Information Fusion 63 (2020) 196–207

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Random forest explainability using counterfactual sets

Rubén R. Fernández, Isaac Martín de Diego, Víctor Aceña, Alberto Fernández-Isabel*, Javier M. Moguerza

Rey Juan Carlos University, Data Science Laboratory, C/ Tulipán, s/n, Móstoles, 28933, Spain

ARTICLE INFO

Keywords:
Explainable machine learning
Counterfactual sets
Counterfactual
Information fusion
Random forest
Decision tree

ABSTRACT

Nowadays, Machine Learning (ML) models are becoming ubiquitous in today's society, supporting people with their day-to-day decisions. In this context, Explainable ML is a field of Artificial Intelligence (AI) that focuses on making predictive models and their decisions interpretable by humans, enabling people to trust predictive models and to understand the underlying processes. A counterfactual is an effective type of Explainable ML technique that explains predictions by describing the changes needed in a sample to flip the outcome of the prediction. In this paper, we introduce counterfactual sets, an explanation approach that uses a set of counterfactuals to explain a prediction rather than a single counterfactual, by defining a sub-region of the feature space where the counterfactual holds. A method to extract counterfactual sets from a Random Forest (RF), the *Random Forest Optimal Counterfactual Set Extractor* ($RF-OCSE$), is presented. The method is based on a partial fusion of tree predictors from a RF into a single Decision Tree (DT) using a modification of the *CART* algorithm, and it obtains a counterfactual set that contains the optimal counterfactual. The proposal is validated through several experiments against existing alternatives on ten well-known datasets by comparing the percentage of valid counterfactuals, distance to the factual sample, and counterfactual sets quality.

1. Introduction

As digital devices are becoming ubiquitously present, we are shifting towards a data-driven society, where most decisions are made using data [1,2]. Within this context, ML has been used to automate tasks and support decision-makers. However, ML models are limited in projects and domains where predictions might be biased towards minorities, might cost lives, or when it is interesting to understand the inner working of the processes [3]. Explainable ML aims to solve these limitations by making ML models and their predictions interpretable by humans.

Counterfactual explanations are one of the most widely used techniques in Explainable ML because they resemble human-thinking [4,5]. They focus on explaining individual predictions rather than the whole model, which might be difficult because of the high complexity of the decision surface. An explanation is expressed as the changes needed in the sample to be explained, often referred to as factual sample, to flip the sample prediction. For instance, given a sample in the transfusion dataset [6] with values *Recency* = 2 months, *Frequency* = 50, *Monetary* = 12,500 and *Time* = 98, classified as donation, a counterfactual explanation might suggest that if the *Monetary* was 1,125 or the *Recency* was 6.5 then it would be classified as no donation [7].

This paper introduces a new explanation technique called counterfactual sets. Counterfactual sets explain predictions using a sub-region of the feature space where the counterfactual holds rather than a single counterfactual. This new explanation technique is not specific to a particular ML model or domain. In this work, for the sake of simplicity, they are restricted to be simply connected. Counterfactual sets are represented without enumerating their elements to make them easier to interpret and to allow having sets with infinite elements. This proposal uses a counterfactual set representation based on feature ranges with the same intuitive interpretation as rules in a DT. A counterfactual set for the previous example might be $1,000 \leq Monetary \leq 1,125$ or $Recency \geq 6.5$.

Counterfactual sets have the same explainability properties as counterfactuals, as they are made of counterfactuals. Besides, they provide the context of the changes. This context helps to apply and to understand counterfactuals, especially when they involve high-variability variables (e.g., speed or temperature) or measurements (e.g., the height or weight of a person) because it is easier to keep these variables in a range than in a constant value. On the other hand, they could also help the user to measure the quality of the explanation or to diagnose a ML model.

* Corresponding author.
E-mail addresses: ruben.rodriguez@urjc.es (R.R. Fernández), isaac.martin@urjc.es (I. Martín de Diego), victor.acena@urjc.es (V. Aceña), alberto.fenandez.isabel@urjc.es (A. Fernández-Isabel), javier.moguerza@urjc.es (J.M. Moguerza).
URL: http://www.datasciencelab.es (R.R. Fernández)

https://doi.org/10.1016/j.inffus.2020.07.001
Received 2 March 2020; Received in revised form 30 June 2020; Accepted 1 July 2020
Available online 3 July 2020

In addition, we propose a method to extract counterfactual sets from a RF with optimality guarantees. This method is called *Random Forest Optimal Counterfactual Set Extractor* ($RF-OCSE$). First, it converts a RF into an equivalent DT using a modification of the *CART* algorithm [8] through a partial fusion of the RF tree predictors. The rules of the tree predictors are considered observations, and the splits are the features. The stop condition is modified to stop when there is only one rule of each tree predictor of the RF. The leaves are defined as the aggregation of these rules as in a RF. The optimal counterfactual set is generated using the rule which classifies the optimal counterfactual according to a given metric. The method is later improved by integrating the counterfactual set extraction procedure into the fusion method. As a result, the method only requires to convert a RF partially.

The proposal is evaluated in ten well-known datasets in terms of percentage of valid counterfactuals, distance to the factual sample, and counterfactual set quality. The experiments allow assessing the differences in performance between methods with optimality guarantees and approximate methods. Besides, counterfactual set explanations are compared to counterfactual explanations to evaluate their explainability, and they are compared with the pseudo-counterfactual sets extracted by alternative methods.

The remainder of the paper is organized as follows. Section 2 details the relevant work related to the proposal. Section 3 introduces counterfactual sets explanation technique. Section 4 describes the proposed method to extract counterfactual sets from a RF. Section 5 addresses the experiments to validate the proposal. Finally, Section 6 concludes and provides further research directions.

2. Related work

There are several types of explanations in Explainable ML that emphasize different aspects of ML models and have different meanings and interpretations. These types of explanations can be categorized based on their scope, origin, and applicability. An analysis of a taxonomy for Explainable Artificial Intelligence can be consulted in [9], including a complete literature review and current trends regarding ML explainability techniques (for an intuitive description of Interpretable ML techniques, see for instance [10]).

The scope of the explanations can be either global or local. Global explanations aim to understand the whole ML model at once, whereas local explanations focus on a small region around a sample of interest. The origin of the explanations can be intrinsic or post-hoc. An explanation is intrinsic when the ML model is considered transparent, and it can be interpreted due to its simple structure (for instance, Linear Regression and DT). On the other hand, post-hoc explanation techniques extract explanations from trained ML models. The applicability of the explanations can be model agnostic, when they apply to a set of ML models that meet a set of requirements (for instance, model differentiability), or model-specific when the technique is designed for a specific ML model [9,10].

The scope and origin of the explanations are often correlated. Global explanations are usually limited to transparent ML models because as their complexity grows, the interpretability is reduced. Global explanations can also be obtained using post-hoc techniques, for example, feature importance or global surrogates that approximate a ML model using a simpler one. Local explanations are mostly extracted using post-hoc techniques when the complexity of the ML model hardly allows us to understand the whole ML model, or we are only interested in a sub-region of the feature space. Examples of post-hoc local explanations techniques are counterfactuals and influential observations. Local explanations can also be obtained from transparent ML models by understanding how the model internals affect a given sample (e.g., coefficients in a Linear Regression) [9,10].

The paper scope is limited to example-based explanations. These techniques use other samples, previously defined or generated, to provide a global or local explanation. Global example-based explanation techniques summarize a model using a set of samples (e.g., influential observations in the model construction). Instances of local example-based techniques are counterfactuals, influential observations for a given sample [11], and similarity-based techniques.

Counterfactuals are a type of local example-based post-hoc explanation. They explain the reason why a ML model has predicted a class P rather than class Q for a sample $\mathbf{x}$ [12]. The sample $\mathbf{x}$ is often referred to as the factual sample, and P and Q are the factual and foil classes, respectively. A counterfactual is usually stated as the set of features changes to $\mathbf{x}$ that would make the model change the prediction to Q. An instance of counterfactual could be "If you had one credit card, then the credit score system would have approved the credit". In this case, the user has two or more credit cards and wants to know why the credit has been denied. Counterfactual explanations are an effective method to explain ML model predictions, as they resemble human-thinking and are easy to understand when only a few changes are required [9,10].

In most cases, the number of counterfactuals for a sample is greater than one, and they are selected using a metric to provide the most relevant explanation. This metric measures the distance from the counterfactual to the factual sample. The desirable properties of the counterfactuals are sparsity (few changes) and closeness to the sample. Therefore, distances that encourage sparsity are used to evaluate the quality of the explanations. Instances used in the literature are the $L0$, $L1$ and $L\infty$ Norms [13], the inverse of the median absolute deviation [14], a modification of the Gower distance to induce sparsity [7] or the number of changes [15]. There is no optimal metric, and different problems require different metrics that emphasize distinct aspects of the individuals. However, the metrics proposed in the literature could be used as prototypes that can be tailored for each particular problem (e.g., weights in the changes).

Counterfactual explanations are closely related to adversarial attacks [5,13]. In both cases, the goal is to find an observation similar to the observation of interest but with different predicted class. However, their purpose is different. In adversarial attacks, the purpose is to make the model misclassify the sample. Further, in most cases, the difference between the observation of interest and the adversarial attack is not easily perceptible by humans (e.g., one-pixel attacks in images [16]). On the other hand, the purpose of counterfactuals is to make the user reason about the difference between the observation of interest and the counterfactual. This difference will help the user to understand the outcome of the classifier. Therefore, even though adversarial attacks are counterfactuals, they usually would not be useful because they do not account for the explainability of the extracted sample. In counterfactuals, this explainability is achieved through distance functions that emphasize different aspects depending on the problem, or they are restricted to a region of the input space.

Counterfactual extraction methods can be categorized based on their optimality, applicability, and plausibility. Optimality is achieved when the extraction method provides guarantees on minimum distance, whereas plausibility is the ability to generate counterfactuals that are likely under the dataset distribution. *LORE* [15], *LEAFAGE* [4] and *CLEAR* [17] are examples of model-agnostic extraction techniques that can not provide optimality guarantees as they rely on approximations, and do not enforce plausibility constraints. These techniques build a local surrogate around the sample using a transparent model and then extract the counterfactuals using model-specific techniques. Other model-agnostic techniques use a gradient to extract counterfactuals [14,18], but they are not applicable when the gradient can not be calculated in the ML model (e.g., DT). Generative models have also been used to extract counterfactuals from other classification models [19,20]. In this approach, it is also possible to ensure plausibility [19], and they might or not require the gradient of the target model. *MACE* [13] is an example of model agnostic with optimality and plausibility guarantees. It represents the counterfactual extraction as a satisfiability problem. *MACE* extracts the closest counterfactual within a given arbitrary tolerance. $Feature-Tweaking$ (FT) [21] is an example of a model-specific

approach that extracts counterfactuals from tree-based ensembles using model-internals. However, *FT* does not have optimality and plausibility guarantees, and in some cases, it might not be able to return a counterfactual.

Many techniques to convert a RF into a single DT have already been proposed in the literature [22–26]. These usually rely on sampling to create an approximate surrogate representation of a RF. This approximate conversion makes them unsuitable for providing counterfactual optimality guarantees. In the case of *Forest Based Tree* (*FBT*), it uses a conversion method similar to the proposal. However, it uses approximations to be able to convert large RFs because an exact conversion becomes intractable as the number of tree predictors and their depth grow. As a result, *FBT* can not provide optimality guarantees. RF-OCSE overcomes this limitation by using a partial conversion around the factual sample without incurring in approximations.

3. Counterfactual sets

This proposal introduces the notion of counterfactual sets, an explanation technique based on counterfactuals. It uses a set of counterfactuals to explain a prediction rather than a single counterfactual. Throughout the rest of the paper, we consider $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_n\}$ as input space, where the type of each feature, $\mathcal{X}_i$, can be $\mathbb{R}$ or $\mathbb{Z}$, although other spaces may be suitable. Given a binary classification model f with inputs in $\mathcal{X}$, a factual sample $\hat{\mathbf{x}} \in \mathcal{X}$, a distance d and a radius n_r that define a neighborhood around $\hat{\mathbf{x}}$, a counterfactual set, $CS_{f,d,n_r}(\hat{\mathbf{x}})$, is defined as follows:

$$CS_{f,d,n_r}(\hat{\mathbf{x}}) = \{\mathbf{x} \in CF_f(\hat{\mathbf{x}}) \mid d(\mathbf{x}, \hat{\mathbf{x}}) < n_r\} \tag{1}$$

where $CF_f(\hat{\mathbf{x}})$ is the counterfactual space [13]. The counterfactual space is a set that contains all possible counterfactuals in the model f for the factual sample $\hat{\mathbf{x}}$. In a binary classification problem, the counterfactual space for a model f and a factual sample $\hat{\mathbf{x}}$, is defined as follows:

$$CF_f(\hat{\mathbf{x}}) = \{\mathbf{x} \in \mathcal{X} \mid f(\mathbf{x}) \neq f(\hat{\mathbf{x}})\}. \tag{2}$$

Regarding the neighborhood, its shape and size are context-aware. Thus, different neighborhoods are required depending on the domain and the region of the input space. This neighborhood could be used to enforce, for example, plausibility constraints (e.g., restrict feature changes) or diversity (e.g., extract different counterfactual sets).

In this work, for the sake of simplicity, counterfactual sets are restricted to be simply connected, and the whole input space has been considered as the predefined neighborhood (i.e., $n_r = \infty$). Thus, each counterfactual set represents a simply connected sub-region of the feature space where a counterfactual holds. In this context, the following desirable properties for counterfactual sets explanations are defined:

- **Representation**: a counterfactual set should be summarized using an interpretable representation without having to enumerate its elements. Thus, this representation aggregates the properties of the elements of the counterfactual set. As the elements of counterfactual sets are not enumerated, explanations with infinite elements are possible. Further, as counterfactual sets are restricted to be simply connected, a counterfactual set that contains more than one possible value for a numeric-real feature can not be enumerated.
- **Set coverage**: The area spawn by a counterfactual set in the input space should cover as many instances as possible from the training set. This property is the same as the coverage in *LORE* explanations. A counterfactual set with broad coverage is easier to understand as the factual sample can be contrasted with more real counterfactuals from the dataset. Given a set $S \subseteq \mathcal{X}$ (e.g., a counterfactual set), and the training set X, the set coverage of S over X is measured as follows:

$$set\text{-}coverage(S, X) = \frac{\#(S \cap X)}{\#X} \tag{3}$$

where # denotes the cardinality of a set.

The optimal counterfactual is defined as the nearest counterfactual to the factual sample in the counterfactual space [13] given a distance function d :

$$\mathbf{x}^* \in arg\ min_{\mathbf{x} \in CF_f(\hat{\mathbf{x}})} d(\mathbf{x}, \hat{\mathbf{x}}) \tag{4}$$

The optimal counterfactual might not be unique, as there might be more than one counterfactual at the same distance from the factual sample. Although this situation is not common with numeric-real variables, it is more frequent when the samples contain ordinal, integer, or categorical variables. Notice that the counterfactual is optimal in terms of the distance chosen, it does not imply that it is the best counterfactual explanation for a given factual sample.

Based on the optimal counterfactual and counterfactual set definition, the optimal counterfactual sets, $OCS_{f,d,n_r}(\hat{\mathbf{x}})$, are defined as all possible counterfactual sets that contain an optimal counterfactual within the neighborhood defined by the distance d and the radius n_r:

$$OCS_{f,d,n_r}(\hat{\mathbf{x}}) = \{CS_{f,d,n_r}(\hat{\mathbf{x}}) \mid \mathbf{x}^* \in CS_{f,d,n_r}(\hat{\mathbf{x}})\} \tag{5}$$

Notice that the distance d used to select the optimal counterfactual is the same that defines the neighborhood. Optimal counterfactual sets are not usually unique, because there may be more than one optimal counterfactual, or we can define subsets from an optimal counterfactual set that contains the optimal counterfactual. Thus, among the optimal counterfactual sets for a given model f and factual sample $\hat{\mathbf{x}}$, counterfactual sets with broader *set coverage* should be preferred.

The definition of counterfactual sets can be relaxed into pseudo counterfactual sets by allowing sets in which there are also instances from the factual class. Given a classification model f, a distance d and a radius n_r that define a neighborhood, a pseudo counterfactual set, $PCS_{f,d,n_r}(\hat{\mathbf{x}})$, is defined as follows:

$$PCS_{f,d,n_r}(\hat{\mathbf{x}}) = \{\mathbf{x} \in \mathcal{X} \mid d(\mathbf{x}, \hat{\mathbf{x}}) < n_r\} \tag{6}$$

The proportion of individuals from the factual class in the pseudo counterfactual set should be small. This proportion, which we call *s-fidelity*, is similar to the fidelity metrics in *LORE* [15]. However, instead of measuring the fidelity of the surrogate model, *s-fidelity* measures the fidelity of the counterfactual set (i.e., percentage of counterfactuals in the pseudo counterfactuals set). Given a binary classification model f, the factual sample $\hat{\mathbf{x}}$, and a set $S \subseteq \mathcal{X}$ (e.g., a pseudo counterfactual set), the *s-fidelity* is defined as follows:

$$s\text{-}fidelity_f(S, \hat{\mathbf{x}}) = \frac{\#\{\mathbf{x} \in S \cap \mathcal{X} \mid f(\mathbf{x}) \neq f(\hat{\mathbf{x}})\}}{\#(S \cap \mathcal{X})} \tag{7}$$

The fidelity measures can not be easily calculated because the input space, $\mathcal{X}$, is usually infinite. For this reason, the fidelity measures are estimated using the training set as input space, a dataset generated using synthetic methods, or a mix of both. An example of pseudo counterfactual sets is *LORE* counterfactual rules [15], which do not make claims about the *s-fidelity*.

This paper uses a representation for counterfactual sets defined by the ranges of each feature, which might be bounded (e.g., $1 < x_0 \leq 10$) or not (e.g., $x_0 > 0$ or $x_0 \leq 3$). As already mentioned, this representation has the same interpretation as rules in a DT. Further, counterfactual sets are easier to apply and understand than counterfactuals, especially when the features of the proposed changes involve high variability (e.g., luminosity and temperature) or measurements (e.g., the height or weight of a person). Notice that a feature with high variability is better explained using a range than a constant value. An example to illustrate this issue is provided. A ML-based system for predicting if a data center temperature is adequate is considered. This system predicts inadequate temperature for a data center whose temperature is 50° F. A counterfactual for this prediction could be *temperature* = 55° F, which is the minimum recommended temperature in a data center. This counterfactual does not indicate if a temperature greater than 55° F is adequate. On the other hand, counterfactual sets help to understand the changes better and to apply them by providing the features sub-region in which the counterfactual

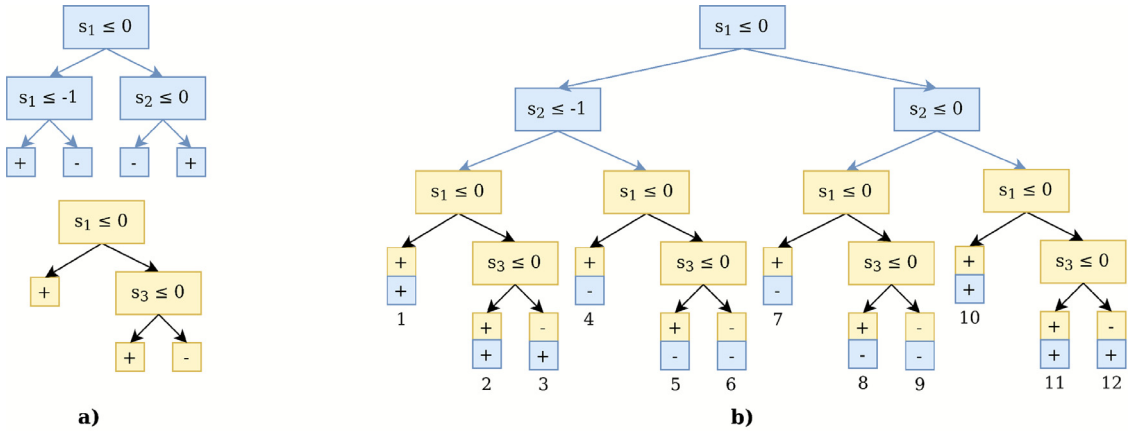

Fig. 1. Example of a simple method to fuse a RF with two tree predictors (a) into a single DT (b).

holds. Thus, a counterfactual set could be 55° F $<$ *temperature* $\leq$ 65° F, which provides more useful information as it is easier to keep the temperature in a range than in a constant value.

In addition to the improvement in explainability, counterfactual sets also help to measure the quality of the explanation and to diagnose ML models. For example, consider a counterfactual set for the previous example that suggests changing the temperature to 64.75° F $<$ *temperature* $\leq$ 65° F to have an adequate temperature. In this context, from the perspective of the domain expert, they could decide not to trust the explanation because there is no noticeable difference between 65° F and 66° F or because the range is too small. From a ML view, this would help to diagnose ill-defined areas in the classifier. As for the counterfactuals, this reasoning could not be made because we do not have the context of the changes. Therefore, counterfactual sets provide a context that helps to understand the explanation in those situations.

4. Counterfactual set extraction from random forest

A RF is defined as a set of tree predictors and a method to aggregate the outcomes of the individual tree predictors into a single result. Each tree predictor is trained with a random subsample of the dataset, in features, observations, or both. The most common aggregation methods are the most common class or the average probability, being the later a generalization over the former. *RF-OCSE* is defined using the second aggregation method.

Counterfactual explanation extraction from a RF differs from a DT, as a consensus among the individual tree predictors is needed. For this reason, counterfactuals can not be extracted independently from each tree predictor, since the proposed counterfactuals might be incompatible. To overcome this limitation, this work proposes to convert a RF into a DT and then to extract the counterfactuals. Counterfactual extraction from a DT is straightforward, as all leaves with different class than the current prediction are counterfactuals [7,15]. As rules represent the decision surface of a DT, the optimal counterfactual can be obtained by searching the rule from the counterfactual class whose distance is smaller. This distance between a sample and a rule is calculated as the minimum distance from the sample to all the samples that satisfy the rule. Further, a counterfactual set can be defined using the rule that satisfies a counterfactual within the fused RF. In this case, if the counterfactual is optimal, then the counterfactual set obtained using this method is considered optimal as it contains the optimal counterfactual.

The proposed method to convert a RF into a DT in order to extract counterfactual sets is presented in Section 4.1. Section 4.2 describes how the fusion can be improved by pruning the resulting DT. Finally, Section 4.3 details how the counterfactual sets can be extracted without fully fusing a RF.

4.1. Fusion of random forest tree predictors

The following notation will be used throughout the rest of this section. Let a RF be defined as a set T of tree predictors. Let R be a set of rules of all tree predictors in a RF, that is, the paths from the root of a tree predictor to the leaves, and r a given rule. The set R has at least one rule of each tree predictor. A tree predictor, t, is defined by a set of rules. Let $t(R)$ be the subset of rules in the set R that belongs to the tree predictor t. Let L be the set of classes in the problem and l a given class. A condition c is defined as a feature $s(c)$, a comparison type $cmp(c)$ ($\leq$ or $>$) and a threshold value $v(c)$. Let $r(c)$ be the rule that contains the condition c. Let C be the set of all conditions. Let a rule r be defined as a set of conditions $C(r)$ and the probability $p_l(r)$ for each class $l \in L$. Let $C(R)$ be the subset of conditions of C in the rules R, and let C_s be the subset of conditions C whose feature is s. Finally, let f be a RF model, $\hat{\mathbf{x}}$ the factual sample, and d a distance function between the factual sample, $\hat{\mathbf{x}}$, and other observations.

The conversion aims to represent a RF as a single DT, without modifying the decision surface. Intuitively, the fusion starts with a tree predictor in a RF, and plugs in every leaf, a tree predictor from the RF. This method is repeated until all tree predictors are added. The class of the rules in the resulting DT is defined using the average probability of all leaves in that rule.

An example of this fusion method is depicted in Fig. 1 for a RF with two tree predictors. Notice that some of the leaves are unreachable because two conditions of their rule are exclusive. The leaves 2, 3, 5, 6, 7 and 10 can not be reached because they have two exclusive conditions, $s_1 \leq 0$ and $s_1 > 0$. The simplified DT without the unreachable paths is depicted in Fig. 2. As the number of tree predictors in a RF grows, the number of unreachable nodes increases exponentially, expending computing time and memory resources. Furthermore, the selection order of the tree predictors has implications on the size of the resulting DT, which might make the problem intractable.

To reduce the size of the resulting DT, the proposed fusion method uses a modification of the *CART* algorithm. It takes a set of tree predictors as input and returns an equivalent DT. *CART* algorithm follows a top-down strategy, splitting at each step the dataset into two partitions. The algorithm is applied recursively on each partition until no further information gain is obtained or a predefined stop condition is met. The splits at each step are selected using a greedy approach to maximize the information gain.

In the proposed fusion method, the observations represent the rules extracted from the tree predictors. Each feature in the observation represents the list of conditions on that feature. The list of conditions for a feature might be empty, when there is no restriction on that feature, can be composed of one condition (e.g., $s_1 \leq 0$ or $s_1 > 0$), or have two conditions (e.g., $s_1 > 0$ or $s_1 \leq 1$). When a rule has two or more condi-

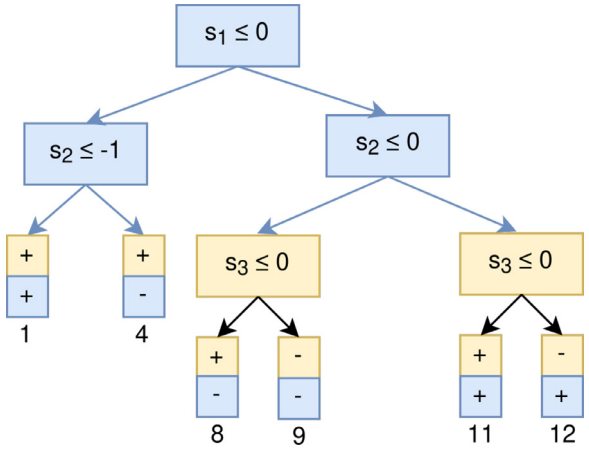

Fig. 2. Simplified DT from the example in the Fig. 1. The number in the leaves indicates the origin of the leaves in the full DT.

tions on a feature of the same type ("less or equal than", and "greater than"), only the most restrictive condition is considered. For example, in the conditions $s_0 > 0$, $s_0 > 1$ and $s_0 > 2$, the most restrictive condition, $s_0 > 2$, is selected.

The splits in the *CART* algorithm are exclusive, meaning that a given observation can only belong to one of the two resulting partitions because every feature has a fixed value. However, rules might not have a condition for every feature, and the condition might be met on both sides (e.g., the condition $s_1 \leq 2$ is met in both sides of the split $s_1 \leq 1$).

The partitions are calculated following Algorithm 1. The inputs are

Algorithm 1 partition_calculation.

Require: (R, s, v)
  $R_l, R_r \leftarrow R$
  for $c \in C(R)_s$ **do**
    if $cmp(c) = '\leq'$ **and** $v(c) > v$ **then**
      $R_l \leftarrow R_l - \{r(c)\}$
    else if $cmp(c) = '>'$ **and** $v(c) \leq v$ **then**
      $R_r \leftarrow R_r - \{r(c)\}$
    end if
  end for
  return R_l, R_r

the rules R in the current partition, the feature to filter s, and the threshold value v. It starts with the resulting partitions R_l and R_r, left and right partitions respectively, being equal to the full partition R. Then, the algorithm iterates over all the conditions in the partition of the given feature $C(R)_s$. In the case of "less or equal than" conditions ($s \leq v$), they filter all the rules which have a condition c such that $v(c) > v$. In the "greater than" conditions ($s > v$), they filter all the rules which have a condition c such that $v(c) \leq v$. Notice that, as previously mentioned, any "less or equal than" condition whose value is greater than the partition threshold falls in both sides, or the "greater than" conditions, if the value is lower than the threshold condition.

In the *CART* algorithm, the information gain calculation procedure uses the average probabilities of each partition. Although the proposed method uses the same approach, the estimation of the probabilities of the partitions is modified to reflect the probability estimation method of the RF. First, the method calculates the average probability for each tree predictor in the partition as follows:

$$P_{l,t}(R) = \frac{\sum_{r \in t(R)} p_l(r)}{\#t(R)} \tag{8}$$

where $\#t(R)$ is the number of rules in the set R belonging to the tree predictor t. This probability is always defined, as there is at least one rule of each tree predictor in a partition. Then, the probability estimation for the partition is defined as the average of the probabilities of each tree predictor:

$$P_l(R) = \frac{\sum_{t \in T} P_{l,t}(R)}{\#T} \tag{9}$$

where $\#T$ is the cardinality of the set T.

To calculate the information gain, first, the Entropy $E(R)$ of a given rule partition, R, is calculated as follows:

$$E(R) = \sum_{l \in L} -1 \cdot P_l(R) \cdot \log(P_l(R)) \tag{10}$$

Then, the entropy for the partitions R_l and R_r is calculated, and it is added proportionally to the number of rules in each partition, $\#R_l$ and $\#R_r$ respectively:

$$E(R_l, R_r) = \frac{E(R_l) \cdot \#R_l + E(R_r) \cdot \#R_r}{\#R_l + \#R_r} \tag{11}$$

Finally, the information gain is calculated as the reduction of Entropy calculated as follows:

$$G(R_l, R_r, R) = E(R) - E(R_l, R_r) \tag{12}$$

The split selection method, described in Algorithm 2, selects the

Algorithm 2 select_split.

Require: (R)
  $best\_s, best\_v$
  $best\_score \leftarrow 0$
  for $c \in C(R)$ **do**
    $R_l, R_r \leftarrow partition\_calculation(R, s(c), v(c))$
    $gain \leftarrow G(R_l, R_r, R)$
    if $gain > best\_score$ **then**
      $best\_score \leftarrow gain$
      $best\_s \leftarrow s(c)$
      $best\_v \leftarrow v(c)$
    end if
  end for
  return $best\_s, best\_v$

split using an exhaustive approach. For each condition from the rules in the partition, the method splits the partition in R_l and R_r using the Algorithm 1, and then calculates the information gain. The algorithm returns the split with the largest information gain.

The fusion method, described in Algorithm 3, is similar to the *CART*

Algorithm 3 rf_to_dt.

Require: (R)
  if $\#R = \#T$ **then**
    $node \leftarrow \{l = P_l(R) : l \in L\}$
  else
    $s, v \leftarrow select\_split(R)$
    $R_l, R_r \leftarrow partition\_calculation(R, s, v)$
    $node_l \leftarrow rf\_to\_dt(R_l)$
    $node_r \leftarrow rf\_to\_dt(R_r)$
    $node \leftarrow (s, v, node_l, node_r)$
  end if
  return node

algorithm, and it follows a recursive partitioning approach. The base case (stop condition) of the recursive method is satisfied when only one rule of each initial tree predictor remains ($\#R = \#T$). This is always possible, as the rules extracted from a tree predictor cover the entire feature space and are mutually exclusive (i.e., an observation can only fall on a leaf in a DT). Therefore, any split selected from the rules in the partition (R) is satisfied by at least one rule in the partition (R) of each tree predictor. When the base case is not satisfied, the method first selects

a split using the Algorithm 2. Then, the algorithm is recursively called with each partition (R_l and R_r) calculated using Algorithm 1. The fusion method returns a binary tree structure, which represents the same decision surface as the input tree predictors.

4.2. Partial random forest to decision tree fusion

The main limitation of the proposed fusion method is the size of the resulting DT. As the number of tree predictors in a RF and their depth grows, the number of nodes in the converted DT increases exponentially. In the worst conversion scenario, when the tree predictors do not share features, the equivalent DT for a RF with N tree predictors with depth m would have depth $N \cdot m$ and $2^{N \cdot m+1} - 1$ nodes.

The fusion method described in Algorithm 3 replicates the decision surface and probabilities of a RF; however, counterfactual extraction procedures only use the class of the leaves. To calculate the class of the leaves, the fusion method only needs to partially merge the RF, vastly reducing the number of nodes in the resulting DT. This can be achieved by modifying the stop condition, stopping when the class is already determined. In some situations, the class will not be modified by adding more splits. For instance, when all the remaining rules in the partition belong to the same class, or when most tree predictors only have one rule, and the remaining tree predictors do not have enough weight to modify the outcome.

To determine the class of the partition when there are unset tree predictors (i.e., a DT with more than one rule in the partition), the stop condition can use a worst-case scenario. The rule that defines this scenario is calculated as follows.

$$r^*_{t,l,z}(R) = arg\,max_{r \in t(R)}\, p_z(r) - p_l(r)\,. \tag{13}$$

That is, for a given tree predictor, the worst-case rule is the rule that implies the maximum difference in probability between a given class z and the expected class l of the partition R.

The definition can be extended for a RF as an average of the probabilities of the worst-case rule for each tree predictor.

Let

$$W_l(R) = \frac{\sum_{t \in T} p_l(r^*_{t,l,z}(R))}{\#T}\,, \tag{14}$$

be the average of the probabilities of the worst-case rule for the expected class l of the partition R, and

$$W_z(R) = \frac{\sum_{t \in T} p_z(r^*_{t,l,z}(R))}{\#T}\,, \tag{15}$$

be the average of the probabilities of the worst-case rule for other class z of the partition R.

The value $W_l(R)$ represents a lower bound of the probability $P_l(R')$, where R' is a subset of R. The subset R', as in the case of R, contains at least one rule of each tree predictor. On the other hand, the value $W_z(R)$ represents an upper bound of the probability $P_z(R')$.

This approximation can select rules in the unset tree predictors that are exclusive, thus representing situations that are not possible. However, it is a fast method to compute an upper and lower bound of the probability of the classes l and z, respectively.

The early-stop method, described in Algorithm 4, checks if the class of the partition is already determined using the worst-case scenario estimations defined above. The method is restricted to check only the class l with the highest probability among the tree predictors that are already set. Thus, only the class which is most likely to be the class of the partition is tested, reducing the method run time. For each class z such that $z \neq l$, the method tests in worst-case scenario $Pw_l(R, T, z)$ if the probability p_l is greater than p_z. If every test is met, then the class of the partition is l. Otherwise, the partition class can not be estimated without adding more conditions using the worst-case approach.

Algorithm 4 early_stop.

Require: (R)
  $l \leftarrow \text{argmax}_{l \in L} P_l(R)$
  for $z \in L : z \neq l$ **do**
    if $W_l(R) \leq W_z(R)$ **then**
      return FALSE
    end if
  end for
  return TRUE

4.3. Partial counterfactual set extraction

Despite reducing the size of the resulting DT, the partial fusion method also becomes intractable when the number of tree predictors and their depth grow. Because of the size of the fused DT, the extraction procedure would yield thousands of counterfactuals. Notice that every leaf with class distinct to the predicted class is a counterfactual, which results in a direct correlation between tree size and the number of counterfactuals. The users usually only expect a few counterfactual explanations and iterate through them if they do not meet their expectations. For this reason, counterfactuals are sorted to give users the most relevant explanations according to a given distance.

The counterfactual generation method extracts counterfactuals from the resulting DT. Moreover, it is possible to extract the counterfactuals when merging a RF. By integrating the fusion and counterfactual extraction method, the method can yield a counterfactual whenever the class determined in the partition is different from the predicted class. Therefore, an explicit memory representation of the resulting DT is not needed, which might be memory taxing. However, the order of the counterfactuals is arbitrary, and it does not follow the given distance. For this reason, the extraction procedure needs to calculate all possible counterfactuals to short them, thus fully converting the RF and incurring in the size problems of the fusion.

To avoid the calculation of all possible counterfactuals, the fusion method can be modified to prune the paths whose current distance to the factual sample is greater than the distance from the closest known counterfactual. Therefore, the fusion becomes more restrictive as closer counterfactuals are found, reducing the size problem. The initial counterfactual can be estimated using the *Minimum Observable* (*MO*) approach that selects the closest instance in the training set that belongs to the counterfactual class. The disadvantage of this method is that it has to partially merge the RF for each counterfactual individual, as opposed to the full fusion that is made only once. However, when working with large RFs, only the former is feasible.

The performance of the proposed method can be further enhanced by focusing the fusion around the factual sample. To focus the fusion, the rules whose distance from the factual sample is greater than a given threshold can be removed. This distance is calculated as the distance of the closest sample from the rule to the factual sample. The threshold, as in the path pruning, is the distance to the closest known counterfactual. This procedure allows building a DT that mimics a RF in a locality without modifying the decision surface (i.e., the rules defined in that locality are not removed).

The *RF-OCSE* extraction procedure, described in Algorithm 5, extracts the set of rules which satisfy the optimal counterfactual from a RF according to a given distance. The algorithm takes as input a set of rules R, a RF model f, a distance d between the factual sample and a possible counterfactual, the factual sample $\hat{\mathbf{x}}$, the factual sample updated to match the current state $\mathbf{x}$, and the maximum distance for the counterfactual set to be extracted max_d.

The *RF-OCSE* method first prunes the rules keeping only those rules in R whose distance to the factual sample $\hat{\mathbf{x}}$ is less than max_d, resulting in R_{prune}. Then, the method checks if the class is already determined using the *early_stop* algorithm. If the class is determined, the method checks if the estimated class is a counterfactual, and if the current distance is less than the maximum distance max_d. If those conditions are

Algorithm 5 *RF-OCSE* .

Require: $(R, f, d, \vec{\hat{x}}, \vec{x}, \max\_d)$
  $R', found \leftarrow \{\}, FALSE$
  $R_{prune} \leftarrow prune\_rules(R, \max\_d, \vec{\hat{x}})$
  if $early\_stop(R_{prune})$ **and** $f(\vec{\hat{x}}) \neq f(\vec{x})$ **and** $d(\vec{\hat{x}}, \vec{x}) < \max\_d$ **then**
    $R', found \leftarrow R_{prune}, TRUE$
    $max\_d \leftarrow d(\vec{\hat{x}}, \vec{x})$
  else
    $s, v \leftarrow select\_split(R_{prune})$
    $R_{left}, R_{right} \leftarrow partition\_calculation(R_{prune}, s, v)$
    if $\vec{x}_s \leq v$ **then**
      $R_{meet}, R_{not\_meet} \leftarrow R_{left}, R_{right}$
    else
      $R_{meet}, R_{not\_meet} \leftarrow R_{right}, R_{left}$
    end if
    $R', \max\_d, found \leftarrow RF - OCSE(R_{meet}, f, d, \vec{\hat{x}}, \vec{x}, \max\_d)$
    $\vec{x}_{not\_meet} \leftarrow update(x, s, v)$
    if $d(\vec{\hat{x}}, \vec{x}_{not\_meet}) \leq \max\_d$ **then**
      $R'_{not\_meet}, \max\_d, found_{not\_meet} \leftarrow RF - OCSE(R_{not\_meet}, f, d, \vec{\hat{x}}, \vec{x}_{not\_meet}, \max\_d)$
      if $found_{not\_meet}$ **then**
        $R', found \leftarrow R'_{not\_meet}, TRUE$
      end if
    end if
  end if
  return $R', \max\_d, found$

satisfied, then the set of rules R' that defines the counterfactual is returned. If the class can not be determined, the method selects a split and divides the rules R_{prune} into those rules that satisfy the split R_{meet} and those which do not R_{not_meet}. Next, the method performs a recursive call with the partition of rules that meet the split R_{meet}, as they are close to the factual sample because they do not require changes. The method updates the current distance to the counterfactual max_d and stores the returned set of rules into R' (if a counterfactual has not been found, it is an empty set) and the flag *found* that indicates if a counterfactual has been found. Then, the method updates c to not meet the split resulting in c_{not_meet}. This update takes into account the feature type. Binary variables are converted from 0 to 1 or vice versa. Real variables are set to the value of the threshold if the condition is "less or equal than", or if it is "greater than" to the threshold plus a small positive constant. Integer variables behave like real, but in the case of greater than they are set to the next integer. If the distance from c_{not_meet} to the factual sample is lower than max_d, then the method is recursively called with the set of rules which are not meet, R_{not_meet}. If the recursive call returns a valid counterfactual, then R' is updated. Finally, the method returns the set of rules R', the maximum distance max_d, and a flag that indicates if a counterfactual was found.

The optimal counterfactual set can be extracted by combining the rules that satisfy the optimal counterfactual. This rule combination only requires to select the most restrictive conditions in the rules for each feature. Notice that the early stop method can determine the class when there is more than one rule active in a tree. However, an observation can only be satisfied by one rule in each tree. Thus, the rules that do not satisfy the counterfactual are excluded from the rule combination.

The *set-coverage* of the resulting counterfactual set can be improved by reducing the number of rules considered in the rule combination procedure. Thus, the minimum number of rules that ensures the outcome of the model is the counterfactual class is considered. A greedy approach to perform this rule selection, *greedy-selection*, is described in Algorithm 6. The algorithm takes as input the resulting set of rules, R, from the *RF-OCSE* extraction procedure that meet the optimal counterfactual, the

Algorithm 6 *greedy-rule-selection* .

Require: $(R, p_c, \hat{y})$
  $R\prime \leftarrow \{\}$
  $R_s \leftarrow R\ sorted\ by\ descending\ p_{\hat{y}(r)}$
  $i \leftarrow 0$
  $acc_{prob} \leftarrow 0$
  while $i < \#R_s\ and\ acc_{prob} < p_c$ **do**
    $acc_{prob} \leftarrow acc_{prob} + p_{\hat{y}}(R_s[i]) / \#R_s$
    $R\prime \leftarrow R\prime \cup \{R_s[i]\}$
    $i \leftarrow i + 1$
  end while
  return $R\prime$

desired probability threshold p_c and the counterfactual class $\hat{y}$ such that $\hat{y} = f(\hat{\mathbf{x}})$.

The *greedy-rule-selection* method first sorts in descending order the rules r in R according to their probability on the label $\hat{y}$. Then, it iterates over the sorted rules adding them to the resulting set of rules, R', until the accumulated probability is greater than the probability threshold p_c. Finally, the method returns R'. The probability threshold p_c is set to 0.5, which is the minimum probability that ensures the label prediction. However, other thresholds might be considered. For example, in multi-class or imbalanced classification problems, or to generate pseudo counterfactual sets if the threshold p_c is set to less than 0.5.

5. Experiments

RF-OCSE has been tested in 10 real datasets against several state-of-the-art alternatives using counterfactual and counterfactual sets evaluation metrics. Section 5.1 describes the baselines, datasets, and the metrics used to evaluate the counterfactual and counterfactual sets. Sections 5.2 and 5.3 describe the counterfactual and counterfactual sets evaluations respectively.

5.1. Experiments setup

Baselines. The performance of *RF-OCSE* is evaluated using several metrics against state-of-the-art methods whose implementation is available online: *MACE*[1] (with tolerance 10^{-5}) [13], *LORE*[2] (using genetic neighborhood) [15], *FT*[1] [21], *FBT*[3] [26], and *MO*[1], which is defined as the closest counterfactual in the training set. In the *FBT* method, the counterfactual and pseudo counterfactual sets are extracted using the same procedure as *LORE*.

In the counterfactual set quality evaluation, only *LORE* and *FBT*, besides *RF-OCSE*, are applicable. The RF was implemented in scikit-learn [27] with default parameters. The code for *RF-OCSE* and the experiments are available online[4].

Datasets. Ten tabular datasets, listed on Table 1, are used. These datasets contain integer, categorical, ordinal, and real features. The datasets *adult, compas* and *credit* are preprocessed using the same approach as in *MACE* experiments [13]. The categorical and binary variables are represented using numerical encoding, and the continuous variables are standardized.

The experiments are carried out using 10-fold cross-validation. The counterfactuals are extracted on the test partitions. The results reported are the average over the test set of all folds. In the case of an invalid counterfactual (i.e., the observation belongs to the factual class or the method did not extract a counterfactual), the observation is not used to compute the average. Invalid counterfactuals are only possible on *LORE*

[1] https://github.com/amirhk/mace
[2] https://github.com/riccotti/lore
[3] https://github.com/sagyome/forest_based_tree
[4] https://github.com/rrunix/libfastcrf

Table 1
Description of the datasets used in the experiments.

Dataset	N. Features	Feature types	N. Instances
abalone	8	real, categorical	4177
adult	11	real, integer, binary, categorical	30,718
banknote	4	real	1372
compas	5	integer, binary	5278
credit	14	real, integer, binary	29,623
mammographic masses	5	integer	830
occupancy	5	real	2665
pima	8	real, integer	768
postoperative	8	integer, binary, categorical	86
seismic	15	integer, binary, categorical	2584

Table 2
Relative counterfactual improvement of *RF-OCSE* over the alternatives. The percentage of valid counterfactuals is in parentheses in those methods that do not always generate valid counterfactuals by design. NA implies that the relative counterfactual improvement could not be calculated because they were no valid counterfactuals. The best relative counterfactual improvement for each dataset is in bold.

Dataset	Method								
	FBT		*FT*		*LORE*		*MACE*	*MO*	RF-OCSE
abalone	78.77	(9%)	90.76	(95%)	67.55	(23%)	7.02	83.42	**0.00**
adult	30.44	(36%)	64.30	(100%)	37.50	(19%)	0.51	75.60	**0.00**
banknote	23.58	(31%)	NA	(0%)	20.65	(27%)	0.22	61.77	**0.00**
compas	0.04	(100%)	54.24	(85%)	8.66	(79%)	**0.00**	0.58	**0.00**
credit	90.14	(12%)	93.68	(100%)	60.79	(24%)	21.87	89.29	**0.00**
m. masses	8.12	(82%)	59.52	(100%)	8.05	(73%)	0.04	43.12	**0.00**
occupancy	28.74	(42%)	71.22	(30%)	31.41	(35%)	0.66	65.29	**0.00**
pima	66.41	(15%)	81.13	(100%)	47.17	(28%)	1.74	86.01	**0.00**
post-op.	5.18	(67%)	39.58	(73%)	0.76	(52%)	**0.00**	40.20	**0.00**
seismic	91.64	(6%)	94.31	(30%)	65.92	(5%)	3.68	77.56	**0.00**

and *FBT*, that rely on local approximations, and *FT*, that only generate counterfactuals when it can be derived from a rule from any of the tree predictors of the RF. In the case of *RF-OCSE, MACE*, and *MO*, they generate valid counterfactuals by design, as long as they exist.

Counterfactual evaluation metrics. The counterfactuals are evaluated in terms of distance to the factual sample and percentage of valid counterfactuals. The selected counterfactual distance is the average of the pairwise distances based on the feature type proposed in *MACE* [13]. The distance ranges from 0 to 1 and it is parametrized with the $L1$ norm to induce sparsity. Lower distance values are desirable, as counterfactuals closer to the factual sample are expected to be easier to understand. In the case of *RF-OCSE* and *LORE*, the distance reported is from the closest counterfactual in the counterfactual set and pseudo counterfactual set respectively. The distance is reported in terms of improvement of *RF-OCSE* with respect to the other approaches. Given the counterfactual generated by *RF-OCSE*, $\mathbf{c}$, the relative counterfactual improvement over other approach is defined as follows [13]:

$$rci(\mathbf{c}, \mathbf{z}, \hat{\mathbf{x}}) = 100 \cdot (1 - d(\mathbf{c}, \hat{\mathbf{x}})/d(\mathbf{z}, \hat{\mathbf{x}})) \tag{16}$$

where $\mathbf{z}$ is the counterfactual from the other approach, $\hat{\mathbf{x}}$ is the factual sample, and d is a distance function.

Counterfactual sets evaluation metrics. The counterfactual sets from *RF-OCSE, FBT*, and *LORE* are evaluated using the desirable properties of counterfactual sets and pseudo counterfactual sets: *s-fidelity* (see Eq. 7), *set coverage* (see Eq. 3), and percentage of populated counterfactuals sets. The percentage of populated counterfactual is defined as the percentage of counterfactual or pseudo counterfactual sets which contains at least one sample from the training set (i.e., *set coverage* is greater than 0).

5.2. Counterfactual evaluation

The average relative counterfactual improvement of *RF-OCSE* over the alternatives for each method is reported in Table 2. Notice that *RF-OCSE* obtains slightly better results than *MACE* due to the choice of tolerance since *MACE* can approximate the optimal counterfactual within an arbitrary tolerance. The tolerance parameter in *MACE* allows reducing the extraction time by performing fewer iterations. On the other hand, *RF-OCSE* can not make such trade-off since it always extracts the optimal counterfactual.

Table 3
Average extraction time for each dataset and method in seconds. The best extraction time for each dataset is in bold.

Dataset	Method					
	FBT	*FT*	*LORE*	*MACE*	*MO*	*RF-OCSE*
abalone	0.89	0.90	16.29	83.09	0.68	**0.17**
adult	**1.19**	7.16	16.37	23.95	1.33	1.96
banknote	0.65	**0.01**	15.51	1.83	0.20	0.02
compas	0.19	1.08	15.99	11.27	0.79	**0.05**
credit	0.89	9.30	17.25	41.86	1.31	**0.49**
m. masses	1.15	0.66	15.73	7.76	0.14	**0.02**
occupancy	0.95	**0.02**	15.53	2.26	0.38	**0.02**
pima	1.17	0.75	16.28	19.80	0.13	**0.05**
post-op.	0.37	0.05	16.39	6.44	**0.01**	**0.01**
seismic	0.27	0.08	16.80	13.68	**0.07**	0.26

RF-OCSE counterfactual distance to the factual sample is always equals or better than *MO*, as the counterfactual obtained with *MO* is used as an approximation for the initial counterfactual in *RF-OCSE*. The high difference in counterfactual distance obtained by *MO* in the *compas* dataset with respect to the other datasets is because the number of possible individuals is considerably less than in the other datasets. The *compas* dataset consists of three binary variables (race, gender, and charge degree), an ordinal variable with three levels based on the individual age and the prior count which is an integer variable ranging from 0 to 37, however, roughly 90% of the observations is in the range $0 - 10$. In this 90% of the dataset, there are 264 possible observations

Table 4
Counterfactual example extracted from the *adult* dataset for each method. Only the changes over the factual sample are shown. In this example, the counterfactuals extracted by *FBT* and *LORE* belong to the factual class and they are invalid.

Feature	Factual sample	Methods					
		FBT	*FT*	*LORE*	*MACE*	*MO*	*RF-OCSE*
Age	25.00					34.00	
CapitalGain	0.49	0.56	1.42	0.86	0.96	-0.15	0.96
CapitalLoss	-0.22						
EducationLevel	Doctorate						
EducationNumber	14.00						
HoursPerWeek	37.00					40	
MaritalStatus	Never-Married						
NativeCountry	United-States						
Occupation	Prof-specialty						
Relationship	Not-in-family						
WorkClass	Private						

that translate to 231 real observations in the dataset. Regarding the *FT* approach, the average improvement obtained by *RF-OCSE* is 64.87%, and it produced valid counterfactuals on the 83.91% of the cases. *FT* method did not produce a valid counterfactual in the banknote dataset because the counterfactuals could not be derived from a single rule of the tree predictors in the RF. The average improvement obtained over the *LORE* approach is 34.84 %, and its variance is high, which might imply that *LORE* highly depends on the data distribution. Further, the method only produced valid counterfactuals in the 31.11% of the cases. Finally, the average improvement over *FBT* is 40.30, and its variance is the highest, while the percentage of valid counterfactuals is 34.45%.

The average extraction times in seconds for each counterfactual and method are reported in Table 2. Extraction times are an important aspect of counterfactual extraction methods because large extraction times are not acceptable in some applications that are meant to be interactive. The time measures could be divided into three groups. In the first group, the methods that take roughly the same time independently of the dataset or the size of the RF. This is the case of *LORE* that took, on average, from 15 to 17 seconds to extract an explanation in the experiments. The time of the second group depends on the size of the dataset, which is the case of *MO* that took less than a second in most cases. Finally, the third group, which includes *FBT, FT, MACE*, and *RF-OCSE* whose time depends on the size of the RF. *RF-OCSE* obtained the best extraction time in most cases. It extracted the optimal counterfactuals in less than a second except for the adult dataset that took, on average, almost two seconds.

A counterfactual example extracted by each method for a sample in the *adult* dataset is shown in Table 4. The *adult* dataset goal is to predict if the income of an individual is less or equals than 50k dollars using demographic information (e.g., age, work hours per week, capital gain, and marital status). In some cases, the generated counterfactuals contain impractical changes such as changing the age. However, these changes are impractical because of the domain information. For example, a system to evaluate if an individual can get a driver license might suggest to wait until an individual has the minimum driving age. In this example, the individual earns less than 50k dollars, and the counterfactuals suggest possible changes to increase the income to more than 50k dollars.

5.3. Counterfactual sets evaluation

In the counterfactual sets experiments, reported in Table 5, the *set coverage*, the percentage of counterfactual sets populated, and the *s-fidelity* are listed for each dataset and method.

The pseudo counterfactual sets extracted by *LORE* provide the broadest *set coverage*. However, it comes at the cost of having, on average, a low *s-fidelity*. Thus, these pseudo counterfactual sets while providing a good *set coverage* do not correctly identify restrictions on the dataset that generate valid counterfactuals. A broader *set coverage* should only be preferred when they contain mostly valid counterfactuals. Otherwise, they do not add relevant information. A broad *set coverage* is only possible when there is a set of features conditions that apply to a high percentage of the individuals belonging to the counterfactual class. However, this is not always possible in complex datasets and ML models where the effect of the features depends on the region of the feature space.

Regarding *FBT*, the *set coverage* is lower than the *set coverage* of *LORE*, but it provides a better *s-fidelity*. The high *s-fidelity* contrast with the high number of invalid counterfactuals in Table 2. This could be explained by the method producing a good approximation of the RF that is not well-defined nearby the surface decision, which is where the optimal counterfactuals are.

On the other hand, for *RF-OCSE*, the counterfactual sets have a low *set coverage*, but they have the maximum *s-fidelity*, as they do not have non-counterfactual by design. Despite having a low *set coverage*, the counterfactuals extracted from the counterfactual set have a high similarity with the samples in the dataset that satisfy the counterfactual set using the proximity measure proposed in [28]. Thus, the proposed changes are realistic, and they are not incompatible. This is a consequence of constructing the counterfactual set from the remaining rules in the conversion, as most counterfactuals from the counterfactual set fall in the same leaves in the RF (they might differ in some leaves, but the class is already determined). Besides, the counterfactual set is guaranteed to contain the optimal counterfactual by design.

As a result of providing very specific counterfactual sets, they are satisfied by at least one sample of the dataset in the 21.14% of the cases. Counterfactual sets that do not contain samples from the training set provide an additional explainability possibility. This property allows us to detect and explain relevant regions currently not represented in the training set. In the case of highly populated datasets, these situations should not be common. This is the case of the *compas* dataset, where most feature combinations are covered.

The large difference in *set coverage* and percentage of counterfactual sets populated between *RF-OCSE* and *FBT*, and pseudo counterfactual sets of *LORE* is because *LORE* uses local rules, whereas *RF-OCSE* and *FBT* use global rules. Local rules only take into account splits relevant in the neighborhood. In contrast, global rules, in addition to those splits relevant in the neighborhood, they also assert all the splits that were relevant for the classification. Thus, global rules provide a context for the changes that allow better generalization and comparison with other individuals. However, this context does not imply that a feature outside the bounds of the rule will result in an invalid counterfactual. It only provides the context in which the prediction was made. Counterfactual sets can only be constructed using global rules, as local rules can not ensure that non-counterfactuals are not present.

Table 5
Evaluation of counterfactual sets extracted by *RF-OCSE* and pseudo-counterfactual sets from *LORE* and *FBT*. The results of *s-fidelity* and *set coverage* are the average over the test samples. The populated percentage is the average number of counterfactual sets with *set coverage* greater than 0, and it is in parentheses right to the *set coverage*. The best result for each dataset and metric is in bold.

Dataset	Method					
	FBT		*LORE*		*RF-OCSE*	
	s-fidelity	*set coverage*	*s-fidelity*	*set coverage*	*s-fidelity*	*set coverage*
abalone	66.72	6.20 (91.55%)	48.58	**18.12** (89.04%)	**100.00**	0.00 (0.79%)
adult	70.97	2.26 (71.87%)	28.69	**12.27** (94.48%)	**100.00**	0.02 (17.10%)
banknote	72.03	10.36 (27.77%)	38.35	**26.08** (98.03%)	**100.00**	0.03 (6.41%)
compas	**100.00**	0.85 (98.69%)	67.73	**28.60** (95.62%)	**100.00**	0.86 (98.94%)
credit	61.44	1.21 (56.29%)	58.33	**11.12** (83.16%)	**100.00**	0.00 (0.81%)
m. masses	88.34	0.64 (50.24%)	68.59	**29.52** (96.63%)	**100.00**	0.18 (60.84%)
occupancy	80.42	3.30 (32.50%)	40.18	**42.44** (98.24%)	**100.00**	0.44 (9.42%)
pima	62.17	3.74 (60.42%)	47.92	**16.35** (96.35%)	**100.00**	0.00 (0.13%)
post-op.	**100.00**	0.24 (16.28%)	56.12	**13.58** (81.40%)	**100.00**	0.41 (26.74%)
seismic	87.58	0.25 (37.50%)	77.53	**4.16** (71.34%)	**100.00**	0.00 (2.67%)

Table 6
Example of counterfactual set extracted by *RF-OCSE* and pseudo counterfactual set from *LORE* and *FBT*. The conditions that are not satisfied by the factual sample are in bold. The factual sample is the same as in the example in Table 4.

Feature	Method		
	LORE	*RF-OCSE*	*FBT*
Age		$\leq$ 26.00	$\leq$ 29.50
CapitalGain	$>$ **0.86**	$>$ **0.96**	**0.55** $<$ **x** $\leq$ **0.82**
CapitalLoss			$\leq$ 4.38735
EducationLevel		Masters, Doctorate, Assoc., ...	Doctorate, Assoc., Prof-school, ...
EducationNumber		$>$ 13.5	$>$ 10.50
HoursPerWeek		$\leq$ 38.00	$>$ 27.50
MaritalStatus		All except married and divorced	
NativeCountry			
Occupation			Other-service, Priv-house-serv, Prof-specialty, ...
Relationship		All except husband	Husband, Not-in-family
WorkClass		Gov., Private, Self-emp-inc	Gov., Private

Table 7
Evaluation of the pseudo counterfactual sets extracted by the rule selection relaxation in *RF-OCSE*. The number after *RF-OCSE* indicates the probability threshold used in Algorithm 6, or if the approach is dynamic. The results of *s-fidelity* and *set coverage* are the average over the test samples. The populated percentage is the average number of counterfactual sets with *set coverage* greater than 0, and it is in parentheses right to the *set coverage*. The best result for each dataset and metric is in bold.

Dataset	Method					
	*RF-OCSE*_dynamic		*RF-OCSE*_0.4		*RF-OCSE*_0.2	
	s-fidelity	set coverage	s-fidelity	set coverage	s-fidelity	set coverage
abalone	88.42	**0.63** (100%)	**99.54**	0.10 (15%)	93.98	0.28 (63%)
adult	96.83	**0.86** (100%)	**99.05**	0.11 (58%)	97.87	0.77 (84%)
banknote	99.55	7.16 (100%)	**99.99**	1.81 (82%)	99.71	**7.67** (99%)
compas	99.13	**0.87** (100%)	**99.85**	0.86 (100%)	99.50	**0.87** (99%)
credit	86.98	**0.11** (100%)	**96.89**	0.00 (9%)	95.20	0.02 (46%)
m. mases	94.67	**0.94** (100%)	**99.89**	0.26 (75%)	95.42	0.67 (90%)
occupancy	98.36	**7.25** (100%)	**99.61**	2.09 (68%)	98.63	6.73 (92%)
pima	88.10	**1.32** (100%)	**98.57**	0.02 (9%)	93.79	0.27 (60%)
post-op.	87.56	**2.45** (100%)	**100.00**	0.69 (44%)	87.52	1.82 (86%)
seismic	77.77	**0.12** (100%)	**96.10**	0.00 (11%)	89.42	0.02 (40%)

An example of pseudo counterfactual set extracted by *LORE* and *FBT*, and a counterfactual set of *RF-OCSE* is shown in Table 6. Notice that, despite counterfactual sets being larger than the pseudo counterfactual sets from *LORE*, this fact does not make it difficult to understand the explanation as most conditions do not imply changes. Counterfactual sets represented using global rules identify a subset of the population that are counterfactuals and are similar to each other using the proximity measure proposed in [28]. Therefore, it is easier for the individual to compare with those individuals as they are more similar to them. In the case of the local rules, it provides broad coverage that includes individuals that are not similar, and further, that do not belong to the counterfactual class.

RF-OCSE counterfactual sets can be relaxed into pseudo counterfactual sets to provide better coverage by modifying the probability threshold in the Algorithm 6. The threshold is set to 0.5 so that the greedy-rule-selection keeps the rules necessary to ensure the counterfactual class.

However, if the probability threshold is less than 0.5, the method produces pseudo counterfactual sets as it allows non-counterfactuals. This process represents a trade-off between the *s-fidelity* and the *set coverage*. Also, by having a broader *set coverage*, the percentage of populated counterfactual sets will increase.

In Table 7, the pseudo counterfactual sets extracted by the relaxation of the rule selection are evaluated. The rule selection is made with different probability thresholds in Algorithm 6. Besides, a dynamic approach that decreases the probability threshold until the *set coverage* is greater than 0 is considered.

The rule selection relaxation vastly increases in the best case (dynamic approach) the *set coverage* from 0.19 to 2.71 while having a high *s-fidelity*. Also, all counterfactual sets contain at least one sample from the training set. This increment in *set coverage* is notably from the threshold 0.4, which suggests that most counterfactual sets have a very restrictive condition that does not highly affect the classification as counterfactuals. The rule selection relaxation behaves similarly in all datasets, except for the seismic dataset. In this dataset, the *s-fidelity* in the dynamic approach is 77.77 which contrasts with the much higher *s-fidelity* values using the probability thresholds 0.2 and 0.4. This could be explained as the counterfactual sets, in some cases, not being realistic because the changes are not met by any sample in the dataset. However, this is not a problem of the extraction procedure but the decision surface of the ML model itself, as counterfactuals by definition, live in the vicinity of the decision surface where the class flip happens. This fact can be used to diagnose ML models by inspecting the decision surface through counterfactual sets.

6. Conclusions and future work

In this work, a new explanation technique called counterfactual sets and a method to extract these counterfactual sets from a RF are proposed. Counterfactual sets explanations use a sub-region of the feature space to explain the factual sample rather than a single counterfactual. This sub-region defines the area where a counterfactual holds, providing its classification context. In this regard, counterfactual sets explanations based on rules have the same interpretation as rules in a DT.

The extraction method is called *RF-OCSE*, and it guarantees that the optimal counterfactual is inside the extracted counterfactual set. The method is based on a partial fusion from a RF to an equivalent single DT using a modification of the *CART* algorithm. The partial fusion allows only converting the locality of the counterfactual by filtering whose distance is greater than the closest known counterfactual. The optimal counterfactual set is generated by combining the remaining rules in the conversion when the optimal counterfactual is found.

The generated counterfactual sets were evaluated in terms of *set coverage*, the percentage of populated counterfactual sets, and *s-fidelity*. *RF-OCSE* counterfactual sets do not contain non-counterfactuals by design. Regarding the set coverage, an advantage of *RF-OCSE* is its ability to find counterfactual sets that do not contain samples from the training set. Far from being a limitation, this property allows detecting and explaining relevant regions currently not represented in the training set.

The counterfactual sets from *RF-OCSE* can be relaxed into pseudo counterfactual sets to provide a broader *set coverage*. This relaxation is achieved by decreasing the threshold in the rule selection algorithm. The pseudo counterfactual sets were evaluated in the same settings as counterfactuals sets, achieving a better *set coverage* while keeping a high *s-fidelity*.

Future work will focus mainly on two areas: interpretability and performance. The quality of the generated counterfactuals could be improved by penalizing non-realistic counterfactuals and counterfactuals far from a prototype [29]. The coverage of the method could be increased by providing several diverse counterfactual sets in a similar way to diverse counterfactuals [30]. Better support for categorical features could also increase the coverage. The extraction procedure could be extended to support Gradient Boosting models by modifying the probability estimation method to be a weighted sum of the individual tree predictors. Finally, the method may convert a RF to a local DT in high-density areas to achieve zero-cost counterfactuals sets using the full conversion procedure in a region of the input space. Then, counterfactual sets could be efficiently extracted from the resulting DT by selecting the rule that satisfies a counterfactual.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Rubén R. Fernández: Conceptualization, Methodology, Software, Writing - review & editing. **Isaac Martín de Diego:** Writing - review & editing, Supervision, Funding acquisition. **Víctor Aceña:** Writing - review & editing. **Alberto Fernández-Isabel:** Writing - review & editing. **Javier M. Moguerza:** Writing - review & editing.

Acknowledgments

Research supported by grants from the Spanish Ministry of Economy and Competitiveness under the Retos-Colaboración program: SABERMED (Ref: RTC-2017-6253-1), Retos-Investigación program: MODAS-IN (Ref: RTI-2018-094269-B-I00); and donation of the Titan V GPU by NVIDIA Corporation. Ministry of Economy and Competitiveness (MINECO).

References

[1] A. Pentland, The data-driven society, Sci. Am. 309 (4) (2013) 78–83.
[2] A. Wolff, D. Gooch, J.J.C. Montaner, U. Rashid, G. Kortuem, Creating an understanding of data literacy for a data-driven society, The Journal of Community Informatics 12 (3) (2016).
[3] F. Doshi-Velez, B. Kim, Towards a rigorous science of interpretable machine learning, arXiv preprint arXiv:1702.08608 (2017).
[4] A. Adhikari, D. Tax, R. Satta, M. Fath, Example and feature importance-based explanations for black-box machine learning models, arXiv preprint arXiv:1812.09044 (2018).
[5] B. Mittelstadt, C. Russell, S. Wachter, Explaining explanations in ai, in: Proceedings of the conference on fairness, accountability, and transparency, ACM, 2019, pp. 279–288.
[6] I.-C. Yeh, K.-J. Yang, T.-M. Ting, Knowledge discovery on rfm model using bernoulli sequence, Expert Syst Appl 36 (3) (2009) 5866–5871.
[7] R.R. Fernández, I.M. de Diego, V. Aceña, J.M. Moguerza, A. Fernández-Isabel, Relevance metric for counterfactuals selection in decision trees, in: Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning, Springer, 2019, pp. 85–93.
[8] W.-Y. Loh, Classification and regression trees, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1 (1) (2011) 14–23.
[9] A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, F. Herrera, Explainable artificial intelligence (xai): concepts, taxonomies, opportunities and challenges toward responsible ai, Information Fusion 58 (2020) 82–115.
[10] C. Molnar, Interpretable machine learning, lulu.com, 2018.
[11] P.W. Koh, P. Liang, Understanding black-box predictions via influence functions, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 1885–1894.
[12] T. Miller, Contrastive explanation: a structural-model approach, arXiv preprint arXiv:1811.03163 (2018).
[13] A.-H. Karimi, G. Barthe, B. Belle, I. Valera, Model-agnostic counterfactual explanations for consequential decisions, arXiv preprint arXiv:1905.11190 (2019).
[14] 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 (2017).
[15] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, F. Giannotti, Local rule-based explanations of black box decision systems, arXiv preprint arXiv:1805.10820 (2018).
[16] J. Su, D.V. Vargas, K. Sakurai, One pixel attack for fooling deep neural networks, IEEE Trans. Evol. Comput. 23 (5) (2019) 828–841.
[17] A. White, A.d. Garcez, Measurable counterfactual local explanations for any classifier, arXiv preprint arXiv:1908.03020 (2019).
[18] 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 (2018).

[19] A. Barredo-Arrieta, J. Del Ser, Plausible counterfactuals: auditing deep learning classifiers with realistic adversarial examples, arXiv preprint arXiv:2003.11323 (2020).
[20] S. Liu, B. Kailkhura, D. Loveland, Y. Han, Generative counterfactual introspection for explainable deep learning, arXiv preprint arXiv:1907.03077 (2019).
[21] G. Tolomei, F. Silvestri, A. Haines, M. Lalmas, Interpretable predictions of tree-based ensembles via actionable feature tweaking, in: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, 2017, pp. 465–474.
[22] O. Bastani, C. Kim, H. Bastani, Interpreting blackbox models via model extraction, arXiv:1705.08504 (2017).
[23] H. Deng, Interpreting tree ensembles with intrees, International Journal of Data Science and Analytics 7 (4) (2019) 277–287.
[24] G. Vandewiele, K. Lannoye, O. Janssens, F. Ongenae, F. De Turck, S. Van Hoecke, A genetic algorithm for interpretable model extraction from decision tree ensembles, in: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2017, pp. 104–115.
[25] Y. Zhou, G. Hooker, Interpreting models via single tree approximation, arXiv preprint arXiv:1610.09036 (2016).
[26] O. Sagi, L. Rokach, Explainable decision forest: transforming a decision forest into an interpretable tree, Information Fusion 61 (2020) 124–138.
[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: machine learning in python, Journal of machine learning research 12 (Oct) (2011) 2825–2830.
[28] L. Breiman, Manual on setting up, using, and understanding random forests v3. 1. 2002, URL: http://oz. berkeley. edu/users/breiman/Using_random_forests_V3 1 (2002).
[29] S. Tan, M. Soloviev, G. Hooker, M.T. Wells, Tree space prototypes: Another look at making tree ensembles interpretable, arXiv:1611.07115 (2016).
[30] R.K. Mothilal, A. Sharma, C. Tan, Explaining machine learning classifiers through diverse counterfactual explanations, arXiv preprint arXiv:1905.07697 (2019).