Flexible Interpretability through Optimizable Counterfactual Explanations for Tree Ensembles

Anonymous author(s), Submission ID #2

Abstract

Model interpretability has become an important problem in machine learning (ML) due to the increased effect algorithmic decisions have on humans. Providing users with counterfactual explanations (CF) can help them understand not only why ML models make certain decisions, but also how these decisions can be changed. We extend previous work that could only be applied to differentiable models by introducing probabilistic model approximations in the optimization framework. We find that our CF examples are significantly closer to the original instances compared to other methods specifically designed for tree ensembles.

1. INTRODUCTION

As machine learning (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 (Doshi-Velez & Kim, 2017). For an individual who is affected by the predictions of these models, it would be useful to have an explanation that provides insight into how these decisions can be *changed*, in order to achieve recourse. Counterfactual (CF) explanations are a natural solution to the recourse 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 CF explanation could be: "*Had your income been* €1000 *higher, you would have been approved for the loan*." We focus on finding *optimal* CF explanations – the *minimal* changes to the input required to change the outcome.

CF explanations are based on CF examples: generated instances that are close to an existing instance but have an alternative prediction. The difference between the original instance and the CF example is the CF explanation. A shortcoming of CF explanation methods is that they can provide potentially unactionable explanations, which is why

Wachter et al. (2018) propose framing the problem as an optimization task, but their work assumes that the underlying machine learning models are differentiable, which excludes an important class of widely applied and highly effective non-differentiable models: tree ensembles. Our method relaxes this assumption and builds upon the work of Wachter et al. by introducing differentiable approximations of tree ensembles that can be used in such an optimization framework. Alternative non-optimization approaches for generating CF explanations for tree ensembles involve an extensive search over many possible paths in the ensemble that could lead to an alternative prediction (Tolomei et al., 2017). This leads us to our main research question: *Are CF examples generated by our method closer to the original input instances than those generated by existing heuristic methods?*

Algorithmic recourse is a line of research that is closely related to CF explanations, except that these methods include the additional restriction that the resulting explanation must be *actionable* (Ustun et al., 2019; Joshi et al., 2019; Karimi et al., 2020a;b). 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 $\rightarrow$ 50 or change *marital_status* from MARRIED $\rightarrow$ 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 (Ustun et al., 2019) or differentiable (Joshi et al., 2019) models, or (ii) require access to causal information (Karimi et al., 2020a;b), 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 (Barocas et al., 2019). 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 CF "menu of options" (Barocas et al., 2019) that can be customized based on the chosen distance function. Although CF 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. CF 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. CF 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. METHOD

A *CF 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 CF 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 CF 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 CF explanation* Δ_x^* is:

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

This definition aligns with previous ML work on CF explanations (Laugel et al., 2017; Karimi et al., 2019; Tolomei et al., 2017). 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 CF 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 CF 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 $\mathcal{M}$.

2.1. Our Method: FOCUS

Wachter et al. (2018) recognized that CF 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 CF 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. (2018) 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:

$$\widetilde{\mathcal{L}}(x, \bar{x} \mid \mathcal{M}, d) = \widetilde{\mathcal{L}_{\mathcal{M}}}(y_x, \bar{x}) + \beta \cdot \mathcal{L}_d(x, \bar{x}). \tag{7}$$

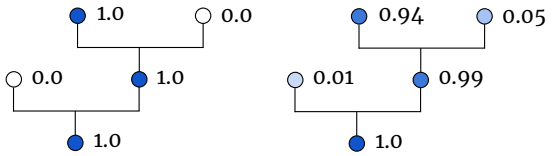

Figure 1: Left: A decision tree $\mathcal{T}$ and per node activations for a single instance. Right: a differentiable approximation of the same tree $\widetilde{\mathcal{T}}$ and activations for the same instance.

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 CF 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 (see Appendix D for intuitive illustrations). 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 CF examples through optimization. In practice, the original model would still be used to make predictions and the approximation would solely be used to generate CF examples.

3. EXPERIMENTAL SETUP

We consider 42 experimental settings to find the best CF explanations by tuning the hyperparameters of FOCUS ($w_d, \alpha, \sigma, \tau$) using Adam (Kingma & Ba, 2017) for 1,000

iterations. We choose the hyperparameters that produce (i) a valid CF example for every instance in the dataset, and (ii) the smallest mean distance between corresponding pairs $(x, \bar{x})$.

We evaluate FOCUS on four binary classification datasets: *Wine Quality* (UCI, 2009), *HELOC* (FICO, 2017b), *COMPAS* (Ofer, 2017), and *Shopping* (UCI, 2019). For each dataset, we train three types of tree-based models: Decision Trees (DTs), Random Forests (RFs), and Adaptive Boosting (AB) with DTs as the base learners. See Appendix E for more details on the datasets and training setup, including hyperparameter settings.

We compare against two baselines that generate CF examples for tree ensembles based on the inner workings of the model: Feature Tweaking (FT) (Tolomei et al., 2017) and Distribution-Aware CF Explanations (DACE) (Kanamori et al., 2020). See Appendix A and B for detailed descriptions of the baselines.

We evaluate the CF examples produced by FOCUS based on how close they are to the original input using three metrics. Mean distance, d_{mean}, measures the distance from the original input, averaged over all examples. Mean relative distance, d_{Rmean}, measures pointwise ratios of distance to the original input. This helps us interpret individual improvements over the baselines; if $d_{Rmean} < 1$, FOCUS's CF examples are on average closer to the original input compared to the baseline. We also evaluate the proportion of FOCUS's CF examples that are closer to the original input compared to the baselines ($\%_{closer}$). We test the metrics in terms of four distance functions: Euclidean, Cosine, Manhattan and Mahalanobis. See Appendix F for exact calculations.

4. Experiment 1: FOCUS vs. FT

We consider 36 experimental settings (4 datasets $\times$ 3 tree-based models $\times$ 3 distance functions) when comparing FOCUS to FT. The results are listed in Table 1. In terms of d_{mean}, FOCUS outperforms Feature Tweaking (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 for Manhattan distance. 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). Overall, we find that FOCUS is effective and efficient for finding CF explanations for tree-based models. Unlike the FT baseline, FOCUS finds valid CF explanations for *every* instance across all settings. In the majority of tested settings, FOCUS's explanations are substantial improvements in terms of distance to the original inputs, across all three metrics.

5. Experiment 2: FOCUS vs. DACE

DACE is based on mixed-integer linear optimization and uses the CPLEX Optimizer[1] to solve the problem. We were only able to run DACE[2] 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. 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 (see Equation 19 in Appendix B). Therefore, when comparing against DACE, we have 6 experimental settings (6 models $\times$ 1 distance function). See Appendix B.1 for more details.

Table 2 shows the results for the 6 settings we could run DACE on. We found that DACE can only generate CF examples for a small subset of the test set, regardless of the λ-value, as opposed to FOCUS, which can generate CF 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 for which DACE was able to generate a CF example. 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 CF examples for all instances in the test set.

6. Case Study: Credit Risk

As a practical example, we investigate what FOCUS explanations look like for instances in the HELOC dataset, where 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 CF 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 2 shows four CF 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

[1] http://www.ibm.com/analytics/cplex-optimizer

[2] Based on code kindly provided to us by the authors.

large in magnitude. 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 relationships between features (e.g., in the *Wine Quality* dataset, it would be difficult to change the amount of citric acid without affecting the pH level). The Mahalanobis explanation 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 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 CF examples.

7. RELATED WORK

Previous XAI methods for generating CF examples are either model-agnostic (Poyiadzi et al., 2020; Karimi et al., 2019; Laugel et al., 2017; Van Looveren & Klaise, 2020; Mothilal et al., 2020) or model-specific (Wachter et al., 2018; Grath et al., 2018; Tolomei et al., 2017; Kanamori et al., 2020; Russell, 2019; Dhurandhar et al., 2018). 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. Our work is a model-specific approach for generating CF examples through optimization. Previous model-specific work for generating CF examples through optimization has solely been conducted on differentiable models (Wachter et al., 2018; Grath et al., 2018; Dhurandhar et al., 2018). Although we focus on tree ensembles, our method can be applied to any non-differentiable model that can be approximated probabilistically. See Appendix H for an in-depth discussion of methods on algorithmic recourse, model-specific CF examples, adversarial examples, local explanations and differentiable trees.

8. CONCLUSION

We propose an optimization-based CF explanation method for tree-based classifiers, FOCUS. We find that examples generated by FOCUS are often 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 CF examples for all instances across all datasets, and the resulting explanations are flexible depending on the distance function.

References

Balestriero, R. Neural decision trees. *arXiv preprint arXiv:1702.07360*, February 2017.

Barocas, S., Selbst, A. D., and Raghavan, M. The Hidden Assumptions Behind Counterfactual Explanations and Principal Reasons. *ACM FAT**, 2019.

Biggio, B. and Roli, F. Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning. *Pattern Recognition*, 84:317–331, December 2018. ISSN 00313203. doi: 10.1016/j.patcog.2018.07.023. URL http://arxiv.org/abs/1712.03141. arXiv: 1712.03141.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. LOF: Identifying Density-Based Local Outliers. *ACM SIGMOD*, 2020.

Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. Adversarial Patch. *NIPS*, May 2018. URL http://arxiv.org/abs/1712.09665. arXiv: 1712.09665.

Dhurandhar, A., Chen, P.-Y., Luss, R., Tu, C.-C., Ting, P., Shanmugam, K., and Das, P. Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives. *NIPS*, February 2018.

Doshi-Velez, F. and Kim, B. Towards a Rigorous Science of Interpretable Machine Learning. *arXiv preprint arXiv:1702.08608v2*, 2017.

FICO. Fico xml challenge, 2017a. URL https://github.com/5teffen/FICO-xML-Challenge/tree/master/xML%20Challenge%20Dataset%20and%20Data%20Dictionary.

FICO. Explainable Machine Learning Challenge. https://community.fico.com/s/explainable-machine-learning-challenge, 2017b.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and Harnessing Adversarial Examples. *ICLR*, March 2015. URL http://arxiv.org/abs/1412.6572. arXiv: 1412.6572.

Grath, R. M., Costabello, L., Van, C. L., Sweeney, P., Kamiab, F., Shen, Z., and Lecue, F. Interpretable Credit Application Predictions With Counterfactual Explanations. *NIPS Workshop on Challenges and Opportunities for AI in Financial Services*, November 2018.

Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., and Giannotti, F. Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820*, May 2018.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *NIPS 2014 Deep Learning Workshop*, March 2014.

Joshi, S., Koyejo, O., Vijitbenjaronk, W., Kim, B., and Ghosh, J. Towards Realistic Individual Recourse and Actionable Explanations in Black-Box Decision Making Systems. *AISTATS*, 2019.

Kanamori, K., Takagi, T., Kobayashi, K., and Arimura, H. DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization. *IJCAI*, pp. 2855–2862, 2020. doi: 10.24963/ijcai.2020/395.

Karimi, A.-H., Barthe, G., Balle, B., and Valera, I. Model-Agnostic Counterfactual Explanations for Consequential Decisions. *AISTATS*, 2019.

Karimi, A.-H., Schölkopf, B., and Valera, I. Algorithmic Recourse: from Counterfactual Explanations to Interventions. *arXiv:2002.06278 [cs, stat]*, 2020a.

Karimi, A.-H., von Kügelgen, J., Schölkopf, B., and Valera, I. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. *ICML Workshop on Human Interpretability in Machine Learning*, 2020b.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *ICLR*, January 2017.

Laugel, T., Lesot, M.-J., Marsala, C., Renard, X., and Detyniecki, M. Inverse Classification for Comparison-based Interpretability in Machine Learning. *arXiv preprint arXiv:1712.08443*, December 2017.

Mothilal, R. K., Sharma, A., and Tan, C. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. *ACM FAccT*, 2020.

Ofer, D. COMPAS Dataset. https://www.kaggle.com/danofer/compass, 2017.

Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., and Flach, P. FACE: Feasible and Actionable Counterfactual Explanations. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020. doi: 10.1145/3375627.3375850.

Ribeiro, M. T., Singh, S., and Guestrin, C. Why should I trust you?: Explaining the predictions of any classifier. In *KDD*, pp. 1135–1144. ACM, 2016.

Russell, C. Efficient Search for Diverse Coherent Explanations. *FAT**, January 2019.

Su, J., Vargas, D. V., and Kouichi, S. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, October 2019. ISSN 1089-778X, 1089-778X, 1941-0026. doi: 10.1109/TEVC.2019.2890858. URL http://arxiv.org/abs/1710.08864. arXiv: 1710.08864.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv:1312.6199 [cs]*, February 2014. URL http://arxiv.org/abs/1312.6199. arXiv: 1312.6199.

Tolomei, G., Silvestri, F., Haines, A., and Lalmas, M. Interpretable Predictions of Tree-based Ensembles via Actionable Feature Tweaking. *KDD*, pp. 465–474, 2017.

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

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

Ustun, B., Spangher, A., and Liu, Y. Actionable Recourse in Linear Classification. *ACM FAT**, 2019.

Van Looveren, A. and Klaise, J. Interpretable Counterfactual Explanations Guided by Prototypes. *arXiv:1907.02584*, 2020.

Wachter, S., Mittelstadt, B., and Russell, C. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *Harvard Journal of Law & Technology*, 31(2):841–888, 2018.

Yang, Y., Morillo, I. G., and Hospedales, T. M. Deep neural decision trees. *ICML Workshop on Human Interpretability in Machine Learning*, June 2018.

A. Detailed Results of Experiment 1

In Experiment 1, we compare against the FT baseline. The results are shown in Table 1. 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 $\mathcal{T}$:

$$\mathcal{T}_{change} = \left\{ j \mid j \in \mathcal{T}_{leaf} \wedge y_x \neq \arg\max_y T(y \mid j) \right\}. \tag{18}$$

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 CF examples generated.

The main problem with FT is that the perturbed examples are not necessarily CF 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 3 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 CF examples.

B. Detailed Results of Experiment 2

In Experiment 2, we compare against DACE. The results are shown in Table 2. DACE generates CF examples that account for the underlying data distribution through a novel cost function using Mahalanobis distance and a local outlier factor (LOF):

$$d_{DACE}(x, \bar{x}|X, C) = {d_{Mahalanobis}}^2(x, \bar{x}|C) + \lambda q_k(x, \bar{x}|X), \tag{19}$$

where C is the covariance matrix, q_k is the k-LOF (Breunig et al., 2020), 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.[3] We refer the reader to the original paper for a more detailed overview of this cost function. The q_k term in the loss function penalizes CF examples that are outliers, and therefore decreasing λ results in a greater number of CF 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 CF examples generated. The main issue with DACE is that even for very small values of α, it is unable to generate CF examples for the majority of instances in the test sets (see Table 2).

[3] 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.

B.1. Models DACE Cannot Run On

We were unable to run DACE on the following settings because the problem size is too large when using the free Python API of CPLEX:

- 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).

We note that these are not unreasonable model sizes, and that unlike DACE, FOCUS can be applied to all 12 models (see Table 1).

C. Case Study: Credit Risk

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 (FICO, 2017a) 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 (FICO, 2017a), as opposed to suggesting nonsensical changes such as increasing one's utilization to decrease the probability of default.

Decreasing one's utilization is completely dependent on the specific situation: an individual who only supports themselves might have more control over their spending in comparison to someone who has multiple dependents. An individual can decrease their utilization in two ways: (i) decreasing their spending, or (ii) increasing their credit limit (or a combination of the two).

We can postulate that (i) is more "actionable" compared to (ii), since (ii) is usually a decision made by a financial institution. However, the degree to which an individual can actually

Table 1: Experiment 1: Evaluation results for comparing FOCUS and FT CF examples. Significant improvements and losses over the baseline (FT) are denoted by $^{\blacktriangledown}$ and $^{\blacktriangle}$, respectively ($p < 0.05$, two-tailed t-test,); $^{\circ}$ denotes no significant difference; $^{\otimes}$ denotes settings where the baseline cannot find a CF example for every instance.

			Euclidean			**Cosine**			**Manhattan**		
Dataset	**Metric**	**Method**	**DT**	**RF**	**AB**	**DT**	**RF**	**AB**	**DT**	**RF**	**AB**
Wine	d_{mean}	*FT*	0.269	**0.174**	0.267$^{\otimes}$	0.030	0.017	0.034$^{\otimes}$	0.269	**0.223**	0.382$^{\otimes}$
		FOCUS	**0.268**$^{\circ}$	0.188$^{\blacktriangle}$	**0.188**$^{\blacktriangledown}$	**0.003**$^{\blacktriangledown}$	**0.008**$^{\blacktriangledown}$	**0.014**$^{\blacktriangledown}$	**0.268**$^{\circ}$	0.312$^{\blacktriangle}$	**0.360**$^{\blacktriangledown}$
Quality	d_{Rmean}	*FOCUS/FT*	0.990	1.256	0.649	0.066	0.821	0.312	0.990	1.977	0.924
	$\%_{closer}$	*FOCUS <FT*	100%	21.0%	87.5%	100%	80.8%	95.1%	100%	5.4%	58.6%
HELOC	d_{mean}	*FT*	**0.120**	0.210	0.185	0.003	0.008	0.007	**0.135**	**0.278**	**0.198**
		FOCUS	0.133$^{\blacktriangle}$	**0.186**$^{\blacktriangledown}$	**0.136**$^{\blacktriangledown}$	**0.001**$^{\blacktriangledown}$	**0.002**$^{\blacktriangledown}$	**0.001**$^{\blacktriangledown}$	0.152$^{\blacktriangle}$	0.284°	0.203°
	d_{Rmean}	*FOCUS/FT*	1.169	0.942	0.907	0.303	0.285	0.421	1.252	1.144	1.364
	$\%_{closer}$	*FOCUS <FT*	16.6%	57.9%	71.9%	91.6%	91.5%	92.9%	51.3%	43.6%	24.2%
COMPAS	d_{mean}	*FT*	**0.082**	**0.075**	0.081	0.013	0.014	0.015	**0.086**	**0.078**	**0.085**
		FOCUS	0.092$^{\blacktriangle}$	0.079°	**0.076**$^{\blacktriangledown}$	**0.008**$^{\blacktriangledown}$	**0.011**$^{\blacktriangledown}$	**0.007**$^{\blacktriangledown}$	0.093$^{\blacktriangle}$	0.085°	0.090°
	d_{Rmean}	*FOCUS/FT*	1.162	1.150	1.062	0.473	0.965	0.539	1.182	1.236	1.155
	$\%_{closer}$	*FOCUS <FT*	29.4%	22.6%	44.8%	82.7%	68.0%	84.8%	65.8%	36.2%	66.9%
Shopping	d_{mean}	*FT*	**0.119**	0.028	0.126$^{\otimes}$	**0.050**	0.027	0.131$^{\otimes}$	**0.121**	0.030	0.142$^{\otimes}$
		FOCUS	0.142$^{\blacktriangle}$	**0.025**$^{\blacktriangledown}$	**0.028**$^{\blacktriangledown}$	0.055$^{\blacktriangle}$	**0.013**$^{\blacktriangledown}$	**0.006**$^{\blacktriangledown}$	0.128°	**0.026**$^{\blacktriangledown}$	**0.046**$^{\blacktriangledown}$
	d_{Rmean}	*FOCUS/FT*	1.051	1.053	0.218	0.795	0.482	0.074	0.944	0.796	0.312
	$\%_{closer}$	*FOCUS <FT*	40.2%	36.1%	99.6%	44.4%	86.1%	99.5%	55.8%	81.9%	97.1%

Table 2: Experiment 2: Evaluation results for comparing FOCUS and DACE CF examples in terms of Mahalanobis distance. Significant improvements over the baseline are denoted by $^{\blacktriangledown}$ ($p < 0.05$, two-tailed t-test,). $^{\circ}$ denotes no significant difference.

		Wine	**HELOC**	**COMPAS**		**Shopping**	
Metric	**Method**	**DT**	**DT**	**DT**	**AB**	**DT**	**AB**
d_{mean}	*DACE*	1.325	1.427	0.814	1.570	0.050	3.230
	FOCUS	**0.542**$^{\blacktriangledown}$	**0.810**$^{\blacktriangledown}$	**0.776**$^{\circ}$	**0.636**$^{\blacktriangledown}$	**0.023**$^{\blacktriangledown}$	**0.303**$^{\blacktriangledown}$
d_{Rmean}	*FOCUS / DACE*	0.420	0.622	1.18	0.372	0.449	0.380
	DACE						
$\%_{closer}$	*FOCUS < DACE*	100%	94.5%	29.9%	96.1%	99.4%	90.8%
	DACE						
# CFs found	*DACE*	241	1,342	842	700	362	448
	FOCUS	1,470	3,138	1,852	1,852	3,699	3,699
# obs in dataset	—	1,470	3,138	1,852	1,852	3,699	3,699

change their spending habits is completely dependent on their specific situation: an individual who only supports themselves might have more control over their spending in comparison to someone who has multiple 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. CF examples should be used as part of a human-in-the-loop system and not as a final solution. The individual should know that utilization is an important component of the model, even if it is not necessarily "actionable" for them. We also note that it is unclear how exactly an individual would change their credit score without further insight into how the score was calculated (i.e., how the risk markers were consolidated). It should be noted that this is not a shortcoming of FOCUS, but rather of using features that are uninterpretable on their own, such as credit scores. Although FOCUS explanations cannot tell a user precisely how to increase their credit score, it is still important for

the individual to know that their credit score is an important factor in determining their probability of getting a loan, as this empowers them to ask questions about how the score was calculated (i.e., how the risk markers were consolidated).

D. Intuitive Illustrations of FOCUS

Figure 3 shows how FOCUS and FT handle an adaptive boosting ensemble 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.

Figure 4 shows the mean Manhattan distance of the perturbed examples in each iteration of FOCUS, along with the proportion of perturbations resulting in valid CF 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 CF 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 CF example (see Equation 1), then decreasing the distance between x and $\bar{x}$.

E. Datasets and Models

We evaluate FOCUS on four binary classification tasks using the following datasets: *Wine Quality* (UCI, 2009), *HELOC* (FICO, 2017b), COMPAS (Ofer, 2017), and *Shopping* (UCI, 2019). 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 CF examples for this test set. In total we have 12 models (4 datasets $\times$ 3 tree-based models).

F. Evaluation Metrics

Here we show exact calculations for mean distance (d_{mean}) and mean relative distance (d_{Rmean}). Let X be the set of N original instances and $\bar{X}$ be the corresponding set of N generated CF examples. The mean distance is defined as:

$$d_{mean}(X, \bar{X}) = \frac{1}{N} \sum_{n=1}^{N} d(x^{(n)}, \bar{x}^{(n)}). \tag{20}$$

Let $\bar{X}$ be the set of CF examples produced by FOCUS and let $\bar{X}'$ be the set of CF examples produced by a baseline. Then the mean relative distance is defined as:

$$d_{Rmean}(\bar{X}, \bar{X}') = \frac{1}{N} \sum_{n=1}^{N} \frac{d(x^{(n)}, \bar{x}^{(n)})}{d(x^{(n)}, \bar{x}'^{(n)})}. \tag{21}$$

G. 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 $\widetilde{\mathcal{M}}$:

$$\mathit{fid}(\widetilde{\mathcal{M}}, X) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\left[y_{x^{(n)}} = \arg\max_{y'} \widetilde{\mathcal{M}}(y' | x^{(n)}) \right], \tag{22}$$

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 CF 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.

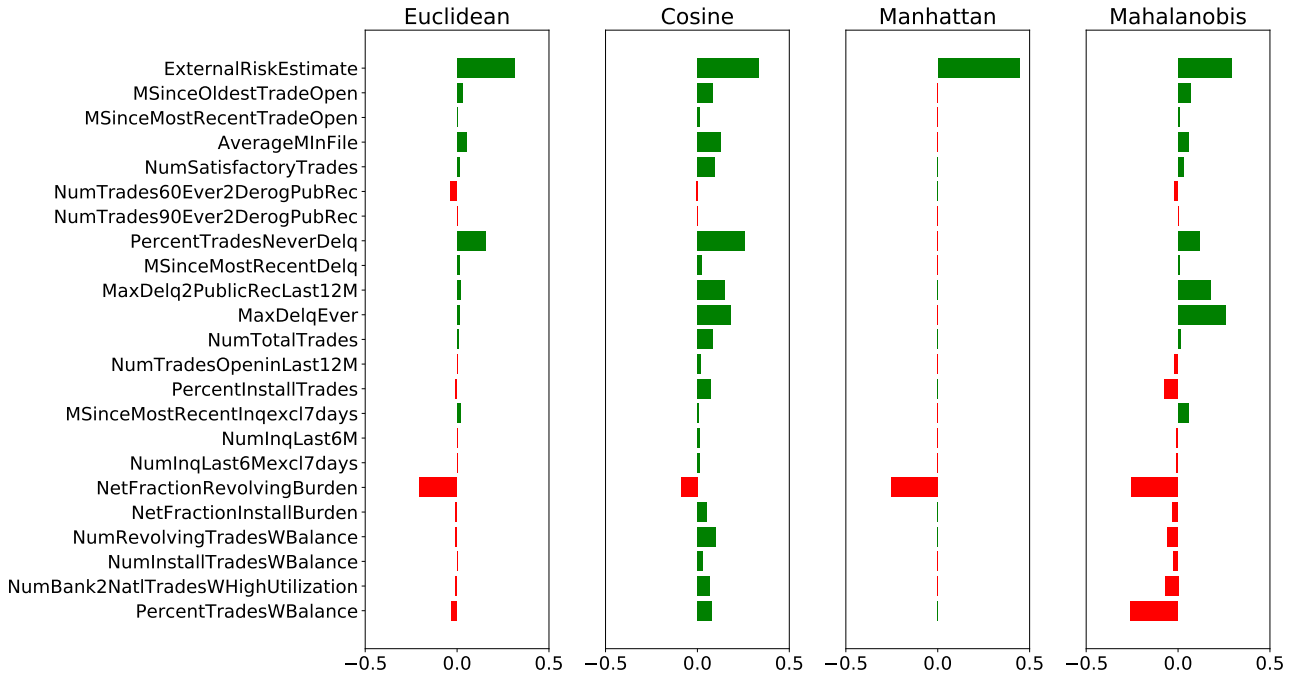

Figure 2: FOCUS explanations for the same model and same x based on different distance functions. Green and red indicate increases and decreases in feature values, respectively. Perturbation values are based on normalized feature values. Left: Euclidean explanation perturbs several features, but only 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.

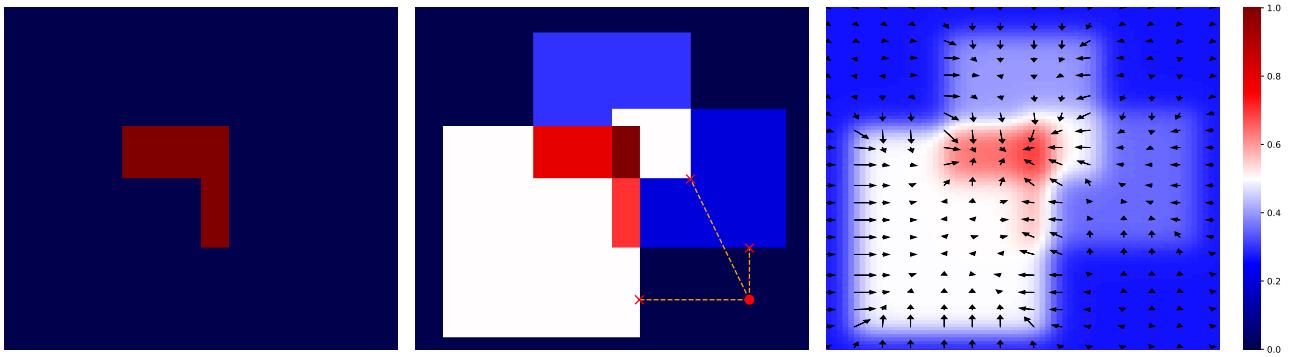

Figure 3: An example of how the FT baseline method (explained in Section A) and our FOCUS method handle an adaptive boosting ensemble with three trees. Left: decision boundary of the ensemble. Middle: three positive leaves that form the decision boundary, an example instance and the perturbed examples suggested by FT. Right: approximated loss $\widetilde{\mathcal{L}_{\mathcal{M}}}$ 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.

H. Related Work

H.1. Model-specific CF Examples

We propose a model-specific approach for generating CF examples. Previous model-specific work has either been done through optimization (**?**Grath et al., 2018; Dhurandhar et al., 2018), mixed integer programming (Russell, 2019; Kanamori et al., 2020), or heuristic search (Tolomei et al., 2017). Some of this work is applicable to linear models or neural networks (**?**Grath et al., 2018; Dhurandhar et al., 2018; Russell, 2019), while other work can be applied to tree-ensembles (Tolomei et al., 2017; Kanamori et al., 2020).

Our work is the first model-specific approach for generating CF examples for tree ensembles through gradient-based optimization. We compare our method against existing approaches for tree-ensembles (Tolomei et al., 2017; Kanamori et al., 2020), and refer the reader to Appendix **??**

Table 3: Fidelity of FOCUS approximations used in Experiments 1 and 2: — denotes models we were unable to run DACE on. The parameters σ and τ are chosen per setting based on which values produce the best CF examples. As a result, different approximations of the same model are used when optimizing for different distance functions, but in some cases (e.g., *Wine Quality* DT), the approximations are the same regardless of the distance function.

Dataset	Model	Euclid.	Cosine	Man.	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

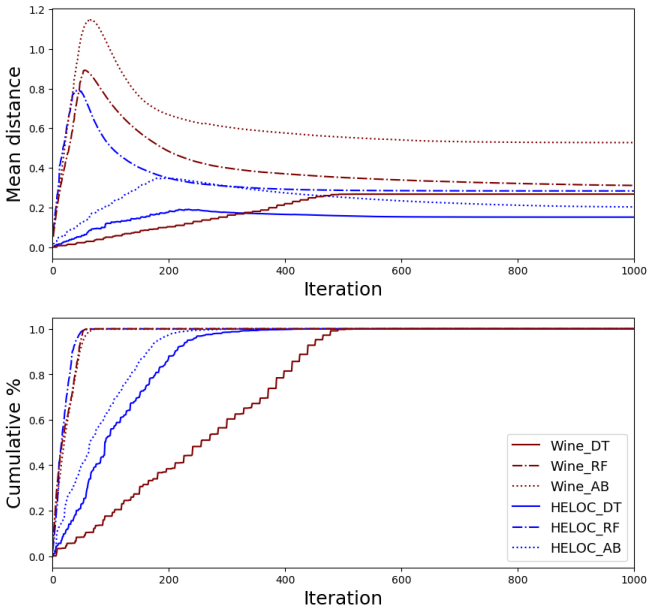

Figure 4: Mean distance (top) and cumulative % (bottom) of CF examples in each iteration of FOCUS for Manhattan explanations.

for a more detailed overview of these methods.

H.2. Adversarial Examples

Adversarial examples are a type of CF 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 (Goodfellow et al., 2015; Szegedy et al., 2014; Su et al., 2019; Brown et al., 2018); a more complete overview can be found in (Biggio & Roli, 2018).

The main difference between adversarial examples and CF examples is in the intent: adversarial examples are meant to *fool* the model, whereas CF examples are meant to *explain* the model.[4] Another difference is that adversarial examples are typically studied in the context of image classification, whereas CF 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.

H.3. 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 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 (Guidotti et al., 2018; Ribeiro et al., 2016). 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.

H.4. 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 (Balestriero, 2017; Yang et al., 2018) 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

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

an obvious choice for finding CF examples for an existing model. Soft decision trees (Hinton et al., 2014) 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.