FOCUS: Flexible Optimizable Counterfactual Explanations for Tree Ensembles

Ana Lucic
University of Amsterdam
Amsterdam, Netherlands
a.lucic@uva.nl

Harrie Oosterhuis
University of Amsterdam
Amsterdam, Netherlands
oosterhuis@uva.nl

Hinda Haned
University of Amsterdam &Ahold Delhaize
Amsterdam, Netherlands
h.haned@uva.nl

Maarten de Rijke
University of Amsterdam &Ahold Delhaize
Amsterdam, Netherlands
derijke@uva.nl

ABSTRACT

Model interpretability has become an important problem in machine learning (ML) due to the increased effect algorithmic decisions have on humans. Counterfactual explanations can help users understand not only why ML models make certain decisions, but also give insight into how these decisions can be modified. We frame the problem of finding counterfactual explanations as an optimization task and extend previous work that could only be applied to differentiable models. In order to accommodate non-differentiable models such as tree ensembles, we propose using probabilistic model approximations in the optimization framework. We introduce a simple approximation technique that is effective for finding counterfactual explanations for predictions of the original model using a range of distance metrics. We show that our counterfactual examples are significantly closer to the original instances compared to other methods designed for tree ensembles for four distance metrics.

1 INTRODUCTION

Model interpretability has become an important problem in machine learning. As ML models are prominently applied and their behavior has a substantial effect on the general population, there is an increased demand for understanding what contributes to their predictions [9]. For an individual who is affected by the predictions of these models, it would be useful to have an explanation that not only identifies the important features, but also provides insight into how these decisions can be *changed*. Counterfactual explanations are a natural solution to this problem since they frame the explanation in terms of what input (feature) changes are required to change the output (prediction). For instance, a user may be denied a loan based on the prediction of an ML model used by their bank. A counterfactual explanation could be of the form:

> *Had your income been* €1000 *higher, you would have been approved for the loan.*

Counterfactual explanations are based on *counterfactual examples*: generated instances that are close to an existing instance but have an alternative prediction. The difference between the original instance and the counterfactual example is the counterfactual explanation. Wachter et al. [36] propose framing the problem as an optimization task: finding *optimal* counterfactual explanations with the *minimal* changes to the input required to change the outcome.

Wachter et al. [36] assume that the underlying ML models are differentiable, which excludes an important class of widely applied and highly effective non-differentiable models: tree ensembles. We extend this work of by introducing differentiable approximations of tree ensembles that can be used in such an optimization framework, and propose a method, **F**lexible **O**ptimizable **Co**Unterfactual Examples for Tree Ensemble**S** (FOCUS). Given a trained tree-based model $\mathcal{M}$, FOCUS probabilistically approximates $\mathcal{M}$ by replacing each split in each tree with a sigmoid function centered at the splitting threshold. This results in a differentiable version of $\mathcal{M}$ from which FOCUS finds the feature values required for counterfactual examples via gradient descent. This approximation allows FOCUS to generate a counterfactual example $\bar{x}$ for an instance x based on the minimal perturbation of x such that the prediction changes: $y_x \neq y_{\bar{x}}$, where $y_x, y_{\bar{x}}$ are the labels $\mathcal{M}$ assigns to x and $\bar{x}$, respectively.

The notion of *minimality* for counterfactual examples is not uniquely defined and may differ depending on the problem setting and end-user [7]. For some tasks it may be more important to perturb the smallest proportion of features, whereas for other tasks it might be preferable to produce the smallest possible perturbations per feature, regardless of the number of features. In FOCUS, these preferences can be made explicit by the choice of distance function in the optimization framework. In our main experiments aimed at assessing FOCUS, we generate three types of counterfactual example for each tree-based model, corresponding to different distance metrics:

- Euclidean: minimize magnitude of each perturbation;
- Cosine: minimize change in relationship between features; and
- Manhattan: minimize proportion of features that are perturbed.

In additional experiments aimed at contrasting FOCUS with a recently proposed counterfactual explanation method, DACE [17], we use a fourth distance metric, used by the authors of DACE in the novel loss function they propose:

- Mahalanobis: minimize magnitude of each perturbation while accounting for correlation between variables.

This flexibility concerning the distance metric used allows FOCUS to generate explanations that can either be tailored to the end-users' needs, or to produce a variety of counterfactual explanations from which the user can decide the best fit for their problem setting. This allows us to customize our notion of 'minimality', which can vary depending on the context [7].

Our main findings concerning FOCUS are that it is (i) a more *effective* counterfactual explanation method for tree ensembles than previous approaches since it manages to produce counterfactual examples that are closer to the original input instances than existing approaches; (ii) a more *efficient* counterfactual explanation method for tree ensembles since it is able to handle larger models than existing approaches; and (iii) a more *reliable* counterfactual explanation method for tree ensembles since it is able to generate counterfactual explanations for all instances in a dataset, unlike existing approaches.

2 RELATED WORK

Counterfactual examples have been used in a variety of ML areas, such as reinforcement learning [23], deep learning [1], and explainable AI (XAI) [8, 13, 17, 18, 22, 24, 26, 28, 31, 35, 36]. Previous XAI methods for generating counterfactual examples are either model-agnostic [18, 22, 24, 26, 35] or model-specific [8, 13, 17, 28, 31, 36]. Model-agnostic approaches treat the original model as a "black-box" and only assume query access to the model, whereas model-specific approaches typically do not make this assumption and can therefore make use of its inner workings.

2.1 Algorithmic Recourse

Algorithmic recourse is a line of research that is closely related to counterfactual explanations, except that these methods include the additional restriction that the resulting explanation must be *actionable* [16, 19, 20, 34]. This is done by selecting a subset of the features to which perturbations can be applied in order to avoid explanations that suggest impossible or unrealistic changes to the feature values (i.e., change *age* from 25 → 50 or change *marital_status* from MARRIED → UNMARRIED). Although this work has produced impressive theoretical results, it is unclear how realistic they are in practice, especially for complex ML models such as tree ensembles. Existing algorithmic recourse methods cannot solve our task because they (i) are either restricted to solely linear [34] or differentiable [16] models, or (ii) require access to causal information [19, 20], which is rarely available in real world settings.

Another shortcoming of algorithmic recourse methods is that they require that the actionable subset of features is determined *in advance*, which can often mean that developers are put in the position of determining what is actionable. Although this is obvious for some features (i.e., age), actionability is not necessarily a binary condition, and what is actionable for one person may not be actionable for another [3]. Moreover, if we restrict the perturbations to only act on actionable features, we lose sight of any non-actionable features that the model deems important – although it is unrealistic to ask someone to change their age, it is still important to know

that the model's decisions depend on age. This also allows the user to contest the use of such a feature in determining their outcome.

Unlike algorithmic recourse methods, our work does not necessarily aim to provide the user with the definitive set of feature value changes they need for a different outcome. In contrast, we provide a counterfactual "menu of options" [3] that can be customized based on the chosen distance function. Although counterfactual explanations may be less actionable than algorithmic recourse methods, they provide a more complete picture of what is important to the model and allow the end-user to decide what is (not) actionable for themselves. Counterfactual explanations are meant to serve as a starting point for recourse: we argue that the task of recourse should not be solved from a purely algorithmic perspective, but rather as part of a human-in-the-loop process. Counterfactual explanations therefore provide the user with a more complete notion of potential feature value changes in comparison to existing algorithmic recourse methods, and serve as the algorithmic component of achieving recourse.

2.2 Model-specific Counterfactual Examples

We propose a model-specific approach for generating counterfactual examples. Previous model-specific work has either been done through optimization [8, 13, 36], mixed integer programming [17, 28], or heuristic search [31]. Some of this work is applicable to linear models or neural networks [8, 13, 28, 36], while other work can be applied to tree-ensembles [17, 31].

Our work is the first model-specific approach for generating counterfactual examples for tree ensembles through gradient-based optimization. We compare our method against existing approaches for tree-ensembles [17, 31], and refer the reader to Section 5.1 and 5.2 for a more detailed overview of these methods.

2.3 Adversarial Examples

Adversarial examples are a type of counterfactual example with the additional constraint that the minimal perturbation results in an alternative prediction that is *incorrect*. There are a variety of methods for generating adversarial examples [6, 12, 29, 30]; a more complete overview can be found in [4].

The main difference between adversarial examples and counterfactual examples is in the intent: adversarial examples are meant to *fool* the model, whereas counterfactual examples are meant to *explain* the model.[1] Another difference is that adversarial examples are typically studied in the context of image classification, whereas counterfactual examples are usually studied in the context of tabular or textual data. Perturbations to pixels in an image often result in changes that are undetectable to the human eye, whereas perturbations to feature values or the presence/absence of terms in a sentence are meant to be detectable, since this is what the explanation is created from.

2.4 Local Explanations

Our work is part of a broader family of local explanation methods that use model approximation in the pipeline of explaining individual predictions. Other common approaches to local explanations

---

[1] However, it also is possible to use counterfactual examples for fooling, and adversarial examples for explaining, as they are closely related.

involve approximating the original model locally with an inherently interpretable explanator (i.e., linear regression, shallow decision tree) to derive feature importances or decision rules [14, 27]. A shortcoming of these approaches is that the approximation is not necessarily guaranteed to mimic the original model, since it is typically a simpler version which is only valid locally.

This work is similar to ours in that it uses model approximations to explain individual predictions, but it differs from ours because: (i) the approximations are local – they only hold around the point in question, and (ii) our objective is not to simplify the model, but rather to produce a differentiable version in order to leverage gradient-based optimization techniques. In other words, previous methods generate local explanations via local approximations, while our method generates local explanations via global approximations.

2.5 Differentiable Trees

Part of our contribution involves constructing differentiable versions of tree ensembles by replacing each splitting threshold with a sigmoid function. This can be seen as using a (small) neural network to obtain a smooth approximation of each tree. Neural decision trees [2, 37] are also differentiable versions of trees, which use a full neural network instead of a simple sigmoid. However, these do not optimize for approximating an already trained model. Therefore, unlike our method, they are not an obvious choice for finding counterfactual examples for an existing model. Soft decision trees [15] are another example of differentiable trees, which instead approximate a neural network with a decision tree. This can be seen as the inverse of our task.

3 METHOD

In this section, we formalize the problem of generating counterfactual explanations and describe our method for generating counterfactual examples specifically for tree ensembles: **F**lexible **O**ptimizable **Co**Unterfactual Examples for Tree Ensemble**S** (FOCUS). FOCUS builds upon the optimization framework of Wachter et al. [36] and extends it to accommodate non-differentiable tree-based models.

3.1 Problem Formulation

A *counterfactual explanation* for an instance x and a model $\mathcal{M}$, Δ_x, is a minimal perturbation of x that changes the prediction of $\mathcal{M}$. $\mathcal{M}$ is a probabilistic classifier, where $\mathcal{M}(y \mid x)$ is the probability of x belonging to class y according to $\mathcal{M}$. The prediction of $\mathcal{M}$ for x is the most probable class label $y_x = \arg\max_y \mathcal{M}(y \mid x)$, and a perturbation $\bar{x}$ is a counterfactual example for x if, and only if, $y_x \neq y_{\bar{x}}$, that is:

$$\arg\max_y \mathcal{M}(y \mid x) \neq \arg\max_{y'} \mathcal{M}(y' \mid \bar{x}). \tag{1}$$

In addition to changing the prediction, the distance between x and $\bar{x}$ should also be minimized. We therefore define an *optimal counterfactual example* $\bar{x}^*$ as:

$$\bar{x}^* := \arg\min_{\bar{x}} d(x, \bar{x}) \text{ such that } y_x \neq y_{\bar{x}}. \tag{2}$$

where $d(x, \bar{x})$ is a differentiable distance function. The corresponding *optimal counterfactual explanation* Δ_x^* is:

$$\Delta_x^* = \bar{x}^* - x. \tag{3}$$

This definition aligns with previous ML work on counterfactual explanations [18, 22, 31]. We note that this notion of *optimality* is purely from an algorithmic perspective and does not necessarily translate to optimal changes in the real world, since the latter are completely dependent on the context in which they are applied. It should be noted that if the loss space is non-convex, it is possible that more than one optimal counterfactual explanation exists.

Minimizing the distance between x and $\bar{x}$ should ensure that $\bar{x}$ is as close to the decision boundary as possible. This distance indicates the effort it takes to apply the perturbation in practice, and an optimal counterfactual explanation shows how a prediction can be changed with the least amount of effort. An optimal explanation provides the user with interpretable and potentially actionable feedback related to understanding the predictions of model $\mathcal{M}$.

3.2 Our Method: FOCUS

Wachter et al. [36] recognized that counterfactual examples can be found through gradient descent if the task is cast as an optimization problem. Specifically, they use a loss consisting of two components: (i) a prediction loss to change the prediction of $\mathcal{M}$: $\mathcal{L}_{\mathcal{M}}(y_x, \bar{x})$, and (ii) a distance loss to minimize the distance d: $\mathcal{L}_d(x, \bar{x})$. The complete loss is a linear combination of these two parts, with a weight $\beta \in \mathbb{R}_{>0}$:

$$\mathcal{L}(x, \bar{x} \mid \mathcal{M}, d) = \mathcal{L}_{\mathcal{M}}(y_x, \bar{x}) + \beta \cdot \mathcal{L}_d(x, \bar{x}). \tag{4}$$

The assumption here is that an optimal counterfactual example $\bar{x}^*$ can be found by minimizing the overall loss: $\bar{x}^* = \arg\min_{\bar{x}} \mathcal{L}(x, \bar{x} \mid \mathcal{M}, d)$. Wachter et al. [36] propose a prediction loss $\mathcal{L}_{\mathcal{M}}$ based on the mean-squared-error. In contrast, we introduce a hinge-loss since we assume a classification task:

$$\mathcal{L}_{\mathcal{M}}(y, \bar{x}) = \mathbb{1}\left[y = \arg\max_{y'} \mathcal{M}(y' \mid \bar{x})\right] \cdot \mathcal{M}(y \mid \bar{x}). \tag{5}$$

Given a differentiable distance function d, the distance loss is: $\mathcal{L}_d(x, \bar{x}) = d(x, \bar{x})$. Allowing for flexibility in the choice of distance function allows us to tailor the explanations to the end-users' needs; we make the preferred notion of *minimality* explicit through the choice of distance function.

A clear limitation of this approach is that it assumes $\mathcal{M}$ is differentiable. This excludes many commonly used ML models, including tree-based models, on which we focus in this paper. We propose a solution through differentiable approximations of such models; an approximation $\widetilde{\mathcal{M}}$ should match the original model closely: $\widetilde{\mathcal{M}}(y \mid x) \approx \mathcal{M}(y \mid x)$. We define the prediction loss for $\widetilde{\mathcal{M}}$ as follows:

$$\widetilde{\mathcal{L}_{\mathcal{M}}}(y, \bar{x}) = \mathbb{1}\left[y = \arg\max_{y'} \mathcal{M}(y' \mid \bar{x})\right] \cdot \widetilde{\mathcal{M}}(y \mid \bar{x}). \tag{6}$$

We note that this loss is both based on the original model $\mathcal{M}$ and the approximation $\widetilde{\mathcal{M}}$: the loss is active as long as the prediction according to $\mathcal{M}$ has not changed, but its gradient is based on the differentiable $\widetilde{\mathcal{M}}$. This prediction loss encourages the perturbation to have a different prediction than the original instance by penalizing an unchanged instance. The approximation of the complete loss becomes:

$$\widetilde{\mathcal{L}}(x, \bar{x} \mid \mathcal{M}, d) = \widetilde{\mathcal{L}_{\mathcal{M}}}(y_x, \bar{x}) + \beta \cdot \mathcal{L}_d(x, \bar{x}). \tag{7}$$
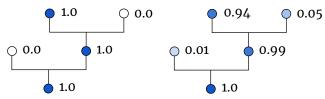
Figure 1: Left: A decision tree $\mathcal{T}$ and per node activations for a single instance. Right: a differentiable approximation of the same tree $\widetilde{\mathcal{T}}$ and activations for the same instance.

Since we assume that it approximates the complete loss,

$$\widetilde{\mathcal{L}}(x, \bar{x} \mid \mathcal{M}, d) \approx \mathcal{L}(x, \bar{x} \mid \mathcal{M}, d), \tag{8}$$

we also assume that an optimal counterfactual example can be found by minimizing it:

$$\bar{x}^* \approx \arg\min_{\bar{x}} \widetilde{\mathcal{L}}(x, \bar{x} \mid \mathcal{M}, d). \tag{9}$$

To obtain the differentiable approximation $\widetilde{\mathcal{M}}$ of $\mathcal{M}$, we construct a probabilistic approximation of the original tree ensemble $\mathcal{M}$. Tree ensembles are based on decision trees; a single decision tree $\mathcal{T}$ uses a binary-tree structure to make predictions about an instance x based on its features. Figure 1 shows a simple decision tree consisting of five nodes. A node j is activated if its parent node p_j is activated and feature x_{f_j} is on the correct side of the threshold θ_j; which side is the correct side depends on whether j is a *left* or *right* child; with the exception of the root node which is always activated. Let $t_j(x)$ indicate if node j is activated:

$$t_j(x) = \begin{cases} 1, & \text{if } j \text{ is the root,} \\ t_{p_j}(x) \cdot \mathbb{1}[x_{f_j} > \theta_j], & \text{if } j \text{ is a left child,} \\ t_{p_j}(x) \cdot \mathbb{1}[x_{f_j} \leq \theta_j], & \text{if } j \text{ is a right child.} \end{cases} \tag{10}$$

$\forall x, \; t_0(x) = 1$. Nodes that have no children are called *leaf nodes*; an instance x always ends up in a single leaf node. Every leaf node j has its own predicted distribution $\mathcal{T}(y \mid j)$; the prediction of the full tree is given by its activated leaf node. Let $\mathcal{T}_{leaf}$ be the set of leaf nodes in $\mathcal{T}$, then:

$$(j \in \mathcal{T}_{leaf} \wedge t_j(x) = 1) \rightarrow \mathcal{T}(y \mid x) = \mathcal{T}(y \mid j). \tag{11}$$

Alternatively, we can reformulate this as a sum over leaves:

$$\mathcal{T}(y \mid x) = \sum_{j \in \mathcal{T}_{leaf}} t_j(x) \cdot \mathcal{T}(y \mid j). \tag{12}$$

Generally, tree ensembles are deterministic; let $\mathcal{M}$ be an ensemble of M many trees with weights $\omega_m \in \mathbb{R}$, then:

$$\mathcal{M}(y \mid x) = \mathbb{1}\left[y = \arg\max_{y'} \sum_{m=1}^{M} \omega_m \cdot \mathcal{T}_m(y' \mid x)\right]. \tag{13}$$

If $\mathcal{M}$ is not differentiable, we are unable to calculate its gradient w.r.t. the input x. However, the non-differentiable operations in our formulation are (i) the indicator function, and (ii) a maximum operation, both of which can be approximated by differentiable functions. First, we introduce the $\widetilde{t_j}(x)$ function that *approximates the activation of node j*: $\widetilde{t_j}(x) \approx t_j(x)$, using a sigmoid function

with parameter $\sigma \in \mathbb{R}_{>0}$: $sig(z) = (1 + \exp(\sigma \cdot z))^{-1}$ and

$$\widetilde{t_j}(x) = \begin{cases} 1, & \text{if } j \text{ is the root,} \\ \widetilde{t_{p_j}}(x) \cdot sig(\theta_j - x_{f_j}), & \text{if } j \text{ is left child,} \\ \widetilde{t_{p_j}}(x) \cdot sig(x_{f_j} - \theta_j), & \text{if } j \text{ is right child.} \end{cases} \tag{14}$$

As σ increases, $\widetilde{t_j}$ approximates t_j more closely. Next, we introduce a *tree approximation*:

$$\widetilde{\mathcal{T}}(y \mid x) = \sum_{j \in \mathcal{T}_{leaf}} \widetilde{t_j}(x) \cdot \mathcal{T}(y \mid j). \tag{15}$$

The approximation $\widetilde{\mathcal{T}}$ uses the same tree structure and thresholds as $\mathcal{T}$. However, its activations are no longer deterministic but instead are dependent on the distance between the feature values x_{f_j} and the thresholds θ_j. Lastly, we replace the maximum operation of $\mathcal{M}$ by a softmax with temperature $\tau \in \mathbb{R}_{>0}$, resulting in:

$$\widetilde{\mathcal{M}}(y \mid x) = \frac{\exp\left(\tau \cdot \sum_{m=1}^{M} \omega_m \cdot \widetilde{\mathcal{T}_m}(y \mid x)\right)}{\sum_{y'} \exp\left(\tau \cdot \sum_{m=1}^{M} \omega_m \cdot \widetilde{\mathcal{T}_m}(y' \mid x)\right)}. \tag{16}$$

The approximation $\widetilde{\mathcal{M}}$ is based on the original model $\mathcal{M}$ and the parameters σ and τ. This approximation is applicable to any tree-based model, and how well $\widetilde{\mathcal{M}}$ approximates $\mathcal{M}$ depends on the choice of σ and τ. The approximation is potentially perfect since

$$\lim_{\sigma, \tau \to \infty} \widetilde{\mathcal{M}}(y \mid x) = \mathcal{M}(y \mid x). \tag{17}$$

Increasing σ eventually leads to exact approximations of the indicator functions, while increasing τ leads to a completely unimodal softmax distribution. It should be noted that our approximation is not intended to replace the original model but rather to create a differentiable version of the model from which we can generate counterfactual examples through optimization. In practice, the original model would still be used to make predictions and the approximation would solely be used to generate counterfactual examples.

Figure 2 shows an intuitive illustration of our differentiable tree approximation using a two-feature ensemble with three trees. On the left is the decision boundary for a standard tree ensemble; the middle visualizes the positive leaf nodes that form the decision boundary; on the right is the approximated loss $\widetilde{\mathcal{L}_{\mathcal{M}}}$ and its gradient w.r.t. $\bar{x}$. The gradients push features close to thresholds harder and in the direction of the decision boundary if $\widetilde{\mathcal{L}}$ is convex.

In summary, the FOCUS approach generates counterfactual examples through optimization by performing gradient descent on an approximate loss $\widetilde{\mathcal{L}}$ that depends on both the original model $\mathcal{M}$ and its differentiable counterpart $\widetilde{\mathcal{M}}$ (see Equation 7). The gradient is taken w.r.t. the perturbation $\bar{x}$, i.e., $\nabla_{\bar{x}} \widetilde{\mathcal{L}}(x, \bar{x} \mid \mathcal{M}, d)$. For each x, we only consider $\bar{x}$ that have a different prediction as x (i.e., $y_x \neq y_{\bar{x}}$) to be valid counterfactual examples (see Equation 1), and the resulting counterfactual explanations Δ_x^* are simply the difference between the original x and the counterfactual example $\bar{x}$. We note that in our experiments FOCUS was able to find valid counterfactual examples for all instance across all 42 tested settings (see Sections 5, 6).
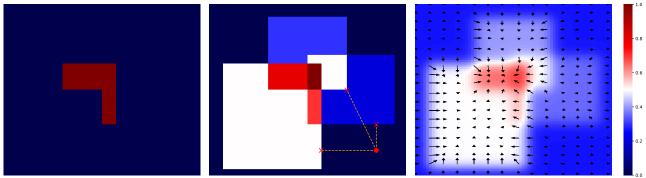
Figure 2: An example of how the Feature Tweaking (FT) baseline method (explained in Section 5.1) and our FOCUS method handle an adaptive boosting ensemble with three trees. Left: decision boundary of the ensemble. Middle: three positive leaves that form the decision boundary, an example instance and the perturbed examples suggested by FT. Right: approximated loss $\widetilde{\mathcal{L}_M}$ and its gradient w.r.t. $\bar{x}$. The FT perturbed examples do not change the prediction of the forest, whereas the gradient of the differentiable approximation leads toward the true decision boundary.

4 EXPERIMENTAL SETUP

We find the best counterfactual explanations by tuning the hyperparameters of FOCUS using Adam [21] for 1,000 iterations in 42 experimental settings (Experiment 1: 36 settings, Experiment 2: 6 settings). The hyperparameters are σ (the steepness of the sigmoid function in Equation 14); τ (the temperature of the softmax in Equation 16); β (the trade-off parameter in Equation 7; and α (the learning rate of Adam). We choose the parameters that (i) produce a valid counterfactual example for every instance in the dataset, and (ii) yield the smallest mean distance between corresponding pairs $(x, \bar{x})$.

4.1 Datasets and Models

We evaluate FOCUS on four binary classification tasks using the following datasets: *Wine Quality* [32], *HELOC* [10], COMPAS [25], and *Shopping* [33]. The *Wine Quality* dataset (4,898 instances, 11 features) is about predicting the quality of white wine on a 0–10 scale. We adapt this to a binary classification setting by labelling the wine as "high quality" if the quality is ≥ 7. The *HELOC* set (10,459 instances, 23 features) is from the Explainable Machine Learning Challenge at NeurIPS 2017, where the task is to predict whether or not a customer will default on their loan. The *COMPAS* dataset (6,172 instances, 6 features) is used for detecting bias in ML systems, where the task is predicting whether or not a criminal defendant will reoffend upon release. The *Shopping* dataset (12,330 instances, 9 features) entails predicting whether or not an online website visit results in a purchase. We scale all features such that their values are in the range [0, 1] and remove categorical features.

We train three types of tree-based models on 70% of each dataset: Decision Trees (DTs), Random Forests (RFs), and Adaptive Boosting (AB) with DTs as the base learners. We use the remaining 30% to (i) choose the best hyperparameter settings for the original tree-based models, and (ii) find counterfactual examples for this test set. In total we have 12 models (4 datasets × 3 tree-based models).

4.2 Distance Metrics

In our experiments, we generate different types of counterfactual explanations using different types of distance functions. We note that the flexibility of FOCUS allows for the use of any differentiable distance function.

Euclidean distance measures the geometric displacement:

$$d_{Euclidean}(x, \bar{x}) = \sqrt{\sum_i (x_i - \bar{x}_i)^2}. \qquad (18)$$

Cosine distance measures the angle by which $\bar{x}$ deviates from x – whether $\bar{x}$ preserves the relationship between features in x:

$$d_{Cosine}(x, \bar{x}) = 1 - \frac{\sum_i (x_i \cdot \bar{x}_i)}{\|x\| \|\bar{x}\|}. \qquad (19)$$

Manhattan distance (i.e., $L1$-norm) measures per feature differences, minimizing the number of features perturbed and therefore inducing sparsity:

$$d_{Manhattan}(x, \bar{x}) = \sum_i |x_i - \bar{x}_i|. \qquad (20)$$

When comparing against DACE [17], we use the Mahalanobis distance, since this is the distance function used in their novel cost function (see Equation 25):

$$d_{Mahalanobis}(x, \bar{x}|C) = \sqrt{(x - \bar{x})C^{-1}(x - \bar{x})}. \qquad (21)$$

C is the covariance matrix of x and $\bar{x}$, which allows us to account for correlations between features. When all features are uncorrelated, the Mahalanobis distance is equal to the Euclidean distance.

4.3 Evaluation Metrics

We evaluate the counterfactual examples produced by FOCUS based on how close they are to the original input using three metrics, in terms of four distance functions (see Section 4.2). The first evaluation metric is distance from the original input averaged over all examples, d_{mean}. Let X be the set of N original instances and $\bar{X}$ be the corresponding set of N generated counterfactual examples. The

mean distance is defined as:

$$d_{mean}(X, \bar{X}) = \frac{1}{N} \sum_{n=1}^{N} d(x^{(n)}, \bar{x}^{(n)}). \tag{22}$$

The second evaluation metric is mean relative distance from the original input, d_{Rmean}. This metric helps us interpret individual improvements over the baselines; if $d_{Rmean} < 1$, FOCUS's counterfactual examples are on average closer to the original input compared to the baseline. Let $\bar{X}$ be the set of counterfactual examples produced by FOCUS and let $\bar{X}'$ be the set of counterfactual examples produced by a baseline. Then the mean relative distance is defined as:

$$d_{Rmean}(\bar{X}, \bar{X}') = \frac{1}{N} \sum_{n=1}^{N} \frac{d(x^{(n)}, \bar{x}^{(n)})}{d(x^{(n)}, \bar{x}'^{(n)})}. \tag{23}$$

The third evaluation metric is the proportion of FOCUS's counterfactual examples that are closer to the original input in comparison to the baselines. For d we consider the Euclidean, Cosine and Manhattan distance functions.

5 EXPERIMENT 1: FOCUS VS. FT AND RP

In this experiment we compare FOCUS to the Feature Tweaking (FT) method by Tolomei et al. [31] and to a Random Perturbation (RP) baseline in terms of the evaluation metrics in Section 4.3: (i) mean distance, (ii) mean relative distance, and (iii) proportion of counterfactual examples that are closer to the original input. We consider 36 experimental settings (4 datasets × 3 tree-based models × 3 distance functions). All hyperparameter settings are listed in the Appendix.

5.1 Baseline: Feature Tweaking

FT identifies the leaf nodes where the prediction of the leaf nodes do not match the original prediction y_x. In other words, it recognizes the set of leaves that if activated, $t_j(\bar{x}) = 1$, would change the prediction of a tree $\mathcal{T}$:

$$\mathcal{T}_{change} = \left\{ j \mid j \in \mathcal{T}_{leaf} \wedge y_x \neq \arg\max_y T(y \mid j) \right\}. \tag{24}$$

For every $\mathcal{T}$ in $\mathcal{M}$, FT generates a perturbed example per node in $\mathcal{T}_{change}$ so that it is activated with at least an ϵ difference per threshold, and then selects the most optimal example (i.e., the one closest to the original instance). For every feature threshold θ_j involved, the corresponding feature is perturbed accordingly: $\bar{x}_{f_j} = \theta_j \pm \epsilon$. The result is a perturbed example that was changed minimally to activate a leaf node in $\mathcal{T}_{change}$. In our experiments, we test $\epsilon \in \{0.001, 0.005, 0.01, 0.1\}$, and choose the ϵ that minimizes the mean distance to the original input, while maximizing the number of counterfactual examples generated.

The main problem with FT is that the perturbed examples are not necessarily counterfactual examples, since changing the prediction of a single tree $\mathcal{T}$ does not guarantee a change in the prediction of the full ensemble $\mathcal{M}$. Figure 2 shows all three perturbed examples generated by FT for a single instance. In this case, none of the generated examples change the model prediction and therefore none are valid counterfactual examples.
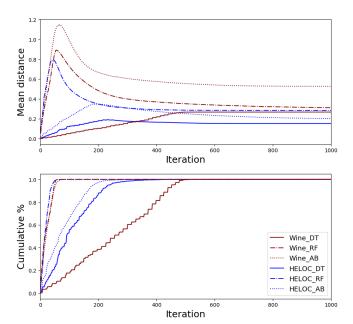

Figure 3: Mean distance (top) and cumulative % (bottom) of counterfactual examples in each iteration of FOCUS for Manhattan explanations.

5.2 Baseline: Random Perturbation

We also compare against a Random Perturbation (RP) baseline where noise is randomly sampled from a Gaussian $\mathcal{N}(0, 0.5)$ and added to the original input x. We use RP to generate 1,000 samples and select the $\bar{x}$ that minimizes the distance to x.

5.3 Results

We consider whether FOCUS explanations are more optimal (i.e., minimal) than the FT and RP baselines (see Table 1). FOCUS outperforms RP in all 36 settings, in terms of all three evaluation metrics, and the difference in d_{mean} is always statistically significant ($\alpha < 0.05$). In terms of d_{mean}, FOCUS outperforms FT in 20 settings while FT outperforms FOCUS in 8 settings. The difference in d_{mean} is not significant in the remaining 8 settings. In general, FOCUS outperforms FT in settings using Euclidean and Cosine distance because in each iteration, FOCUS perturbs many of the features by a small amount. Since FT perturbs only the features associated with an individual leaf, we expected that it would perform better for Manhattan distance but our results show that this is not the case – there is no clear winner between FT and FOCUS. We also see that FOCUS usually outperforms FT in settings using Random Forests (RF) and Adaptive Boosting (AB), while the opposite is true for Decision Trees (DT).

In most settings, the method with the lower d_{mean} also has the lower d_{Rmean} and the higher $\%_{closer}$. However, this is not always the case (i.e., *COMPAS* results), which highlights the importance of using multiple metrics to evaluate counterfactual examples. FOCUS outperforms FT in 22/36 settings in terms of d_{Rmean}, and in 24/36 settings in terms of $\%_{closer}$.

Figure 3 shows the mean Manhattan distance of the perturbed examples in each iteration of FOCUS, along with the proportion of perturbations resulting in valid counterfactual examples found for two datasets (we omit the others due to space considerations). These trends are indicative of all settings: the mean distance increases until a counterfactual example has been found for every x, after which the mean distance starts to decrease. This seems to be a result of the hinge-loss in FOCUS, which first prioritizes finding a valid counterfactual example (see Equation 1), then decreasing the distance between x and $\bar{x}$.

Our results for Experiment 1 show that FOCUS is effective and efficient for finding counterfactual explanations for tree-based models. Unlike the FT baseline, FOCUS finds valid counterfactual explanations for *every* instance across all settings. In the majority of tested settings, FOCUS explanations are substantial improvements in terms of distance to the original inputs.

6 EXPERIMENT 2: FOCUS VS. DACE

In this experiment, we compare FOCUS to the Distribution-Aware Counterfactual Explanations (DACE) method [17]. The flexibility of FOCUS allows us to plug in our choice of differentiable distance function: here, we use the Mahalanobis distance for both (i) generation of FOCUS explanations, and (ii) evaluation in comparison to DACE, since this is the distance function used in the DACE loss function (Equation 25). We use the same evaluation metrics from Section 4.3, as in Experiment 1.

We found two main limitations of DACE: (i) in all of our settings, it can only generate counterfactual examples for a subset of the test set, and (ii) it is limited by the size of the tree-based model. All hyperparameter settings are listed in the Appendix.

6.1 Baseline: DACE

DACE generates counterfactual examples that account for the underlying data distribution through a novel cost function using Mahalanobis distance (see Equation 21) and a local outlier factor (LOF):

$$d_{DACE}(x,\bar{x}|X,C) = d_{Mahalanobis}^2(x,\bar{x}|C) + \lambda q_k(x,\bar{x}|X), \quad (25)$$

where C is the covariance matrix, q_k is the k-LOF [5], X is the training set, and λ is the trade-off parameter. The k-LOF measures the degree to which an instance is an outlier in the context of its k-nearest neighbors.[2] To generate counterfactual examples, DACE formulates the task as a mixed-integer linear optimization problem and uses the CPLEX Optimizer[3] to solve it. The q_k term in the loss function penalizes counterfactual examples that are outliers, and therefore decreasing λ results in a greater number of counterfactual examples. In our experiments, we test $\lambda \in \{0.001, 0.01, 0.1, 0.5, 1.0\}$, and choose the λ that minimizes the mean distance to the original input, while maximizing the number of counterfactual examples generated.

We were only able to run DACE on 6 out of our 12 models because the problem size is too large (i.e., there are too many model parameters for DACE) for the remaining 6 models when using the free Python API of CPLEX (the optimizer used in DACE). Specifically, we were unable to run DACE on the following settings:

- Wine Quality AB (100 trees, max depth 4)
- Wine Quality RF (500 trees, max depth 4)
- HELOC RF (500 trees, max depth 4)
- HELOC AB (100 trees, max depth 8)
- COMPAS RF (500 trees, max depth 4)
- Shopping RF (500 trees, max depth 8).

Therefore, when comparing against DACE, we have 6 experimental settings (6 models × 1 distance function). We note that these are not unreasonable model sizes, and that unlike DACE, FOCUS can be applied to all 12 models (see Table 1).

6.2 Results

We consider whether FOCUS explanations are more minimal than DACE explanations. Table 2 shows the results for the 6 settings we could run DACE on. We found that DACE can only generate counterfactual examples for a small subset of the test set, regardless of the λ-value, as opposed to FOCUS, which can generate counterfactual examples for the entire test set in all cases. To compute d_{mean}, d_{Rmean}, and $\%_{closer}$, we compare FOCUS and DACE only on the instances DACE was able to generate a counterfactual example for. We find that FOCUS significantly outperforms DACE in 5 out of 6 settings in terms of all three evaluation metrics, indicating that FOCUS explanations are indeed more minimal than those produced by DACE. FOCUS is also more reliable since (i) it is not restricted by model size, and (ii) it can generate counterfactual examples for all instances in the test set.

7 DISCUSSION

In this section we examine the fidelity of FOCUS approximations, look at a specific example of a FOCUS explanation, and investigate the effect of different distance functions in a brief case-study. We also discuss the limitations of counterfactual explanations in general.

7.1 Fidelity of Approximations

Fidelity is commonly used to evaluate XAI methods that are based on approximations of a given model: it is a measure of the agreement between the original model $\mathcal{M}$ and its approximation $\widetilde{\mathcal{M}}$:

$$fid(\widetilde{\mathcal{M}}, X) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\left[y_{x^{(n)}} = \arg\max_{y'} \widetilde{\mathcal{M}}(y'|x^{(n)})\right], \quad (26)$$

where $y_x = \arg\max_y \mathcal{M}(y \mid x)$. In our case, the purpose of the approximations is not to replace the original models, but rather to construct differentiable versions of the models so we can generate counterfactual examples through gradient-based optimization. Examining the fidelity is meant as a *sanity check* – to ensure our approximations are reasonably representative of the original model.

Table 3 shows the fidelity of the model approximations used in our experiments: a value of 1 indicates perfect alignment between $\mathcal{M}$ and $\widetilde{\mathcal{M}}$. We see that the alignment is at least 0.7, which indicates that FOCUS approximations are indeed reasonable representations of the original model – both in terms of their inner workings (i.e., same tree structure, same features, same splitting thresholds but "softer" versions) as well as their predictions.

[2]We use $k = 1$ in our experiments, since this is the value of k that is supported in the code kindly provided to us by the authors, for which we are very grateful.
[3]http://www.ibm.com/analytics/cplex-optimizer

Table 1: Experiment 1: Evaluation results for comparing FOCUS, FT and RP counterfactual examples. Significant improvements and losses over the baselines are denoted by ▼ and ▲, respectively ($p < 0.05$, two-tailed t-test,); ° denotes no significant difference; ⊗ denotes settings where the baselines cannot find a counterfactual example for every instance.

Dataset	Metric	Method	Euclidean			Cosine			Manhattan		
			DT	RF	AB	DT	RF	AB	DT	RF	AB
Wine Quality	d_{mean}	RP	1.191	1.158	1.166	0.266	0.290	0.254	3.276	3.229	3.111
		FT	0.269	**0.174**	0.267⊗	0.030	0.017	0.034⊗	0.269	**0.223**	0.382⊗
		FOCUS	**0.268**▼°	0.188▼▲	0.188▼▼	**0.003**▼▼	**0.008**▼▼	**0.014**▼▼	**0.268**▼°	0.312▼▲	**0.360**▼▼
	d_{Rmean}	FOCUS/RP	0.186	0.130	0.108	0.009	0.023	0.044	0.068	0.079	0.087
		FOCUS/FT	0.990	1.256	0.649	0.066	0.821	0.312	0.990	1.977	0.924
	$\%_{closer}$	FOCUS <RP	100%	100%	100%	100%	100%	100%	100%	100%	100%
		FOCUS <FT	100%	21.0%	87.5%	100%	80.8%	95.1%	100%	5.4%	58.6%
HELOC	d_{mean}	RP	1.638	1.647	1.654	0.260	0.267	0.267	5.834	5.842	5.775
		FT	**0.120**	0.210	0.185	0.003	0.008	0.007	**0.135**	**0.278**	**0.198**
		FOCUS	0.133▼▲	**0.186**▼▼	**0.136**▼▼	**0.001**▼▼	**0.002**▼▼	**0.001**▼▼	0.152▼▲	0.284▼°	0.203▼°
	d_{Rmean}	FOCUS/RP	0.073	0.101	0.074	0.003	0.005	0.003	0.024	0.043	0.032
		FOCUS/FT	1.169	0.942	0.907	0.303	0.285	0.421	1.252	1.144	1.364
	$\%_{closer}$	FOCUS <RP	100%	100%	100%	100%	100%	100%	100%	100%	100%
		FOCUS <FT	16.6%	57.9%	71.9%	91.6%	91.5%	92.9%	51.3%	43.6%	24.2%
COMPAS	d_{mean}	RP	0.816	0.809	0.812	0.436	0.427	0.410	1.488	1.460	1.484
		FT	**0.082**	**0.075**	0.081	0.013	0.014	0.015	**0.086**	**0.078**	**0.085**
		FOCUS	0.092▼▲	0.079▼°	**0.076**▼▼	**0.008**▼▼	**0.011**▼▼	**0.007**▼▼	0.093▼▲	0.085▼°	0.090▼°
	d_{Rmean}	FOCUS/RP	0.115	0.090	0.093	0.025	0.027	0.019	0.066	0.054	0.060
		FOCUS/FT	1.162	1.150	1.062	0.473	0.965	0.539	1.182	1.236	1.155
	$\%_{closer}$	FOCUS <RP	100%	99.9%	100%	100%	100%	99.9%	100%	100%	100%
		FOCUS <FT	29.4%	22.6%	44.8%	82.7%	68.0%	84.8%	65.8%	36.2%	66.9%
Shopping	d_{mean}	RP	0.963	1.015	0.994	0.587	0.580	0.606	2.000	2.079	2.014
		FT	**0.119**	0.028	0.126⊗	**0.050**	0.027	0.131⊗	**0.121**	0.030	0.142⊗
		FOCUS	0.142▼▲	**0.025**▼▼	**0.028**▼▼	0.055▼▲	**0.013**▼▼	**0.006**▼▼	0.128▼°	**0.026**▼▼	**0.046**▼▼
	d_{Rmean}	FOCUS/RP	0.049	0.027	0.031	0.048	0.025	0.014	0.022	0.013	0.024
		FOCUS/FT	1.051	1.053	0.218	0.795	0.482	0.074	0.944	0.796	0.312
	$\%_{closer}$	FOCUS <RP	99.9%	100%	100%	99.9%	99.9%	100%	100%	100%	100%
		FOCUS <FT	40.2%	36.1%	99.6%	44.4%	86.1%	99.5%	55.8%	81.9%	97.1%

7.2 Case Study: Credit Risk

As a practical example, we investigate what FOCUS explanations look like for individuals in the HELOC dataset. Here, the task is to predict whether or not an individual will default on their loan. This has consequences for loan approval: individuals who are predicted as defaulting will be denied a loan. For these individuals, we want to understand how they can change their profile such that they are approved. For example, given an individual who has been denied a loan from a bank, a counterfactual explanation could be:

> Your loan application has been denied. In order to have
> your loan application approved, you need to (i) increase
> your ExternalRiskEstimate score by 62, and (ii) decrease
> your NetFractionRevolvingBurden by 58

Figure 4 shows four counterfactual explanations generated using different distance functions for the same individual and same model. We see that the Manhattan explanation only requires a few changes

to the individual's profile, but the changes are large. In contrast, the individual changes in the Euclidean explanation are smaller but there are more of them. In settings where there are significant dependencies between features, the Cosine explanations may be preferred since they are based on perturbations that try to preserve the relationship between features. For instance, in the *Wine Quality* dataset, it would be difficult to change the amount of citric acid without affecting the pH level. The Mahalanobis explanations would be useful when it is important to take into account not only correlations between features, but also the training data distribution. This flexibility allows users to choose what kind of explanation is best suited for their problem.

Different distance functions can result in different *magnitudes* of feature perturbations as well as different *directions*. For example, the Cosine explanation suggests increasing *PercentTradesWBalance*, while the Mahalanobis explanations suggests decreasing it. This
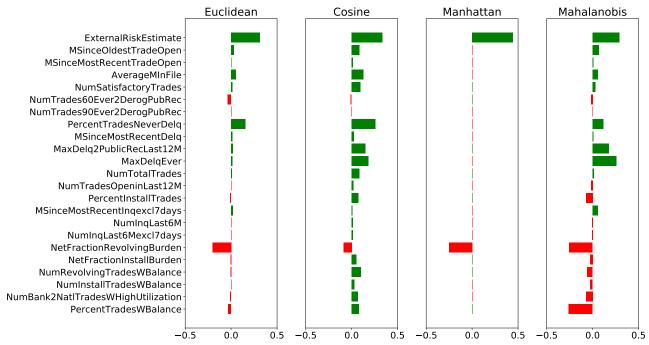
Figure 4: FOCUS explanations for the same model and same x based on different distance functions. Green and red indicate increases and decreases in feature values, respectively. Perturbation values are based on normalized feature values. Left: Euclidean explanation perturbs several features, but only slightly. Middle Left: Cosine explanation perturbs almost all of the features. Middle Right: Manhattan explanation perturbs two features substantially. Right: Mahalanobis explanation perturbs almost all of the features.

Table 2: Experiment 2: Evaluation results for comparing FOCUS and DACE counterfactual examples in terms of Mahalanobis distance (see Equation 21). Significant improvements over the baseline are denoted by ▾ ($p < 0.05$, two-tailed t-test,). ° denotes no significant difference.

Metric	Method	Wine DT	HELOC DT	COMPAS DT	COMPAS AB	Shopping DT	Shopping AB
d_{mean}	DACE	1.325	1.427	0.814	1.570	0.050	3.230
	FOCUS	**0.542**▾	**0.810**▾	**0.776**°	**0.636**▾	**0.023**▾	**0.303**▾
d_{Rmean}	FOCUS / DACE	0.420	0.622	1.18	0.372	0.449	0.380
$\%_{closer}$	FOCUS < DACE	100%	94.5%	29.9%	96.1%	99.4%	90.8%
# CFs found	DACE	241	1342	842	700	362	448
	FOCUS	1470	3138	1852	1852	3699	3699
# obs in	dataset	1470	3138	1852	1852	3699	3699

Table 3: Fidelity of FOCUS approximations used in Experiments 1 and 2: — denotes models we were unable to run DACE on.

Dataset	Model	Euclid.	Cosine	Manhat.	Mahal.
Wine Quality	DT	0.836	0.836	0.836	0.900
	RF	0.940	0.940	0.940	—
	AB	0.926	0.926	0.926	—
HELOC	DT	0.836	0.836	0.836	0.930
	RF	0.954	0.887	0.887	—
	AB	0.936	0.744	0.905	—
COMPAS	DT	0.844	0.894	0.807	0.807
	RF	0.742	0.809	0.700	—
	AB	0.922	0.922	0.814	0.814
Shopping	DT	0.902	0.906	0.902	0.902
	RF	0.810	0.780	0.871	—
	AB	0.919	0.919	0.919	0.919

is because the loss space of the underlying RF model is highly non-convex, and therefore there is more than one way to obtain an alternative prediction. In this case, both options result in valid counterfactual examples.

We examine the Manhattan explanation in more detail. We see that FOCUS suggests two main changes: (i) increasing the *ExternalRiskEstimate*, and (ii) decreasing the *NetFractionRevolvingBurden*. We obtain the definitions and expected trends from the data dictionary [11] created by the authors of the dataset. The *ExternalRiskEstimate* is a "consolidated version of risk markers" (i.e., a credit score). A higher score is better: as one's *ExternalRiskEstimate*

increases, the probability of default decreases. The *NetFractionRevolvingBurden* is the "revolving balance divided by the credit limit" (i.e., utilization). A lower value is better: as one's *NetFractionRevolvingBurden* increases, the probability of default increases. We find that the changes suggested by FOCUS are fairly consistent with the expected trends in the data dictionary [11], as opposed to suggesting nonsensical changes such as increasing one's utilization to decrease the probability of default.

7.3   Limitations of Counterfactual Explanations

In Section 7.2, we show an example of FOCUS Manhattan explanations that identifies two features that need to be changed in order to achieve a positive outcome: increase the credit score and decrease the utilization. However, the plausibility of either of these changes depends greatly on the context, since increasing one's credit score requires knowledge of the components of the credit score (along with the ability to change those components), and decreasing one's utilization depends on one's existing financial commitments. Although FOCUS explanations cannot tell a user precisely how to increase their credit score, they do provide the user with the knowledge that their credit score is too low for a loan approval, which empowers them to ask questions about how the score was calculated (i.e., how the risk markers were consolidated). We can postulate that decreasing one's utilization is fairly actionable, but the degree to which an individual can actually change their spending habits is completely dependent on their specific situation. For example, an individual who is only responsible for themselves might have more control over their spending compared to someone who has several dependents. In either case, we argue that deciding what is (not) actionable is not a decision for the developer to make, but for the individual who is affected by the decision.

To position these reflections more broadly, we recall the downfalls of counterfactual explanations as noted by Barocas et al. [3]: (i) perturbations do not necessarily map to real-world actions, (ii) normalizing features based on the training data does not necessarily make them commensurable, (iii) perturbations that have a positive effect in one domain can have a negative effect in another, and (iv) underlying models in production are not necessarily stable or monotonic, and do not always have binary outcomes. These aspects are especially problematic when counterfactual explanations are applied as an off-the-shelf solution without regard for the specific context in which they are used. That is not what we advocate.

Counterfactual explanations should not be viewed as an ultimate solution to achieving recourse, but rather as a "menu of options" [3] that is used as part of a human-in-the-loop process. FOCUS is meant to serve as an effective, efficient, and reliable option for the algorithmic component of the human-in-the-loop process as opposed to constituting the entire process on its own.

8   CONCLUSION

We propose a local explanation method for tree-based classifiers, FOCUS which casts the problem of finding counterfactual examples as a gradient-based optimization task and provides a differentiable approximation of tree-based models to be used in the optimization framework. Given an input instance x, FOCUS generates an optimal counterfactual example based on the minimal perturbation to the input instance x which results in an alternative prediction from a model $\mathcal{M}$. Unlike previous methods that assume that $\mathcal{M}$ is differentiable, we propose a solution for when $\mathcal{M}$ is a non-differentiable, tree-based model that provides a differentiable approximation of $\mathcal{M}$ that can be used to find counterfactual examples through gradient descent. In the majority of experiments, examples generated by FOCUS are significantly closer to the original instances in terms of three different evaluation metrics compared to those generated by the baselines. FOCUS is able to generate valid counterfactual examples for all instances across all datasets, and the resulting explanations are flexible depending on the distance function.

Future work involves including additional criteria in the loss function as well as conducting a user study to determine how this, along with varying the distance functions, impacts user preferences for counterfactual explanations. Another direction is to apply an approach similar to FOCUS to other non-differentiable models and thereby enabling counterfactual explanations for an even wider range of models. Finally, while existing research from the cognitive sciences has shown that humans are able to interpret counterfactual explanations, the notion of what constitutes a *minimal* perturbation is currently not clear [7]. Further research into the interpretability and cognitive efficacy of counterfactual explanations could help the field better understand the appropriate criteria to optimize for.

REPRODUCIBILITY

To facilitate the reproducibility of the results reported in this work, our code for the experimental implementation of FOCUS is available at http://github.com/a-lucic/focus.

REFERENCES

[1] Ahmed M. Alaa, Michael Weisz, and Mihaela van der Schaar. 2017. Deep Counterfactual Networks with Propensity-Dropout. *arXiv preprint arXiv:1706.05966* (June 2017).

[2] Randall Balestriero. 2017. Neural Decision Trees. *arXiv preprint arXiv:1702.07360* (Feb. 2017).

[3] Solon Barocas, Andrew D. Selbst, and Manish Raghavan. 2019. The Hidden Assumptions Behind Counterfactual Explanations and Principal Reasons. *arXiv:1912.04930 [cs]* (Dec. 2019). https://doi.org/10.1145/3351095.3372830 arXiv: 1912.04930.

[4] Battista Biggio and Fabio Roli. 2018. Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning. *Pattern Recognition* 84 (Dec. 2018), 317–331. https://doi.org/10.1016/j.patcog.2018.07.023 arXiv: 1712.03141.

[5] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2020. LOF: Identifying Density-Based Local Outliers. *ACM SIGMOD* (2020).

[6] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. 2018. Adversarial Patch. *arXiv:1712.09665 [cs]* (May 2018). http://arxiv.org/abs/1712.09665 arXiv: 1712.09665.

[7] Ruth M.J. Byrne. 2016. Counterfactual Thought. *Annual Review of Psychology* 67 (2016), 135–157.

[8] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives. *arXiv preprint arXiv:1802.07623* (Feb. 2018).

[9] Finale Doshi-Velez and Been Kim. 2017. Towards a Rigorous Science of Interpretable Machine Learning. *arXiv preprint arXiv:1702.08608v2* (2017).

[10] FICO. 2017. Explainable Machine Learning Challenge. https://community.fico.com/s/explainable-machine-learning-challenge.

[11] FICO. 2017. FICO xML Challenge. https://github.com/5teffen/FICO-xML-Challenge/tree/master/xML%20Challenge%20Dataset%20and%20Data%20Dictionary

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572 [cs, stat]* (March 2015). http://arxiv.org/abs/1412.6572 arXiv: 1412.6572.

[13] Rory Mc Grath, Luca Costabello, Chan Le Van, Paul Sweeney, Farbod Kamiab, Zhao Shen, and Freddy Lecue. 2018. Interpretable Credit Application Predictions With Counterfactual Explanations. *arXiv preprint arXiv:1811.05245* (Nov. 2018).

[14] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local Rule-Based Explanations of Black Box Decision Systems. *arXiv preprint arXiv:1805.10820* (May 2018).

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2014. Distilling the Knowledge in a Neural Network. *NIPS 2014 Deep Learning Workshop* (March 2014).

[16] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. 2019. Towards Realistic Individual Recourse and Actionable Explanations in Black-Box Decision Making Systems. *AISTATS* (2019).

[17] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Hiroki Arimura. 2020. DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization. *IJCAI* (2020), 2855–2862. https://doi.org/10.24963/ijcai.2020/395

[18] Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. 2019. Model-Agnostic Counterfactual Explanations for Consequential Decisions. *AISTATS* (2019).

[19] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. 2020. Algorithmic Recourse: from Counterfactual Explanations to Interventions. *arXiv:2002.06278 [cs, stat]* (2020).

[20] Amir-Hossein Karimi, Julius von Kügelgen, Bernhard Schölkopf, and Isabel Valera. 2020. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. *ICML Workshop on Human Interpretability in Machine Learning* (2020).

[21] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (Jan. 2017).

[22] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. 2017. Inverse Classification for Comparison-based Interpretability in Machine Learning. *arXiv preprint arXiv:1712.08443* (Dec. 2017).

[23] Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere. 2019. Explainable Reinforcement Learning Through a Causal Lens. *arXiv preprint arXiv:1905.10958* (May 2019).

[24] Ramaravind Kommiya Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. *ACM FAccT* (2020).

[25] Dan Ofer. 2017. COMPAS Dataset. https://www.kaggle.com/danofer/compass.

[26] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. 2020. FACE: Feasible and Actionable Counterfactual Explanations. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* (2020). https://doi.org/10.1145/3375627.3375850

[27] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why Should I Trust You?: Explaining the Predictions of Any Classifier. In *KDD*. ACM, 1135–1144.

[28] Chris Russell. 2019. Efficient Search for Diverse Coherent Explanations. *arXiv preprint arXiv:1901.04909* (Jan. 2019).

[29] Jiawei Su, Danilo Vasconcellos Vargas, and Sakurai Kouichi. 2019. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Transactions on Evolutionary Computation* 23, 5 (Oct. 2019), 828–841. https://doi.org/10.1109/TEVC.2019.2890858 arXiv: 1710.08864.

[30] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing Properties of Neural Networks. *arXiv:1312.6199 [cs]* (Feb. 2014). http://arxiv.org/abs/1312.6199 arXiv: 1312.6199.

[31] Gabriele Tolomei, Fabrizio Silvestri, Andrew Haines, and Mounia Lalmas. 2017. Interpretable Predictions of Tree-based Ensembles via Actionable Feature Tweaking. (2017), 465–474.

[32] UCI. 2009. Wine Quality Data Set. https://archive.ics.uci.edu/ml/datasets/Wine+Quality.

[33] UCI. 2019. Online Shoppers Intention Dataset. https://archive.ics.uci.edu/ml/datasets/Online+Shoppers+Purchasing+Intention+Dataset.

[34] Berk Ustun, Alexander Spangher, and Yang Liu. 2019. Actionable Recourse in Linear Classification. (2019). https://doi.org/10.1145/3287560.3287566

[35] Arnaud Van Looveren and Janis Klaise. 2020. Interpretable Counterfactual Explanations Guided by Prototypes. *arXiv:1907.02584* (2020).

[36] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2018. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *Harvard Journal of Law & Technology* 31, 2 (2018), 841–888.

[37] Yongxin Yang, Irene Garcia Morillo, and Timothy M. Hospedales. 2018. Deep Neural Decision Trees. *arXiv preprint arXiv:1806.06988* (June 2018).