negatives, arXiv preprint arXiv:1802.07623.

[19] Z. Cui, W. Chen, Y. He, Y. Chen, Optimal action extraction for random forests and boosted trees, in: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 179–188.

[20] K. Kanamori, T. Takagi, K. Kobayashi, H. Arimura, Dace: Distribution-aware counterfactual explanation by mixed-integer linear optimization, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, Christian Bessiere (Ed.). International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 2855–2862.

[21] R. M. Grath, L. Costabello, C. L. Van, P. Sweeney, F. Kamiab, Z. Shen, F. Lecue, Interpretable Credit Application Predictions With Counterfactual Explanations, arXiv:1811.05245 [cs]ArXiv: 1811.05245.

[22] A. Van Looveren, J. Klaise, Interpretable counterfactual explanations guided by prototypes, arXiv preprint arXiv:1907.02584.

[23] M. Pawelczyk, K. Broelemann, G. Kasneci, Learning model-agnostic counterfactual explanations for tabular data, in: Proceedings of The Web Conference 2020, 2020, pp. 3126–3132.

[24] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, M. Detyniecki, Comparison-based inverse classification for interpretability in machine learning, in: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Springer, 2018, pp. 100–111.

[25] S. Dandl, C. Molnar, M. Binder, B. Bischl, Multi-objective counterfactual explanations, arXiv preprint arXiv:2004.11165.

[26] C. Bliek1ú, P. Bonami, A. Lodi, Solving mixed-integer quadratic programming problems with ibm-cplex: a progress report, in: Proceedings of the twenty-sixth RAMP symposium, 2014, pp. 16–17.

[27] A. Artelt, B. Hammer, Convex density constraints for computing plausible counterfactual explanations, arXiv preprint arXiv:2002.04862.

[28] C. Russell, Efficient search for diverse coherent explanations, in: Proceedings of the Conference on Fairness, Accountability, and Transparency, 2019, pp. 20–28.

[29] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, Advances in neural information processing systems 30.

[30] H. Jia, Y.-m. Cheung, J. Liu, A new distance metric for unsupervised learning of categorical data, IEEE transactions on neural networks and learning systems 27 (5) (2015) 1065–1079.

[31] L. Kaufman, P. J. Rousseeuw, Finding groups in data: an introduction to cluster analysis, Vol. 344, John Wiley & Sons, 2009.

[32] M. van de Velden, A. Iodice D'Enza, A. Markos, Distance-based clustering of mixed data, Wiley Interdisciplinary Reviews: Computational Statistics 11 (3) (2019) e1456.

[33] A. H. Foss, M. Markatou, B. Ray, Distance metrics and clustering methods for mixed-type data, International Statistical Review 87 (1) (2019) 80–109.

[34] J. Pearl, Causal inference in statistics: An overview, Statistics Surveys 3 (2009) 96–146. `doi:10.1214/09-SS057`.

[35] J. Pearl, et al., Models, reasoning and inference, Cambridge, UK: CambridgeUniversityPress 19.

[36] S. Shimizu, Lingam: Non-gaussian methods for estimating causal structures, Behaviormetrika 41 (1) (2014) 65–98.

[37] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii, in: International conference on parallel problem solving from nature, Springer, 2000, pp. 849–858.

[38] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: Nsga-ii, IEEE transactions on evolutionary computation 6 (2) (2002) 182–197.

[39] C. R. Raquel, P. C. Naval Jr, An effective use of crowding distance in multiobjective particle swarm optimization, in: Proceedings of the 7th annual conference on Genetic and evolutionary computation, 2005, pp. 257–264.

[40] X. Xu, Z. Shi, Multi-objective based spectral unmixing for hyperspectral images, ISPRS Journal of Photogrammetry and Remote Sensing 124 (2017) 54–69.

[41] D. Whitley, A genetic algorithm tutorial, Statistics and computing 4 (2) (1994) 65–85.

[42] J. Blank, K. Deb, Pymoo: Multi-objective optimization in python, IEEE Access 8 (2020) 89497–89509.

[43] A. Magrini, S. Di Blasi, F. M. Stefanini, A conditional linear gaussian network to assess the impact of several agronomic settings on the quality of tuscan sangiovese grapes, Biometrical Letters 54 (1) (2017) 25–42.

[44] D. Dua, C. Graff, UCI machine learning repository (2017).
URL http://archive.ics.uci.edu/ml

[45] L. F. Wightman, Lsac national longitudinal bar passage study. lsac research report series.

[46] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, M. Detyniecki, Inverse classification for comparison-based interpretability in machine learning, arXiv preprint arXiv:1712.08443.

[47] S. Sharma, J. Henderson, J. Ghosh, Certifai: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models, arXiv preprint arXiv:1905.07857.

[48] A. Zheng, A. Casari, Feature engineering for machine learning: principles and techniques for data scientists, " O'Reilly Media, Inc.", 2018.

[49] C. R. Turner, A. Fuggetta, L. Lavazza, A. L. Wolf, A conceptual basis for feature engineering, Journal of Systems and Software 49 (1) (1999) 3–15.

[50] G. Dong, H. Liu, Feature engineering for machine learning and data analytics, CRC Press, 2018.