Counterfactual Generation Through Multi-objective Constrained Optimisation

Wellington Rodrigo Monteiro (rodrigo.wellington@pucpr.edu.br)
Pontifical Catholic University of Parana: Pontificia Universidade Catolica do Parana
https://orcid.org/0000-0001-8450-8714

Gilberto Reynoso-Meza
Pontifical Catholic University of Parana: Pontificia Universidade Catolica do Parana

Research Article

Keywords: Explainable artificial intelligence, Counterfactual generation, Multi-objective optimisation, Genetic algorithms

Posted Date: February 21st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1325730/v1

Counterfactual Generation Through Multi-objective Constrained Optimisation

Wellington Rodrigo Monteiro[1*] and Gilberto Reynoso-Meza[1†]

[1*]Industrial and Systems Engineering Graduate Program, Pontifícia Universidade Católica do Paraná, R. Imaculada Conceição 1155, Curitiba, 80215-901, Paraná, Brazil.

*Corresponding author(s). E-mail(s): wellington.monteiro@pucpr.edu.br;
Contributing authors: g.reynosomeza@pucpr.br;
†Wellington designed the study and wrote the manuscript. Gilberto supervised the project and contributed to the final version of the manuscript.

Abstract

Explainable artificial intelligence (XAI) techniques are often adopted to provide transparent feedback to decision-makers evaluating the results of trained machine learning (ML) models. One of these techniques covers the generation of counterfactuals – slightly modified instances based on a given input that can change its predicted outcome. These counterfactuals enable decision-makers to know what actions they could do to modify an outcome and improve the trust and transparency of a trained ML model. This research aims to develop and apply a novel constrained multi-objective optimisation design (MOOD) to generate counterfactuals in classification and regression problems. The MOOD is composed of three steps: the definition of a constrained multi-objective problem (MOP); the multi-objective optimisation (MOO) using a multi-objective evolutionary algorithm (MOEA); and the multi-criteria decision making (MCDM) with a domination filter and the TOPSIS ranking technique. This MOOD generated large numbers of valid counterfactuals and selected and ranked only the most relevant to the decision-maker. Furthermore, the results show that the proposed MOOD outperformed other architectures available for production use in at least one of the objectives measured for all datasets analysed in both regression and classification problems.

Keywords: Explainable artificial intelligence, Counterfactual generation, Multi-objective optimisation, Genetic algorithms

1 Introduction

Real-world machine learning (ML) problems in industry often have a large number of features [1]. Depending on the dataset such features can be categorical or continuous; engineered (e.g., calculated values) or raw features (e.g., brand category or a sensor reading). In order to process, learn from and predict from these features, ML algorithms play a preeminent role. There are four ML algorithm types: supervised, unsupervised, semi-supervised, and reinforcement learning [2]. Supervised learning attempts to create models from labeled data samples - both continuous (regression) or discrete (classification) values [1]. These models are used in industrial applications where large datasets are commonplace [3–5] and require equally large and complex trained ML models [6].

Meanwhile, black-box models are too complex to have its underlying logic properly extracted and

understood by humans [7] or that the user can only send the inputs and read the outputs from it [8]. These models are spread in many applications used in the industry [9]. Depending on the application, the transparency of decisions taken by these models are required by humans to build trust [10] on them - this is paramount in safety critical or financial applications, for instance [9].

In order to enable transparency, the adoption of explainable artificial intelligence (XAI) techniques are presented as a field of study which includes different approaches to explain the predictions or a model behavior to humans [7]. In XAI, there are techniques ranging from simple surrogate functions to more complex meta-explanations [9], or approaches attempting to understand how an overall trained model works (global interpretability) to explainining a particular prediction (local interpretability) [11]. This manuscript proposes a strategy focused on local interpretability – namely *counterfactual explanations* [12].

Counterfactual explanations are hypothetical examples based on a given case with a different outcome [13]. One of the classical examples [12] is based on credit loan applications. Consider a person who tried to apply for a loan and was denied. A counterfactual is a fictitious example generated from an algorithm which is as close as possible to the original individual in all the reported characteristics such as age, income and job and with the desired outcome. In this example, the desired outcome is the approved loan. It is also expected that these counterfactuals are diverse and feasible [13]. By *diverse* it is understood that it is interesting to generate different combinations capable of leading to the desired outcome. By *feasible* it is expected that the counterfactuals respect real-world constraints. Feasible examples include suggestions with small increases in the age and/or income. Unfeasible examples include doubling the income or being two years younger.

While there are counterfactual generation algorithms available for use in industry [14–17], pitfalls hinder their adoption in real-world applications such as not supporting regression problems [15], not supporting classification problems [14], or not being model-agnostic [17]. Besides, the counterfactual generation "working space" can be seen as a constrained multi-objective optimization problem where there is a set of objectives to be met (e.g. being feasible, having a different outcome and as close as possible to the original individual) and constraints (e.g. the allowed working range for one or more variables). Multi-objective Optimization Design (MOOD) procedures using Multi-objective Evolutionary Algorithms (MOEAs) [18] thrive well in these cases. They are shown as valuable tools for success in areas such as industrial control systems and engineering [19, 20]. Surprisingly, these techniques were not deeply studied in XAI applications.

In this paper, we propose using a counterfactual generation MOOD covering both classification and regression problems considering counterfactual generation is a constrained multi-objective optimization problem. Furthermore, the design is model-agnostic and is expected to work with already deployed models in the industry since it does not require retraining the base ML model. We evaluated the effectiveness of the proposed MOOD against three classification benchmark problems and two regression benchmark problems by comparing the counterfactuals generated by the proposed method against other popular implementations in Python.

The main contributions of this paper can be summarized as follows:

- We propose a novel multi-objective problem (MOP) for counterfactual generation. This MOP definition is formatted as a constrained problem with three objectives and three constraints. The constraints ensure a hard limit on the amount of allowed variables to be modified from the input data. The new outcomes follow a user-defined range of what would be considered a valid counterfactual prediction range.
- We propose using the standard implementation of the Unified Nondominated Sorting Genetic Algorithm (U-NSGA-III) in the multi-objective optimization (MOO) step. U-NSGA-III maintains diversity of the solutions through reference directions and has better performance than NSGA-III due to the introduction of tournament pressure [21].
- We propose repeating the MOO step with different limits on the amount of variables changed to increase the diversity of counterfactuals generated.
- We propose a multi-criteria decision making (MCDM) step with a domination filter and the

Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) method to rank and select the best counterfactuals.
- We demonstrate our proposed method against other counterfactual generation architectures considering both regression and classification problems.

The rest of the paper is organized as follows: the second section described related earlier works. The third section covers the theoretical background required for this manuscript, explaining in deeper detail what are counterfactuals and counterfactual generation algorithms. It also introduces the reader to concepts related to optimization: single-objective optimization (SOO) and MOO. While the proposed implementation does not specifically cover it, SOO lays the ground rules of MOO and was also used by other techniques. The fourth section covers the proposed, novel constrained MOOD for counterfactual generation problems. The fifth and sixth sections shows the results found by the implementation of the proposed method against other implementations for classification and regression problems, respectively. Finally, the seventh and last section has the concluding remarks of the authors of this manuscript.

2 Related works

There are some proposals in the literature with the objective of generating counterfactuals through optimisation. [22] formulated a constrained single-objective optimisation problem to minimise the distance between an input and its counterfactual. This minimisation problem is subject to calculating the difference between the predicted outcome of the input and its counterfactual. Only a single counterfactual is provided in this research since it is formulated as a single-objective problem. Furthermore, it does not consider the number of variables being modified.

[23] proposed a method named *Generalized Inverse Classification*. It is also a constrained single-objective optimisation problem. It is intended for classification problems, and its objective is to minimise the cost of changing a subset of directly changeable features from the input subject to a user-defined budget of allowed changes.

[24] proposed an unconstrained MOP where it attempts to generate counterfactuals. The outcome of these counterfactuals should be, at the same time: as close as possible to the desired outcome; with the smallest distance as possible from the original input; with the least number of changed features and as close as possible from an individual (or set of individuals) in the original dataset where the input originated. It includes a modified version of the Nondominated Sorting Genetic Algorithm (NSGA-II) to operate in a mixed discrete and continuous search space. However, it does not offer alternatives for the users to select the best counterfactuals. While it is suited to regression problems, it also does not show how it could work with continuous allowed outcomes – e.g., in a housing price dataset, the decision-maker may be interested what actions he or she can do to increase the median house value from USD 300000 to USD 500000. However, relative values (e.g., USD 500500, USD 499500, USD 500100) can be acceptable counterfactuals depending on the use-cases instead of the exact prediction of USD 500000.

[12] proposed an unconstrained single-objective optimisation problem that generates counterfactuals as close as possible from the original input while being equal to the desired outcome. [13] proposed an unconstrained single-objective optimisation problem that considers the proximity of a counterfactual from its output as well as the diversity of the generated counterfactuals.

3 Background

In this section, we present the technical concepts behind our proposal. It includes the concepts behind the counterfactual generation and multi-objective optimisation. We also include the concepts of single-objective optimisation since it is used by other proposals and is a starting point to understand the concepts of multi-objective optimisation better.

3.1 Counterfactual Generation

In order to better understand how do intelligent systems make their decisions, there are two terms often used in an interchangeable fashion in literature: *interpretability* and *explainability* [10].

However, there are meaningful differences between them [25]. Interpretability is in charge of providing meaningful descriptions to humans and in a way they can understand. On the other hand, explainability is a broader term and covers both interpretability and completeness. Completeness is another term that targets an *accurate* and auditable description of how a machine learning model makes its decisions. Therefore, even though explainable models are also interpretable, not all interpretable models are also explainable due to their lack of completeness [25]. For example, decision trees generated to explain more complex machine learning models such as deep neural networks (DNNs) may be interpretable due to their easily understandable nature to humans. However, they might not respect the completeness compared against the original model, for instance [26].

Moreover, in order to better understand the outputs of machine learning models, it is possible to consider two different types of interpretability: global and local [27]. Global interpretability defines the general patterns and information that describes the overall system. In contrast, local interpretability deals with a single individual or instance where it attempts to explain the outcomes of this individual mentioned earlier – in other words, *why* the output of a machine learning model had a given result. One type of local interpretability technique is the *counterfactual explanations*. Counterfactual explanations are a type of *ex-post* explanation [11] that generates modified individuals based on a single, original example where these individuals have a different output when compared against the original one [12].

Counterfactual explanations try to identify the minimum changes required to a given individual to change its outcome. However, the slightest change does not necessarily represent a feasible alternative [12]. In an anecdotal example represented in Table 1, an individual had his loan denied. In order to change the outcome from 'denied' to 'approved', the first counterfactual suggested turning only one day younger. Naturally, it is impossible to happen. However, the second counterfactual could be feasible since it suggests that getting a 1% pay rise also modifies the outcome.

While the first counterfactual is the smallest change scale-wise since the age is commonly measured in years, it is not a feasible counterfactual in a real-world situation. The third may also not be feasible considering the reality of the individual under analysis, but the third could be deemed a good alternative. These counterfactual explanations are a way to help understand how a given machine learning algorithm works and identify its weaknesses and a potential tool for decision-makers in the industry to assess possible scenarios and changes required in their contexts, observing possible improvements.

Table 1 Example of different counterfactuals generated from an original individual. Values in bold indicate a change from the original.

Features	Original	CF 1	CF 2
Age	27.50	**27.49**	27.50
Car owner	Yes	Yes	Yes
Income	50000	50000	**50500**
Education	High school	High school	High school
Outcome	Denied	**Approved**	**Approved**

Let us consider another anecdotal example, where a faulty part had been manufactured and correctly detected by a machine learning algorithm considering a myriad of sensors in a factory. A set of counterfactuals generated based on this faulty part can show that small changes in the manufacturing line could prevent newer cases from showing up. On the other hand, if a given case is a false negative – i.e. it was a faulty part but incorrectly shown as a good part by the machine learning algorithm – the counterfactuals show what had contributed to the wrong decision. Additionally, it helps the decision-makers to determine whether new or modified information (e.g. new sensors or new calculated columns) are required.

The counterfactual generation should respect two main objectives: diversity and proximity [13]. Diverse solutions are, by definition, sufficiently diverse from each other to increase the chances of having at least one valuable solution to the decision-maker. In contrast, proximity dictates the expectation of having counterfactuals similar to the original individual.

The counterfactual generation algorithms are part of the XAI field and, according to the ontology proposed in [28], are post-hoc, local and model-agnostic. Among the techniques used to generate the counterfactuals are, for example, linear models; Naive Bayes; and decision tree-based models [29] in addition to the single-objective optimisation algorithms [12, 13, 15] and, more

recently, multi-objective optimisation implementations [24, 30]. This manuscript uses constrained multi-objective optimisation to generate possible results considering industrial applications.

3.2 Single-objective Optimisation

Single-objective Optimisation (SOO) covers techniques capable of solving a mathematical problem with a single value that must be minimised or maximised as an output. These problems are composed of multiple mathematical variables and can be subject to constraints defined beforehand. The general form of an SOO problem is [31]:

$$Find\ \mathbf{x} = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ ... \\ x_n \end{Bmatrix} which\ minimises\ F(\mathbf{x}) \quad (1)$$

Here, $x_i, i = 1, 2, ..., n$ are decision variables, and $F(\mathbf{x})$ is the objective function. The combination of these variables will result in a single value stated by $F(\mathbf{x})$. Therefore, the idea of SOO is to find the combination of the decision variables which will result in the minimum value of $F(\mathbf{x})$. The problem can be bound to constraints which are generally defined as:

$$\begin{aligned} g_j(\mathbf{X}) \leq 0, j = 1, 2, ..., m \\ l_j(\mathbf{X}) = 0, j = 1, 2, ..., p \end{aligned} \quad (2)$$

where n, m and p can be unrelated [31]. Eq. 1 and 2 can be rewritten as:

$$\begin{aligned} \min_{\mathbf{x}} \quad & F(\mathbf{x})(x_1, x_2, ..., x_n) && (3a) \\ \text{subject to} \quad & g(\mathbf{x}) \leq 0 && (3b) \\ & l(\mathbf{x}) = 0 && (3c) \end{aligned}$$

where $g(\mathbf{x})$ and $l(\mathbf{x})$ refer to the constraints in Eq. 2.

In the context of counterfactual generation, the single-objective problem could be represented in the form of minimising the difference between the target output and the counterfactual output *or* the distance between original individual inputs against the counterfactual inputs in a single metric [12]. It also could be represented as the minimisation of a combined weighted loss function including the distance of the inputs of a single counterfactual compared against other counterfactuals [13], or minimising the factual loss combined with the imbalance error and the introduction of a weight decay [32, 33].

By attempting to use SOO strategies to generate counterfactuals, these strategies fall short in considering tradeoffs between different generated solutions (i.e. counterfactuals) – one counterfactual can have a more significant distance on its inputs against the original solution but be closer to the desired target. In this example, depending on the sum of these differences and the configuration of the hyperparameters, one feasible and acceptable solution can be removed because it does not represent a better counterfactual when compared against other counterfactuals generated by the own algorithm. In another example, if the single objective minimises the distance between a counterfactual and an individual, only one solution would be suggested ignoring all other aspects equally valuable for the decision-maker.

The three most popular Github repositories for counterfactual generation sorted by "most stars" at the time of writing this manuscript were DiCE [16], an implementation of [13]; Alibi [15], based on [12, 34]; and cfrnet [14], based on [32, 33], as shown in Table 2. DiCE [16] is an implementation that offers customisation options for limiting which variables must be changed, which variables are easier to be modified, the upper and lower bounds for continuous features and the allowed labels for categorical features. It only works with binary classification problems or regression problems and includes a filtering process to remove infeasible examples after all the counterfactuals were generated, which, in turn, impacts the algorithm efficiency [13]. Alibi [15] is an implementation that allows customising the upper and lower bounds for continuous variables or the allowed labels for categorical features in one of its implementations. However, it does require a black-box model that must expose the predicted class probabilities. It also does not work with either regression problems or classification models that do not return the probabilities for all the considered classes. Cfrnet [14] is an implementation intended to work with regression problems. On the other hand, in addition to not offering support for classification

problems due to its intended use-case application, it also does not offer similar input customisation options.

Table 2 Example of different counterfactuals generated from an original individual. Values in bold indicate a change from the original.

Support	DiCE	Alibi	cfrnet
Immutable variables	Yes	Yes	No
Upper/lower bounds per variable	Yes	Yes	No
Model-agnostic	No	Yes	No
Categorical features	Yes	Yes	No
Binary classification	Yes	Yes	No
Multiclass classification	No	Yes	No
Regression problems	No	No	Yes

3.3 Multi-objective Optimisation

MOO encompasses a set of techniques that attempts to find all the designs that can satisfy the optimisation conditions presented in the form of a mathematical function. This function returns an objective vector instead of a single value [35] and is generally defined as:

$$\min_{\mathbf{x}} \quad \mathbf{J}(\mathbf{x}) = [J_1(\mathbf{x}), ..., J_n(\mathbf{x})] \tag{4a}$$

$$\text{subject to} \quad \mathbf{g}(\mathbf{x}) \leq 0 \tag{4b}$$

$$\mathbf{h}(\mathbf{x}) = 0 \tag{4c}$$

Since the results are presented in a vector, instead of having one single value representing the best solution available (global minimum) there is a Pareto front, a location within the n-dimension space where each dimension represents an objective. In this front all the non-dominated solutions are located [36]. Since in real-world situations the Pareto front is often unknown, the non-dominated solutions found by the MOO algorithms are part of the Pareto front *approximation* instead. All of the solutions in the Pareto front are non-dominated (i.e. there were no better solutions found considering all the objectives). There are a number of techniques used to implement MOO algorithms such as genetic algorithms, differential evolution, swarm intelligence, simulated annealing and variable neighborhood search [37]. MOO strategies were commonly used in industrial applications but recently were found to be useful in XAI and, more specifically, in counterfactual generation.

Since the results are presented in a vector, instead of having one value representing the best solution available (global minimum), there is a Pareto front, a location within the n-dimensional space where each dimension represents an objective. On this front, all the non-dominated solutions are located [36]. Since the Pareto front is often unknown in real-world situations, the non-dominated solutions found by the MOO algorithms are part of the Pareto front *approximation* instead. All of the solutions in the Pareto front are non-dominated (i.e. there were no better solutions found considering all the objectives). Several techniques are used to implement MOO algorithms, such as genetic algorithms, differential evolution, swarm intelligence, simulated annealing and variable neighbourhood search [37]. MOO strategies were commonly used in industrial applications but only recently were found to be helpful in XAI and, more specifically, for counterfactual generation.

Specifically on image classification tasks, [30] has introduced a counterfactual generation algorithm using MOO considering the simultaneous minimisation of three objectives: unlikeliness of a counterfactual to be generated by a computer program; the probability of confusing the target model; and the number of adversarial changes done in the original image. These three objectives were solved through a single MOO algorithm. It is noted that the intention was not to minimise the number of changes in itself but to have plausible changes instead.

The algorithm proposed by [24] is model-agnostic and can work with classification and regression problems. Furthermore, it works with datasets containing both categorical and continuous features and has been tested against binary classification problems. On the other hand, it does not work with trained classification models that do not return the probabilities for all the considered classes. It is also prone to generate counterfactuals acceptable by the machine learning model but unusable in real-world situations since it does not set upper and lower bounds to the variables (e.g., one of the tests proposed using the diabetes dataset [38], one of the recommendations included being younger). Nevertheless, this proposal is promising considering its agnostic nature and for generating multiple

solutions for different problems. Four objectives are introduced: the proposed MOP is stated as being the minimisation of the difference between the desired outcome and the outcome found by the counterfactual; the minimisation of the Gower distance between the inputs of the original individual and the counterfactual; the minimisation of the number of changed features; and the minimisation of the weighted average Gower distance between the counterfactual and a set of observed data. The Gower distance [39] is a helpful measure for instances containing both continuous and categorical features, as seen in [40], for example. However, this problem is not bound to any constraints, and the plausibility of the counterfactuals generated is rooted in the observed data's probability distribution.

This section provided the background covering counterfactual generation, single-objective optimisation and multi-objective optimisation. Based on this background, the next section will provide the proposal of the present manuscript, which is a MOOD procedure intended to be model-agnostic and work with both classification and regression problems. The proposed approach suggests treating counterfactual generation as a constrained multi-objective optimisation problem.

4 Proposal

Considering the strengths of other algorithms and recognising the usefulness of MOO implementations in XAI, this paper proposes a new MOOD procedure. The proposal uses a generate-first, choose-later (GFCL) approach intended to be used in real-world XAI applications. It does generate counterfactuals respecting bounds imposed by the decision-maker. It can also be used alongside library-agnostic machine learning models to avoid reworking established code.

By "MOOD in a GFCL" approach, it is understood as being both the statement of the MOP, the MOO itself and the MCDM displaying the counterfactuals generated. There were no other studies on the usage of MOO algorithms to generate counterfactuals by treating the MOP as a constrained problem and with an MCDM step when this manuscript had been created, according to the research from the authors.

4.1 Multi-objective Problem

The MOP of the proposed MOOD implementation is stated as:

$$\min_{\mathbf{x}} \quad \mathbf{J}(\mathbf{x}) = [J_1(\mathbf{x}), J_2(\mathbf{x}, \mathbf{X}), J_3(\mathbf{x})] \tag{5a}$$

$$\text{subject to} \quad \mathbf{g}(\mathbf{x}) \leq 0 \tag{5b}$$

$$\mathbf{h}(\mathbf{x}) \leq 0 \tag{5c}$$

$$\mathbf{i}(\mathbf{x}) \leq 0 \tag{5d}$$

$$\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U \tag{5e}$$

The three objectives described in Eq. 5 refer to a counterfactual $\mathbf{x}$ generated from an original individual $\mathbf{x}^0$ member of a larger dataset $\mathbf{X}$ respecting the lower and upper bounds determined by $\mathbf{x}^L$ and $\mathbf{x}^U$, respectively. The general statement of $J_1(\mathbf{x})$ is shown in Eq. 6 and 7. The definition in the Eq. 6 is valid for both regression models or classification models returning the probabilities for each class. It minimises the unsigned difference of the outcome y found by the machine learning model relative to the counterfactual $\mathbf{x}$ and the desired outcome y^d.

$$J_1(\mathbf{x}) = \|y - y^d\| \tag{6}$$

For the cases when the classification models only provide the predicted class, Eq. 7 can be used. $y = y^d$ refers to where the predicted class y from a given counterfactual and the desired class y^d are the same.

$$J_1(\mathbf{x}) = \begin{cases} 0 & \text{if } y = y^0 \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

The general statement of $J_2(\mathbf{x})$ is shown in Eq. 8 and addresses the Gower distance δ_{Gower} between $\mathbf{x}$ and $\mathbf{x}^0$ considering all their n attributes. This objective was first introduced in [24].

$$J_2(\mathbf{x}, \mathbf{X}) = \sum_{j=1}^{n} \delta_{Gower}(\mathbf{x}_j, \mathbf{x}_j^0, \mathbf{X}) \tag{8}$$

The Gower distance δ_{Gower} is stated in Eq. 9 and considers both categorical and numeric attributes, and the difference between the upper and lower bounds for the j-th column (attribute) in the original dataset $\mathbf{X}$. $\mathbb{I}$ is adopted to the cases where the j-th column is categorical. In these

cases, 0 is attributed where the classes are the same and 1 otherwise.

$$\delta_{Gower}(\mathbf{x}_j, \mathbf{x}_j^0, \mathbf{X}) = \begin{cases} \frac{\|\mathbf{x}_j - \mathbf{x}_j^0\|}{\|max(\mathbf{X}_j) - min(\mathbf{X}_j)\|} & \text{if } \mathbf{x}_j \text{ is numeric} \\ \mathbb{I}_{\mathbf{x}_j \neq \mathbf{x}_j^0} & \text{otherwise} \end{cases} \tag{9}$$

The general statement of $J_3(\mathbf{x})$ is shown in Eq. 10 and refers to the determination of the number of modified attributes using the L_0 norm [24] considering both categorical and numeric attributes.

$$J_3(\mathbf{x}) = \mathbf{x} - \mathbf{x}^0{}_0 \tag{10}$$

Considering the possibilities of using the proposed algorithm in real-world scenarios where the decision-maker might require tighter controls on the attributes bound for changes or possess additional knowledge on the feasible decision space considering the particularities and requirements specific to $\mathbf{x}^0$, we propose three constraints first presented in Eq. 5b, 5c and 5d. The first constraint – namely $g(\mathbf{x})$ and shown in Eq. 11 – introduces a hyperparameter containing the maximum number of changed variables allowed by the decision-maker (mcv). This constraint ensures a hard limit on the number of allowed variables to be modified in $\mathbf{x}$.

$$\mathbf{g}(\mathbf{x}) = \mathbf{x} - \mathbf{x}^0{}_0 - mcv \tag{11}$$

The constraints $\mathbf{h}(\mathbf{x})$ (Eq. 5c) and $\mathbf{i}(\mathbf{x})$ (Eq. 5d) introduce an allowed range of operation for y. While J_1 attempts to get as close to y^d as possible, these constraints ensure that the generated counterfactuals are helpful to the decision-maker. Having that said, Eq. 12 shows how $\mathbf{h}(x)$ is designed: for classification models having as the output only the predicted class (namely *simple* models), the same logic applied to J_1 (Eq. 7) applies. Otherwise, the difference between the minimum acceptable value determined as a hyperparameter in y^{min} and y is considered. For both regression and classification models returning the class probabilities, y^{min} must be lower than or equal to y^d.

$$\mathbf{h}(\mathbf{x}) = \begin{cases} J_1 & \text{for classification (simple)} \\ y^{min} - y & \text{otherwise} \end{cases} \tag{12}$$

On the other hand, Eq. 13 shows the logic behind $\mathbf{i}(\mathbf{x})$, which is intended to be a counterpart of $\mathbf{g}(\mathbf{x})$. Therefore, while $\mathbf{h}(\mathbf{x})$ attempts y respecting the lower bound established by y^{min}, another hyperparameter is introduced to hold the upper bound – y^{max}.

$$\mathbf{i}(\mathbf{x}) = \begin{cases} J_1 & \text{for classification (simple)} \\ y - y^{max} & \text{otherwise} \end{cases} \tag{13}$$

4.2 Multi-objective Optimisation

This proposal was implemented in Python, considering the adoption of this programming language in real-world, industrial machine learning applications and uses the U-NSGA-III algorithm as a backend. It is a genetic algorithm (GA) intended for multi-objective optimisation problems. This algorithm uses concepts from nature such as mating and survival selection to generate the fittest solutions. It also implements the concept of reference directions to be used in order to maintain diversity between the solutions found. This algorithm also introduces a modified tournament algorithm to select the fittest individuals in an attempt to achieve better performance [21, 41].

This implementation introduces a repair function specific for counterfactual generation. The repair function modifies the counterfactuals created at each generation and does the following:

1. Integer attributes are rounded to the next integer;
2. Values generated for label encoded categories are rounded to the next valid value;
3. Values generated for one-hot encoded categories are reset to the closest row in an identity matrix with the same size measured by its Euclidean distance.

In order to introduce greater diversity on J_3, the MOO runs mcv times. Initially, mcv starts with the number of variables defined by the decision-maker and, after each run, is decreased by one until it reaches the minimum value of 1. This rationale was introduced to ensure greater diversity in the counterfactuals considering J_3.

The MOO implementation also introduces the following hyperparameters:

- *model*: the trained machine learning model to be used to evaluate the generated counterfactuals;

- *method_name*: the machine learning model function used to evaluate the generated counterfactuals (e.g. *predict_proba*; *predict*);
- *immutable_column_indexes*: allows the decision-maker to force the columns that must not be modified (e.g. in a credit card evaluation problem, the age of the requester is an example);
- *upper_bounds* and *lower_bounds*: contains the working ranges for each of the attributes considering the counterfactuals (e.g. in a credit card evaluation problem, the allowed ranges can include only the possible salary increases the requestor could have in the near future or minor changes in the requestor age considering the next n months);
- *categorical_columns* and *integer_columns*: allows the decision-maker to determine which columns allow only integer values and only categories. The algorithm also allows the decision-maker to inform which categories are allowed considering the categories;
- $\mathbf{x}^0$, y^d and mcv.

4.3 Multicriteria Decision Making

Depending on the problem being considered, the number of generated solutions is too high and confusing to generate valuable insights for the decision-maker. Therefore, this manuscript uses two combined techniques to consider only a few but valuable counterfactuals: the domination filter [18] and the TOPSIS method [42].

The domination filter was briefly introduced in the previous section. It is a quick filter executed after the MOO step is finished. After finding all feasible solutions, only the unique, non-dominated solutions are selected [18].

After running the domination filter, we ensure that only the non-dominated solutions are selected. On the other hand, the number of solutions might still be too high. TOPSIS was initially proposed in [42]. It is a ranking method that attempts to find the solutions as close as possible to an ideal solution (i.e., where the result is zero for all three objectives) and as far as possible to the negative ideal solution (i.e., where the result is $+\infty$ for all three objectives). The solution that best follows these expectations is ranked first; the second best is ranked second, and so on. This technique still finds common use in the present day [43] and was chosen due to its ranking nature. The decision-maker can then check only the n best solutions according to TOPSIS, where n is defined by himself.

5 Classification

In this section, we test the proposed algorithm against two state-of-the-art implementations in Python, which are model-agnostic: DiCE [13, 16] and Alibi [12, 15]. In addition, three datasets commonly used in literature [44] were used to evaluate the performance of the algorithms: the ProPublica risk assessment system (COMPAS), the US Census Income dataset, and the German credit dataset.

All categorical features from all datasets were transformed by label encoding [45]. Each converted dataset was then oversampled through Synthetic Minority Over-sampling Technique – Nominal Continuous (SMOTE-NC) [46]. The resulting dataset was then sampled through stratified random sampling, with 70% of each dataset being used for model training and the remaining 30% for evaluation and counterfactual generation.

All datasets were trained on a random forest classifier. The random forest was selected because it is not very explainable [28] and because it is widely used [47].

5.1 Census

The Census dataset comprises 32561 instances and 14 attributes [48]. Each instance represents a group of people within the United States with measured features such as age, capital gains and losses, hours worked per week, education, relationship, race, native country and work class. Eight features were identified as categorical; 5 attributes were identified as numeric, and the remaining feature measures whether that instance has an income of over USD 50000 per year.

5.1.1 Counterfactual generation

The trained ML model was a scikit-learn pipeline [49] covering the pre-processing and training steps. First, all attributes were normalised after the label encoding considering the categorical columns mentioned earlier. Then, a random forest classifier was trained on the modified dataset.

After training the ML model, 150 individuals from the evaluation dataset were randomly

selected. The trained model had the following confusion matrix based on these individuals:

- 98 true positives, 12 false negatives;
- 30 true negatives, 10 false positives.

For each of these 150 individuals, the three techniques (DiCE, Alibi and the proposed algorithm in this paper) were tasked to find up to 100 counterfactuals. A valid counterfactual is a generated individual with a different outcome than the one initially found by the trained ML model with a probability of over 50% of being a member of the other class. In order to test all three implementations, all their default values remained unchanged except the following parameters in order to ensure any implementation was able to find a counterfactual in up to 5 minutes in a desktop with an Intel© i9-9900K processor, 64 GB DDR4 RAM:

- DiCE: no changes.
- Proposal:
  Maximum number of generations: 100;
  Population size: 100.
- Alibi:
  Number of iterations to adjust the constant scaling the attack loss term: 3;
  Maximum number of iterations: 200.

5.1.2 Results

All three algorithms found valid counterfactuals considering these 150 random individuals. DiCE found a higher number of counterfactuals, followed by the proposed method and Alibi. The descriptive statistics are shown in Table 3.

Table 3 Results for the US Census dataset considering 150 individuals for DiCE (n=14470), the proposal of this paper (n=3675) and Alibi (n=339). The best values for each objective are shown in bold.

Objective	Algorithm	Minimum	Maximum	Median	Mean	Std. Dev.
J_1	DiCE	**0.0000**	**0.4900**	0.2700	0.2749	0.1205
J_1	Proposal	**0.0000**	**0.4900**	**0.1600**	**0.1805**	0.1241
J_1	Alibi	0.0400	**0.4900**	0.3900	0.3702	**0.0936**
J_2	DiCE	0.0008	0.6932	0.0895	0.1037	0.0830
J_2	Proposal	**0.0001**	0.2437	**0.0150**	**0.0348**	**0.0425**
J_2	Alibi	**0.0001**	**0.2320**	0.0778	0.0892	0.0461
J_3	DiCE	**1.0000**	11.0000	**2.0000**	1.9654	1.2978
J_3	Proposal	**1.0000**	5.0000	3.0000	2.5771	0.9883
J_3	Alibi	**1.0000**	**4.0000**	**2.0000**	**1.9440**	**0.7496**

The distribution of these counterfactuals is also shown in Fig. 1. Here, the counterfactuals are shown in a normalised space for visualisation purposes: it is desirable to have counterfactuals as close as possible to zero in J_1 (i.e., when the probability of being a member of the other class is as high as possible according to the base ML model); as close as possible to zero in J_2 (i.e., when the counterfactual is as similar as possible to the original individual); and as close as possible to zero in J_3 (i.e., when the changes affect the smallest number of variables as possible).

Another way to visualise the distribution of these counterfactuals is shown in Fig. 2. The distribution of the populations found for both algorithms are equal only between DiCE and Alibi concerning J_2, according to the Mann-Whitney U test [50].

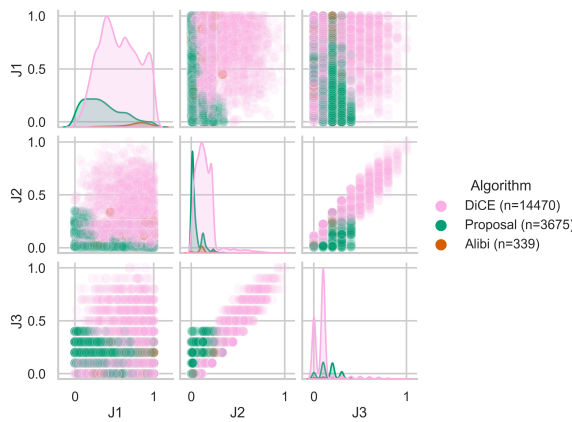

Fig. 1 Pair plot with the counterfactuals found for the US Census dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

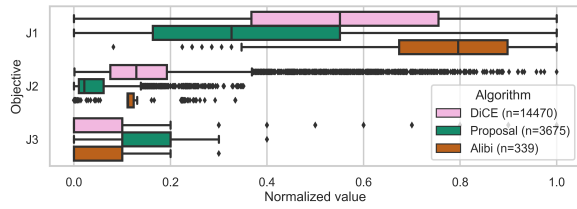

Fig. 2 Boxplot containing the counterfactuals found for the US Census dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

5.1.3 Discussion

In this dataset, the proposed method provided counterfactuals that were as close as possible to the original individuals compared to DiCE and Alibi, as shown by the distribution of the counterfactuals in J_2. However, this does not necessarily mean the number of variables being modified was also the smallest compared to the other methods, as seen in the distribution of J_3. An example was the individual 8043 with the original features:

- Age: 31;
- Work class: Private;
- Fnlwgt (weight): 27494;
- Education: Some-college;
- Marital status: Married-civ-spouse;
- Occupation: Sales;
- Relationship: Husband;
- Race: Asian-Pac-Islander;
- Gender: Male;
- Capital gain: 0;
- Capital loss: 0;
- Hours per week: 50;
- Native country: Taiwan;
- Class: false.

The proposed method found valid counterfactuals with the smallest J_2 values when considering the following changes:

1. $J_2 = 0.000178$:
   Fnlwgt: 24675 (from 27494);
   Capital loss: 1 (from 0).
2. $J_2 = 0.000289$:
   Fnlwgt: 27187 (from 27494);
   Capital loss: 10 (from 0).
3. $J_2 = 0.001057$:
   Capital loss: 10 (from 0);
   Hours per week: 49 (from 50);

The original range for *fnlwgt* was $[13769, 1484705]$, for *capital loss* was $[0, 4356]$, and for *hours per week* was $[1, 99]$, representing proportionally small changes. On the other hand, the three valid counterfactuals found by DiCE with the smallest J_2 values were:

1. $J_2 = 0.003140$:
   Hours per week: 46 (from 50).
2. $J_2 = 0.003814$:
   Capital gain: 4958 (from 0).
3. $J_2 = 0.006261$:
   Fnlwgt: 41868 (from 27494);
   Hours per week: 43 (from 50);

In this case, the changes were proportionally greater than the counterfactuals found by the proposed method. The original range for the *capital gain* was $[0, 99999]$. On the other hand, Alibi found three counterfactuals with the following changes:

1. $J_2 = 0.076923$:
   Relationship: *wife* (from *husband*).
2. $J_2 = 0.077006$:
   Age: 31.070950 (from 31);
   Relationship: *wife* (from *husband*).
3. $J_2 = 0.077032$:
   Age: 31.103214 (from 31);
   Relationship: *wife* (from *husband*).

The changes in the categories cause larger increases in the value of J_2 considering the Gower distance compared to the numeric features. Meanwhile, the differences for the J_2 between the three counterfactuals found by Alibi are small considering the proportion of the changes in the age (original range $[17, 90]$). On the other hand, these counterfactuals have more differences than their counterparts found by the proposed method and DiCE.

5.2 COMPAS

The COMPAS dataset comprises 6172 instances and 11 attributes [51]. Each instance represents an individual recorded in the United States criminal justice system and has features such as age, race, gender and the number of previous crimes. Moreover, six features were identified as categorical; 4 as numeric and the remaining feature measures whether an individual was arrested again within two years from their last release from prison.

5.2.1 Counterfactual generation

The trained ML model was a scikit-learn pipeline [49] with three steps: first, all missing data were imputed with k-nearest neighbours [52]. Then, all values were normalised. Finally, a random forest classifier was used on the training data.

150 random individuals from the evaluation dataset were selected. Based on these individuals, the following confusion matrix was found from the trained model:

- 79 true positives, 1 false negative;

- 65 true negatives, 5 false positives.

Based on the predicted values, DiCE, Alibi and the proposal in this document were tasked to identify 100 counterfactuals per individual. A counterfactual in this case is an individual with an inverse classification than the one predicted initially by the base model – i.e., if the base model predicted a true outcome, only counterfactuals with false outcomes are expected and vice-versa. An acceptable probability of being a member of the expected class was set between 51% up to 100% with the ideal value being 100% (y^d). The variable lower and upper bounds were selected from the minimum and maximum values found for each attribute from the training dataset.

All default values were kept for all counterfactual implementations except for the following parameters to ensure any implementation could execute its process for a given individual in up to 5 minutes in a desktop with an Intel© i9-9900K processor, 64 GB DDR4 RAM:

- DiCE: no changes.
- Proposal:
  Maximum number of generations: 100;
  Population size: 100.
- Alibi:
  Number of iterations to adjust the constant scaling the attack loss term: 3;
  Maximum number of iterations: 200.

5.2.2 Results

For these 150 individuals, only DiCE and the proposal of this paper were capable of finding counterfactuals. Alibi did not find counterfactuals for all the individuals randomly chosen. The results are shown in Table 4. Since all objectives are determined as minimisation, it is desired to have all values as close to zero as possible.

Furthermore, the distribution of these values is better shown in Fig. 3 since it shows the relationships of the solutions across the three objectives. In order to better display the results, all objectives were scaled in the $[0, 1]$ range. The same applies to the boxplot seen in Fig. 4.

The distribution of the populations found for both algorithms is equal between DiCE and the proposed method only for J_1, according to the Mann-Whitney U test [50].

Table 4 Results for the COMPAS dataset considering 150 individuals for DiCE (n=14984), the proposal of this paper (n=214) and Alibi (n=0). The best values for each objective are shown in bold.

Objective	Algorithm	Minimum	Maximum	Median	Mean	Std. Dev.
J_1	DiCE	**0.0000**	**0.4900**	0.0400	**0.0737**	**0.0838**
J_1	Proposal	**0.0000**	**0.4900**	**0.0350**	0.1281	0.1587
J_1	Alibi	-	-	-	-	-
J_2	DiCE	**0.0026**	0.4000	0.2000	0.1908	0.0598
J_2	Proposal	0.0060	**0.2041**	**0.1069**	**0.0948**	**0.0414**
J_2	Alibi	-	-	-	-	-
J_3	DiCE	**1.0000**	**4.0000**	**3.0000**	**2.5188**	**0.5303**
J_3	Proposal	**1.0000**	5.0000	**3.0000**	2.9439	0.7547
J_3	Alibi	-	-	-	-	-

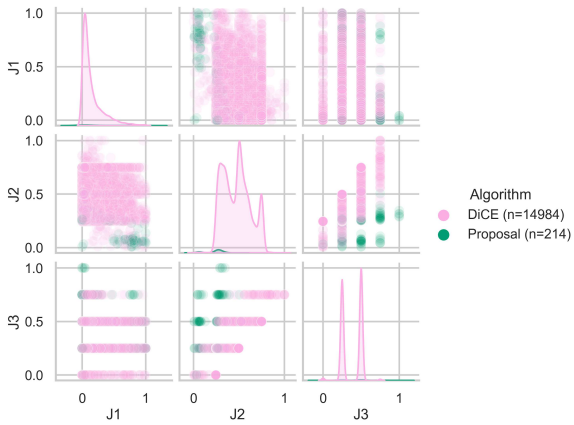

Fig. 3 Pair plot with the counterfactuals found for the COMPAS dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

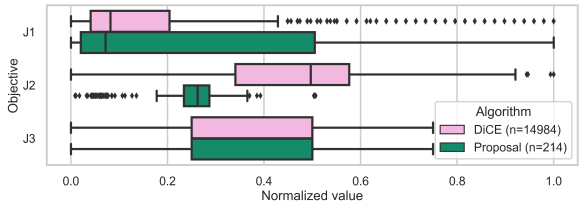

Fig. 4 Boxplot containing the counterfactuals found for the COMPAS dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

5.2.3 Discussion

DiCE was able to generate a large number of valid counterfactuals. However, its counterfactuals were usually farther to the original individual as opposed to the counterfactuals generated by the

proposed algorithm, as shown by the difference in their distribution considering J_2.

An example was the individual 5711. His original features were as follows:

- Age: 21;
- Charge degree: F;
- Race: African-American;
- Age category: Less than 25;
- Score: High;
- Gender: Male;
- Priors count: 0;
- Days before screening arrest: 0;
- Decile score: 9;
- Is recidivist: true;
- Class: true.

For this individual, 100 unique counterfactuals were found by DiCE without a ranking. On the other hand, the proposed method gave a ranked list by TOPSIS of only three counterfactuals, as follows:

- The first counterfactual modified the priors count to 1 and the number of days before screening arrest to -12;
- The second counterfactual modified the priors count to 2, the number of days before screening arrest to -1 and the decile score to 8;
- The third counterfactual modified the priors count to 2, the number of days before screening arrest to -11 and the decile score to 8.

Fig. 5 shows the comparison of the results of both algorithms. While the results for J_2 are better for the proposed method, for the other two objectives DiCE performed better. The difference was mainly on the *is recidivist* attribute: DiCE found that we could generate valid counterfactuals by modifying the value on this attribute only. On the other hand, the proposed method could not find it within the stopping criteria provided (i.e., the maximum number of generations and population size).

5.3 German

The German dataset comprises 1000 instances and 10 attributes [53]. Each instance represents a person requesting credit with features such as age, credit amount, housing conditions and credit purpose. In addition, six features were identified as categorical, three as numeric, and the remaining

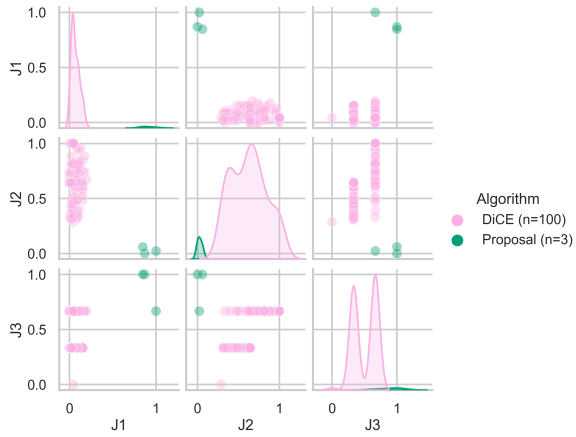

Fig. 5 Pair plot with the counterfactuals found for the individual 979 in the COMPAS dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

feature measures whether that person should have bad credit or not.

5.3.1 Counterfactual generation

Following the same logic of the other two classification datasets, the trained ML model was a scikit-learn pipeline [49] with missing value imputation, scaling and a random forest classifier trained on the modified dataset.

After training the ML model, 150 individuals from the evaluation dataset were randomly chosen. The trained model had the following confusion matrix based on these individuals:

- 33 true positives, 34 false negatives;
- 57 true negatives, 26 false positives.

Similarly to the other datasets, for each one of these 150 individuals, the three counterfactual generation techniques were tasked to find up to 100 valid counterfactuals. In order to test all three implementations, all their default values remained unchanged except for the following parameters in order to ensure any implementation can find a counterfactual in up to 5 minutes in a desktop with an Intel© i9-9900K processor, 64 GB DDR4 RAM:

- DiCE: no changes.
- Proposal:
  Maximum number of generations: 100;
  Population size: 100.
- Alibi:

Number of iterations to adjust the constant scaling the attack loss term: 3;
Maximum number of iterations: 200.

5.3.2 Results

All three algorithms were able to find valid counterfactuals considering the scope of the 150 randomly selected individuals from the evaluation dataset. The descriptive statistics are shown in Table 5.

Table 5 Results for the German dataset considering 150 individuals for DiCE (n=14027), the proposal of this paper (n=2688) and Alibi (n=168). The best values for each objective are shown in bold.

Objective	Algorithm	Minimum	Maximum	Median	Mean	Std. Dev.
J_1	DiCE	**0.0000**	**0.4900**	0.3900	0.3743	0.0921
J_1	Proposal	**0.0000**	**0.4900**	**0.2200**	**0.2232**	0.1278
J_1	Alibi	0.3000	**0.4900**	0.4500	0.4388	**0.0492**
J_2	DiCE	**0.0007**	0.6180	0.1481	0.1527	0.0830
J_2	Proposal	0.0013	0.4609	0.0747	0.0955	0.0825
J_2	Alibi	0.0011	**0.2515**	**0.0199**	**0.0564**	**0.0669**
J_3	DiCE	**1.0000**	6.0000	**2.0000**	1.9393	0.7148
J_3	Proposal	**1.0000**	5.0000	3.0000	2.6127	0.9435
J_3	Alibi	**1.0000**	**4.0000**	**2.0000**	**1.6429**	**0.6946**

The distribution of these counterfactuals are also shown in the Fig. 6 as a pairplot. It is desirable to have counterfactuals as close as possible to zero considering all three objectives since they would represent scenarios where the smallest changes in the number of variables and in the scale of the changes done within the variables would result in predictions with a large percentage of probability of being of the inverse class according to the base ML model.

The distribution of these counterfactuals is also shown in Fig. 6 as a pair plot. It is desirable to have counterfactuals as close as possible to zero considering all three objectives since they would represent scenarios where the slightest changes in the number of variables and the scale of the changes done within the variables would result in predictions with a large percentage of probability of being of the inverse class according to the base ML model.

The boxplot in Fig. 7 also highlights the distribution of these changes. According to the Mann-Whitney U test [50], all the distributions are not equal considering each algorithm and objective.

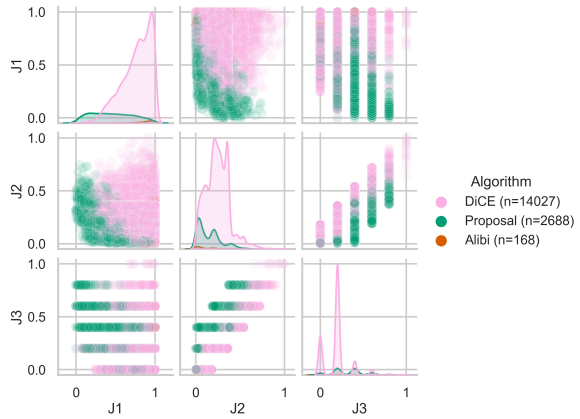

Fig. 6 Pair plot with the counterfactuals found for the German credit dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

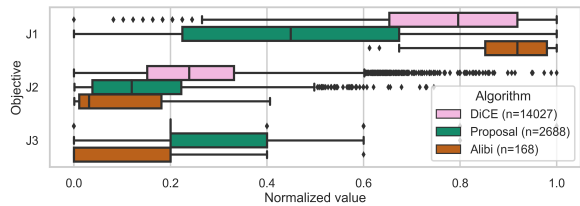

Fig. 7 Boxplot containing the counterfactuals found for the German credit dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

5.3.3 Discussion

While all the techniques could generate counterfactuals, the proposed method found suitable solutions with a high probability of being a member of the inverted class as evidenced in J_1 while maintaining small changes within the variables, as shown by J_2. Alibi had better overall results in J_2 and J_3 – however, it impacted J_1.

Another point to be highlighted is the presence of the ranking to aid in the decision making. As an example, let us consider individual 11, which initially had the following features:

- Age: 24;
- Gender: female;
- Job: category 2;
- Housing: rent;
- Saving accounts: little;
- Checking account: little;
- Credit amount: 4308;
- Duration: 48;

Table 6 Counterfactuals found by the proposed method for the individual 11 in the German dataset and ranked by TOPSIS with equal weights considering the three objectives. Cells marked with a hyphen ("-") mean no changes from the original values. *quite* means *quite rich* and *mod.* means *moderate*. The columns *Sex*, *Job*, *Housing*, and *Purpose* are hidden since the algorithm proposed no changes.

Rank	Age	Saving accounts	Checking account	Credit amount	Duration	J_1	J_2	J_3
1	27	-	-	-	21	0.3100	0.0516	0.2222
2	27	-	-	-	19	0.2800	0.0550	0.2222
3	28	-	-	-	27	0.3800	0.0436	0.2222
4	28	-	-	-	30	0.4100	0.0385	0.2222
5	28	-	-	-	40	0.4500	0.0217	0.2222
6	28	-	-	-	21	0.2900	0.0537	0.2222
7	28	-	-	-	26	0.3700	0.0453	0.2222
8	28	-	-	5880	40	0.4100	0.0314	0.3333
9	28	-	-	4371	47	0.4900	0.0103	0.3333
10	28	-	-	4235	21	0.2800	0.0541	0.3333
11	27	-	-	2628	16	0.2100	0.0704	0.3333
12	28	rich	-	-	-	0.2100	0.1193	0.2222
13	28	-	-	-	19	0.2700	0.0571	0.2222
14	28	rich	-	-	42	0.1600	0.1294	0.3333
15	28	quite	-	-	27	0.1200	0.1547	0.3333
16	28	rich	-	4405	47	0.1800	0.1216	0.4444
17	28	-	-	4235	27	0.3700	0.0440	0.3333
18	28	-	-	5251	40	0.4200	0.0275	0.3333
19	28	quite	-	-	26	0.1100	0.1564	0.3333
20	28	rich	-	4405	25	0.0900	0.1587	0.4444
21	28	-	-	4405	47	0.4800	0.0105	0.3333
22	28	-	-	4235	19	0.2600	0.0575	0.3333
23	39	quite	mod.	4405	40	0.0300	0.2672	0.5556
24	39	rich	mod.	-	40	0.0600	0.2666	0.4444
25	27	quite	-	2686	19	0.0700	0.1761	0.4444
26	26	-	-	4477	21	0.3400	0.0506	0.3333

- Purpose: business;
- Class: false.

DiCE had as an output a table with 100 counterfactuals without a ranking. Alibi found no counterfactuals. The proposed method found 26 counterfactuals, as seen in Table 6. The ranking first suggested the counterfactuals that best balanced the three objectives.

6 Regression

In this section, we test the proposed algorithm against DiCE. Alibi does not provide an implementation for counterfactual generation covering regression problems. Two benchmark datasets were used to evaluate the performance of the algorithms: the California housing dataset and the Abalone age dataset.

Similarly to the classification problems, all regression datasets were sampled through random sampling, with 70% of each dataset being used for model training and the remaining 30% for evaluation and counterfactual generation.

All datasets were trained on a random forest regressor. This model was chosen because it is not very explainable [28] and because it is widely used [47].

6.1 California housing dataset

The California housing dataset is composed of 20640 instances and nine attributes [54]. Each instance represents a house in California, and each attribute represents information such as latitude and longitude, the population in the region, the number of households, median income, and the number of rooms and bedrooms. All features were identified as numeric. The target feature measures the median house value – measured in USD – for households within a block.

6.1.1 Counterfactual generation

The trained ML model was a scikit-learn [49] pipeline with a scaler and a random forest regressor with its default settings and using the training dataset.

In order to test the counterfactual generation, 75 random individuals with the target feature below the median in the training dataset were selected out of the evaluation dataset, and 75 random individuals with the target feature above the median were also selected with a total of 150 randomly chosen individuals. Since this is a regression dataset, we determined a valid counterfactual as being as follows:

- When the individual has its target feature below the median, its counterfactual should have a predicted value between the median and the maximum value of the target of the training dataset – ideally, on the third quartile;
- When the individual has its target feature above the median, its counterfactual should have a predicted value between the minimum and the median of the target of the training dataset – ideally, on the first quartile.

While DiCE kept its default settings except for the number of counterfactuals per individual (set to 100), the proposed method was adjusted to have its maximum number of generations and the population size set to 100.

6.1.2 Results

The two algorithms were able to find counterfactuals considering the 150 randomly selected individuals. The descriptive statistics are shown in Table 7.

Table 7 Results for the California housing dataset considering 150 individuals for DiCE (n=15000) and the proposal of this paper (n=2715). The best values for each objective are shown in bold.

Objective	Algorithm	Minimum	Maximum	Median	Mean	Std. Dev.
J_1	DiCE	**0.0000**	2.3528	0.4827	0.6896	0.6515
J_1	Proposal	**0.0000**	**0.8316**	**0.1149**	**0.2168**	**0.2390**
J_2	DiCE	**0.0000**	0.3599	0.0813	0.0871	0.0516
J_2	Proposal	**0.0000**	**0.3013**	**0.0172**	**0.0234**	**0.0233**
J_3	DiCE	**1.0000**	**5.0000**	**2.0000**	**1.7643**	**0.7096**
J_3	Proposal	**1.0000**	**5.0000**	**3.0000**	2.9448	1.3351

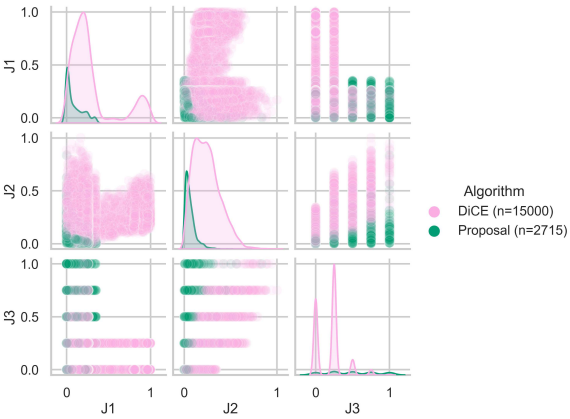

Fig. 8 Pair plot with the counterfactuals found for the California housing dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

The distribution of these counterfactuals is better displayed in Fig. 8. Since it is a regression problem, J_1 calculates the difference between the predicted and the target values instead of the class probabilities. It was considered for classification problems. However, the definitions of J_2 and J_3 remain unchanged. This distribution is also shown in the format of a boxplot in Fig. 9. The distribution of the populations found for both algorithms is not equal as the null hypothesis had been rejected for the three objectives through the Mann-Whitney U test [50].

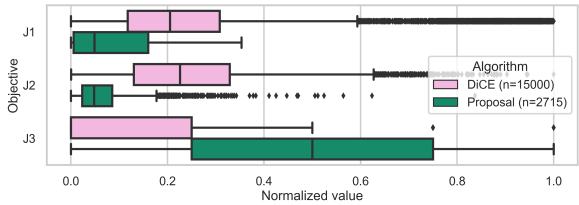

Fig. 9 Boxplot containing the counterfactuals found for the California housing dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

6.1.3 Discussion

The proposed method was able to generate counterfactuals closer to the desired value in J_1 than the ones found by DiCE. Furthermore, its counterfactuals required were also closer to the original individuals, as noticed by J_2. However, to achieve this, the proposed method had to change more variables, as shown by the greater values in J_3. In real scenarios, it means the proposed method changed more variables, but these changes were more subtle as opposed to DiCE. DiCE changed fewer variables in this scenario, but the changes were more drastic. An example of this is evidenced by the counterfactuals generated by the individual 9. Its original values were as follows:

- Median income: 4.1518;
- House age: 22;
- Average number of rooms: 5.663073;
- Average number of bedrooms: 1.075472;
- Population: 1551;
- Average occupation: 4.180593;
- Latitude: 32.58;
- Longitude: -117.05.

One of the counterfactuals found by DiCE with the objective values close to the median values shown in Table 7 had the following characteristics:

- $J_1 = 0.4982$; $J_2 = 0.0928$; $J_1 = 2.0000$;
- Median income: 6.1000 (from 4.1518, range: $[0.4999, 15.0001]$);
- Average bedrooms: 21.5000 (from 1.075472, range: $[0.3333, 34.0667]$).

The counterfactuals generated by the proposed method, which had its values close to the median values, had the following characteristics:

- $J_1 = 0.0117$; $J_2 = 0.0123$; $J_1 = 5.0000$:
  Median income: 4.0000 (from 4.1518, range: $[0.4999, 15.0001]$);
  Average rooms: 6.0000 (from 5.6631, range: $[0.8462, 141.9091]$);
  Average bedrooms: 1.0000 (from 1.075472, range: $[0.3333, 34.0667]$);
  Population: 2133 (from 1551, range: $[3, 35682]$);
  Average occupation: 2.0000 (from 4.180593, range: $[0.6923, 1243.3333]$).
- $J_1 = 0.0143$; $J_2 = 0.0232$; $J_1 = 2.0000$:
  Median income: 6.0000 (from 4.1518, range: $[0.4999, 15.0001]$);
  Population: 3627 (from 1551, range: $[3, 35682]$).

While the changes found by DiCE mean fewer variables are changed, these changes are proportionally greater than those found by the proposed method, as evidenced by J_2.

6.2 Abalone

The Abalone age dataset comprises 4177 instances and nine attributes [55]. Each instance represents a single abalone, and each attribute measures its physical features such as shell weight, diameter and length. One attribute was identified as categorical (sex); all the remaining features were numeric. The target feature measures the number of rings found in its shell – the count of the number of rings is used to measure its age.

6.2.1 Counterfactual generation

Similar to the California housing dataset, this dataset also used a scikit-learn [49] pipeline with a scaler and a random forest regressor with its default settings for training an ML model on its training dataset.

The same rationale to select individuals from the evaluation dataset also applied: 75 random individuals with the target feature below the median and 75 random individuals with the target feature above the median were selected. Furthermore, the determination of the desired range for the counterfactuals target was also the same: when the individual had its target feature below the median, its counterfactual should have a predicted value between the median and the maximum value of the target of the training dataset – ideally, on the third quartile; and when the individual has its target feature above the median, its counterfactual should have a predicted value between the minimum and the median of the target of the training dataset – ideally, on the first quartile.

In order to generate the counterfactuals, DiCE kept its default settings except for the number of counterfactuals per individual (set to 100). The proposed method was adjusted to have its maximum number of generations and the population size set to 100.

6.2.2 Results

For this dataset, both algorithms were capable of successfully generating counterfactuals. The descriptive statistics are shown in the Table 8.

Table 8 Results for the abalone dataset considering 150 individuals for DiCE (n=14998) and the proposal of this paper (n=848). The best values for each objective are shown in bold.

Objective	Algorithm	Minimum	Maximum	Median	Mean	Std. Dev.
J_1	DiCE	**0.0000**	8.5900	0.9600	2.2813	2.6770
J_1	Proposal	**0.0000**	**7.5000**	**0.2400**	**0.8860**	**1.5520**
J_2	DiCE	**0.0000**	**0.5200**	**0.0991**	**0.1154**	**0.0790**
J_2	Proposal	0.0006	0.7104	0.1450	0.1817	0.1463
J_3	DiCE	**0.0000**	7.0000	**2.0000**	**2.2850**	**1.2413**
J_3	Proposal	1.0000	**5.0000**	**2.0000**	2.6014	1.2490

The distribution of the counterfactuals can be seen in Fig. 10. It is desirable to have counterfactuals as close to zero as possible considering the three objectives at the same time: counterfactuals meeting these criteria simultaneously would indicate scenarios where the least amount of variables is changed and to the slightest degree as possible while producing predictions as close as possible to the expected value.

The same distribution is seen as a boxplot in Fig. 11 comparing the performance of the two algorithms within the three objectives. The three objectives were scaled in order to analyse the results better. The distribution of the populations found for both algorithms is not equal as the

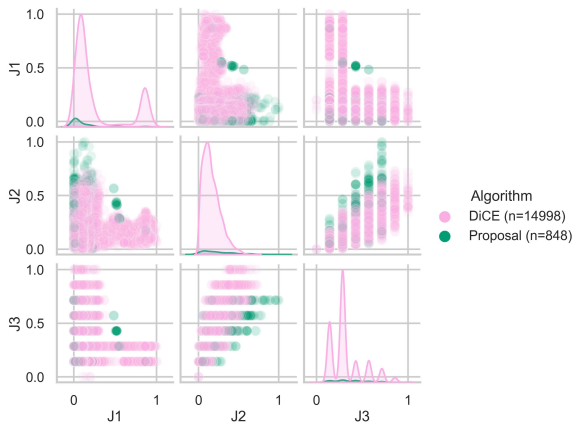

Fig. 10 Pair plot with the counterfactuals found for the abalone dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

null hypothesis had been rejected for the three objectives through the Mann-Whitney U test [50].

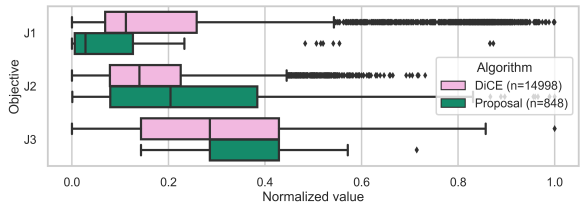

Fig. 11 Boxplot containing the counterfactuals found for the abalone dataset with the scaled results within the $[0, 1]$ range. It is desired to have the values as close to zero as possible.

6.2.3 Discussion

Similarly to the California housing dataset, the proposed method could also find counterfactuals closer to the desired value in J_1 than the ones found by DiCE. However, in some scenarios, it came at the expense of J_2 and J_3.

As an example, let us consider the individual 2377. Originally it had the following features:

- Sex: I;
- Length: 0.535;
- Diameter: 0.41;
- Height: 0.155;
- Whole weight: 0.6315;
- Shucked weight: 0.2745;
- Viscera weight: 0.1415;
- Shell weight: 0.1815.

A counterfactual found by DiCE with its objective values close to the medians in Table 8 had the following characteristics:

- $J_1 = 0.9700$ $J_2 = 0.0932$ $J_3 = 2.0000$:
  Length: 0.600 (from 0.535; range: $[0.075, 0.815]$);
  Shucked weight: 0.9600 (from 0.1415; range: $[0.001, 1.488]$).

On the other hand, a counterfactual found by the proposed method with its objective values also close to its medians had the following characteristics:

- $J_1 = 0.2500$ $J_2 = 0.1476$ $J_3 = 2.0000$:
  Sex: F (from I);
  Shell weight: 0.0001 (from 0.1415; range: $[0.001, 1.005]$).

While both only changed two features, the proposed method changed a categorical attribute. According to the Gower distance calculation in Eq. 9, it does result in a larger J_2. In perspective, the proposed method found 848 counterfactuals: 270 of which (31.84%) modified the categorical attribute (sex). DiCE found 14998 counterfactuals: 2807 (18.72%) modified the categorical attribute. Since DiCE had proportionally fewer counterfactuals where the categorical column was changed, its J_2 was less impacted.

7 Conclusion and future work

This paper proposed a counterfactual generation method using a constrained multi-objective optimisation design for classification and regression problems. To ensure valid counterfactuals are generated, three objectives were determined to reduce the number of variables changed, the amount of the changes within the variables, and the difference between the prediction of a counterfactual and its desired value. These objectives are constrained to acceptable prediction ranges, the number of variables that may be modified, and the allowed lower and upper bounds. Since multiple counterfactuals are generated from an individual, we proposed using TOPSIS to rank and choose the most suitable counterfactuals.

The proposed method was tested against two other popular counterfactual generation methods available in Python – DiCE and Alibi – in three classification datasets and two regression datasets.

The proposed method generated counterfactuals for these datasets, outperforming the other methods in at least one objective per dataset. In addition, the proposed method generated counterfactuals for both numeric and categorical features and ranked them considering the three objectives simultaneously.

However, while the adoption of the Gower distance enabled the comparison of counterfactuals with mixed features, it caused heavier penalties on categorical features as evidenced by the US Census and the abalone dataset. We plan to focus on changing the distance metric used to evaluate the generated counterfactuals in future research. Furthermore, we expect to test this method on classification datasets with multiple classes.

Declarations

Conflict of Interest. The authors have no conflict of interest to declare.

Acknowledgments. This study was financed in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), and the Fundação Araucária (FAPPR) - Brazil - Finance Codes: 310079/2019-5-PQ2, 437105/2018-0-Univ, 51432/2018-PPP and PRONEX-042/2018.

Data Availability. The source code used to implement the proposed method and to test it against other libraries is available in the GitHub repository at https://github.com/wmonteiro92/cgmoc.

References

[1] Ge, Z., Song, Z., Ding, S.X., Huang, B.: Data mining and analytics in the process industry: The role of machine learning. IEEE Access **5**, 20590–20616 (2017)

[2] Bishop, C.M.: Pattern Recognition and Machine Learning. Information Science and Statistics. Springer, New York, United States of America (2016)

[3] Witkowski, K.: Internet of things, big data, Industry 4.0 – innovative solutions in logistics and supply chain management. Procedia Engineering **182**, 763–769 (2017)

[4] Mourtzis, D., Vlachou, E., Milas, N.: Industrial big data as a result of IoT adoption in manufacturing. Procedia CIRP **55**, 290–295 (2016)

[5] Bi, Z.: Embracing internet of things (IoT) and big data for industrial informatics. Enterprise Information Systems **11**(7), 949–951 (2017). https://doi.org/10.1080/17517575.2016.1258734

[6] Lwakatare, L.E., Raj, A., Crnkovic, I., Bosch, J., Olsson, H.H.: Large-scale machine learning systems in real-world industrial settings a review of challenges and solutions. Information and Software Technology, 106368 (2020)

[7] Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.-R.: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning vol. 11700. Springer, Cham, Switzerland (2019)

[8] Oh, S.J., Schiele, B., Fritz, M.: Towards reverse-engineering black-box neural networks. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 121–144. Springer, Cham, Switzerland (2019)

[9] Samek, W., Müller, K.-R.: Towards explainable artificial intelligence. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 5–22. Springer, Cham, Switzerland (2019)

[10] Hansen, L.K., Rieger, L.: Interpretability in intelligent systems – a new concept? In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 41–49. Springer, Cham, Switzerland (2019)

[11] Weller, A.: Transparency: motivations and challenges. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, pp. 23–40. Springer, Cham, Switzerland (2019)

[12] Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: Automated decisions and the

GDPR. Harvard Journal of Law & Technology **31**, 841 (2017)

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

[14] Johansson, F.D., Sontag, D.: Cfrnet. https://github.com/clinicalml/cfrnet

[15] Klaise, J., Van Looveren, A., Vacanti, G., Coca, A.: Alibi: Algorithms for Monitoring and Explaining Machine Learning Models. https://github.com/SeldonIO/alibi

[16] Mothilal, R.K., Sharma, A., Tan, C.: Diverse Counterfactual Explanations (DiCE) For ML. https://github.com/interpretml/DiCE

[17] Arya, V., Bellamy, R.K.E., Chen, P.-Y., Dhurandhar, A., Hind, M., Hoffman, S.C., Houde, S., Liao, Q.V., Luss, R., Mojsilović, A., Mourad, S., Pedemonte, P., Raghavendra, R., Richards, J., Sattigeri, P., Shanmugam, K., Singh, M., Varshney, K.R., Wei, D., Zhang, Y.: One explanation does not fit all: A toolkit and taxonomy of AI explainability techniques. arXiv (2019)

[18] Reynoso-Meza, G., Blasco, X., Sanchis, J., Martínez, M.: Controller tuning using evolutionary multi-objective optimisation: current trends and applications. Control Engineering Practice **28**, 58–73 (2014)

[19] Coello, C.A.C., Lamont, G.B., Van Veldhuizen, D.A., *et al.*: Evolutionary Algorithms for Solving Multi-objective Problems vol. 5. Springer, New York, United States of America (2007)

[20] Coello, C.A.C., Lamont, G.B.: Applications of Multi-objective Evolutionary Algorithms vol. 1. World Scientific, Singapore (2004)

[21] Seada, H., Deb, K.: U-NSGA-III: a unified evolutionary optimization procedure for single, multiple, and many objectives: proof-of-principle results. In: International Conference on Evolutionary Multi-criterion Optimization, pp. 34–49 (2015). Springer

[22] Sharma, S., Henderson, J., Ghosh, J.: CERTIFAI: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models. arXiv (2019)

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

[24] Dandl, S., Molnar, C., Binder, M., Bischl, B.: Multi-objective counterfactual explanations. In: International Conference on Parallel Problem Solving from Nature, pp. 448–469 (2020). Springer

[25] Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining explanations: An overview of interpretability of machine learning. In: 2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA), pp. 80–89 (2018). IEEE

[26] Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., *et al.*: Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion **58**, 82–115 (2020)

[27] Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608 (2017)

[28] Adadi, A., Berrada, M.: Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). IEEE Access **6**, 52138–52160 (2018)

[29] Artelt, A., Hammer, B.: On the computation of counterfactual explanations – a survey. ArXiv **abs/1911.07749** (2019)

[30] Barredo-Arrieta, A., Del Ser, J.: Plausible

counterfactuals: Auditing deep learning classifiers with realistic adversarial examples. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–7 (2020). IEEE

[31] Rao, S.S.: Engineering Optimization: Theory and Practice. John Wiley & Sons, Hoboken, United States of America (2019)

[32] Johansson, F., Shalit, U., Sontag, D.: Learning representations for counterfactual inference. In: International Conference on Machine Learning, pp. 3020–3029 (2016)

[33] Shalit, U., Johansson, F.D., Sontag, D.: Estimating individual treatment effect: generalization bounds and algorithms. In: International Conference on Machine Learning, pp. 3076–3085 (2017)

[34] Looveren, A.V., Klaise, J.: Interpretable counterfactual explanations guided by prototypes, 650–665 (2021). Springer

[35] Meza, G.R., Ferragud, X.B., Saez, J.S., Durá, J.M.H.: Controller Tuning with Evolutionary Multiobjective Optimization: A Holistic Multiobjective Optimization Design Procedure vol. 85. Springer, Cham, Switzerland (2016)

[36] Mattson, C.A., Messac, A.: Pareto frontier based concept selection under uncertainty, with visualization. Optimization and Engineering **6**(1), 85–115 (2005)

[37] Boussaïd, I., Lepagnot, J., Siarry, P.: A survey on optimization metaheuristics. Information Sciences **237**, 82–117 (2013). https://doi.org/10.1016/j.ins.2013.02.041. Prediction, Control and Diagnosis using Advanced Neural Computations

[38] UCI Machine Learning Repository: Diabetes Data Set. https://archive.ics.uci.edu/ml/datasets/diabetes

[39] Gower, J.C.: A general coefficient of similarity and some of its properties. Biometrics, 857–871 (1971)

[40] Tuerhong, G., Kim, S.B.: Gower distance-based multivariate control charts for a mixture of continuous and categorical variables. Expert systems with applications **41**(4), 1701–1707 (2014)

[41] Blank, J., Deb, K.: Pymoo: Multi-objective Optimization in Python. IEEE Access, 1–1 (2020)

[42] Yoon, K., Hwang, C.L.: Multiple attribute decision making: Methods and applications. Berlin: Springer Verlag (1981)

[43] Zyoud, S.H., Fuchs-Hanusch, D.: A bibliometric-based survey on AHP and TOPSIS techniques. Expert systems with applications **78**, 158–181 (2017)

[44] Pessach, D., Shmueli, E.: Algorithmic fairness. arXiv (2020)

[45] Von Eye, A., Clogg, C.C.: Categorical Variables in Developmental Research: Methods of Analysis. Elsevier, San Diego, United States of America (1996)

[46] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research **16**, 321–357 (2002)

[47] Speiser, J.L., Miller, M.E., Tooze, J., Ip, E.: A comparison of random forest variable selection methods for classification prediction modeling. Expert Systems with Applications **134**, 93–101 (2019). https://doi.org/10.1016/j.eswa.2019.05.028

[48] UCI Machine Learning Repository: Census Income Data Set. https://archive.ics.uci.edu/ml/datasets/census+income

[49] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

[50] Fay, M.P., Proschan, M.A.: Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. Statistics Surveys **4**, 1–39 (2010). https://doi.org/10.1214/09-SS051

[51] ProPublica: COMPAS: Data and Analysis for 'machine Bias'. https://github.com/propublica/compas-analysis/

[52] Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for dna microarrays. Bioinformatics **17**(6), 520–525 (2001)

[53] UCI Machine Learning Repository: Statlog (German Credit Data) Data Set. https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

[54] Pace, R.K., Barry, R.: Sparse spatial autoregressions. Statistics & Probability Letters **33**(3), 291–297 (1997)

[55] UCI Machine Learning Repository: Abalone Data Set. https://archive.ics.uci.edu/ml/datasets/abalone