Cost-Aware Counterfactuals for Black Box Explanations

Natalia Martinez Gil
IBM Research
Yorktown Heights, NY, USA
natalia.martinez.gil@ibm.com

Kanthi Sarpatwar
IBM Research
Yorktown Heights, NY, USA
sarpatwa@ibm.com

Sumanta Mukherjee
IBM Research
Yorktown Heights, NY, USA
sumanm03@ibm.com

Roman Vaculin
IBM Research
Yorktown Heights, NY, USA
vaculin@ibm.com

Abstract

Counterfactual explanations provide actionable insights into the minimal change in a system that would lead to a more desirable prediction from a black box model. We address the challenges of finding valid and low cost counterfactuals in the setting where there is a different cost or preference for perturbing each feature. We propose a multiplicative weight approach that is applied on the perturbation, and show that this simple approach can be easily adapted to obtain multiple diverse counterfactuals, as well as to integrate the importance features obtained by other state of the art explainers to provide counterfactual examples. Additionally, we discuss the computation of valid counterfactuals with numerical gradient-based methods when the black box model presents flat regions with no reliable gradient. In this scenario, sampling approaches, as well as those that rely on available data, sometimes provide counterfactuals that may not be close to the decision boundary. We show that a simple long-range guidance approach, which consist of sampling from a larger radius sphere in search of a direction of change for the black box predictor when no gradient is available, improves the quality of the counterfactual explanation. In this work we discuss existing approaches, and show how our proposed alternatives compares favourably on different datasets and metrics.

1 Introduction

Complex machine learning models such as deep neural networks (DNNs) or large ensembles of models have become increasingly popular due to their state of the art performance across multiple prediction tasks. However, their inference process is hard to interpret, and turns them, in practice, into black-box models. On the other hand, less accurate but more interpretable models such as generalized linear models or decision trees are easier to trust in high-risk domains such as finance or health care. Explainability in Artificial Intelligence (XAI) [1, 2, 3, 4, 5] has gained popularity on the promise of bridging the gap between these two seemingly conflicting goals. Combining the accuracy of complex machine learning models with the interpretability of simpler models.

An exciting direction that has emerged recently is that of *Counterfactual* explanations [6]. Here, one tries to explain a model outcome in terms of the minimal perturbation needed to *change* the model outcome (potentially to a more desirable one) for a given instance or input. In certain cases, this can also be viewed as an *actionable insight* providing recourse [7] to a user that can

XAI in Action: Past, Present, and Future Applications @ NeurIPS 2023.

now understand what could be changed in the current state of a system in order to reach a more favourable prediction. This is different to attribution based explanations such as LIME [8] and SHAP [9, 10] which generate an importance vector indicating which features are the most relevant for the model's current prediction, but may not indicate a direction of recourse. Moreover, counterfactuals are locally exact explanations, in the sense that they have been evaluated on the predictive model and therefore have no uncertainty in the outcome they would produce.

Finding counterfactuals requires navigating the input space in search of the decision boundary of a predictive model. Here, we focus on post-hoc black box counterfactual methods which only have access to the model's inference function. This is in contrast to recent in-training counterfactual approaches such as [11, 12], where the counterfactual generation model is trained jointly with the predictive model. In the post-hoc black box setting finding a valid counterfactual with numerical gradient based methods becomes challenging with the existence of *flat regions* in the model's output, where the gradient vanishes entirely. This uncertainty in explanation generation is unacceptable in practice, especially given the knowledge that there is always a valid counterfactual for any given instance.

In this work we consider the problem of finding counterfactuals when there is a different cost for perturbing each feature, denoted as "cost-aware counterfactuals". These costs can be associated to a specific user preference, e.g., a user must provide recourse but wants to take into account that changing some features would be more feasible than others. Moreover, features costs can be informed by other XAI importance feature methods as a way to simplify the search for CF or evaluate consistency across explainers [13, 14]. Cost-aware counterfactuals can be directly applied to generate diverse counterfactuals for a given instance, which compose a set of valid counterfactuals for the same base instance that perturb different combinations of input features.

Main contributions:

1. We consider the problem of "cost-aware counterfactuals", where there is a preference over which features should be perturbed. We describe a natural formulation of this problem that can be used with any standard counterfactual algorithm, and empirically demonstrate that the formulation effectively finds counterfactuals in the promoted directions. We leverage this approach to sequentially generate a set of diverse counterfactuals for a given instance. We compare the diversity and quality of the generated solutions against three black box approaches based on DiCE [15]. We show how "cost-aware counterfactuals" can be used to analyze existing importance feature based explainers such as SHAP [9, 10] and LIME [8].
2. We review and compare the performance of different state of the art counterfactual generation algorithms for black box models. We report that, while gradient estimation approaches typically yield best results, these approaches do not always guarantee the recovery of a *counterfactual*. Sampling based methods or proto-guided approaches addresses the problem of finding *valid counterfactuals* effectively, but they do so at the expense of the quality of the solution. We address this trade-off by using a hybridized approach, called `gradsearch`, which effectively combines the advantages of both methodologies.

2 Background

We consider the machine learning classification setting, where a model $f : \mathrm{R}^d \rightarrow \Delta^Y$ outputs a vector in the simplex of probabilities over $|Y|$ from an input $X \quad X$, where $X \quad \mathrm{R}^d$. Given an instance $\mathbf{x}_0 \quad X$, and a distance function $d : \mathrm{R}^d \times \mathrm{R}^d \rightarrow \mathrm{R}_+$ in the input space, the counterfactual objective is to find the closest sample to instance $\mathbf{x}_0$ that changes the outcome of the classification model. This is formalized as follows,

$$\mathbf{x}_{cf} = \arg\min_{\mathbf{x} \quad X} d(\mathbf{x}, \mathbf{x}_0) \quad s.t. \quad \arg\max_{i \quad Y} f_i(\mathbf{x}_0) = \arg\max_{j \quad Y} f_j(\mathbf{x}),^{1}. \tag{1}$$

A common choice is to define $\mathbf{x}_{cf}$ as the original instance plus an additive perturbation, $\mathbf{x}_{cf} = \mathbf{x}_0 + \delta_{cf}$. Moreover, $d(\mathbf{x}, \mathbf{x}_0)$ can be expressed as a regularization function on the additive

[1] Additionally, we can ask for a counterfactual of a particular class, $\arg\max_{j \quad Y} f_j(\mathbf{x_{cf}}) = y_{cf}$, as in [6]

perturbation, $R(\delta) = d(\mathbf{x}, \mathbf{x}_0)$, where $R(\delta)$ has a unique minimum on the null perturbation $\arg\min_\delta R(\delta) = 0$, e.g., elastic norm $||\delta||_2^2 + \beta||\delta||_1$. In this scenario, the objective presented in Eq. 1 becomes

$$\delta_{cf} = \arg \min_{\delta:\delta+\mathbf{x}_0 \in X} R(\delta) \quad s.t. \quad \arg\max_i f_i(\mathbf{x}_0) = \arg\max_j f_j(\mathbf{x}_0 + \delta). \tag{2}$$

Let us denote the predicted class for the given instance $\mathbf{x}_0$ to be $\hat{y}_0 = \arg\max_{i \in Y} f_i(\mathbf{x}_0)$. Then, the constrained problem in Eq. 2 can be formulated as the following optimization problem

$$\begin{aligned} &\min_{\delta:\delta+\mathbf{x}_0 \in X} \max_{\lambda} R(\delta) + \lambda C_{f,\mathbf{x}_0}(\delta), \\ &C_{f,\mathbf{x}_0}(\delta) = \left(f_{\hat{y}_0}(\mathbf{x}_0 + \delta) - \max_{i \in Y \setminus \hat{y}_0} f_i(\mathbf{x}_0 + \delta) \right)_+. \end{aligned} \tag{3}$$

Here $C_{f,\mathbf{x}_0}(\delta)$ measures how much more likely label $\hat{y}_0$ (the class predicted for $\mathbf{x}_0$) is under the perturbed instance $\mathbf{x}_0 + \delta$ than any other label; $[\cdot]_+$ denotes the ReLU operator. $C_{f,\mathbf{x}_0}(\delta)$ is a penalty function that enforces the counterfactual condition (i.e., change in the classification decision).

3 Cost-aware Counterfactuals

We consider the setting in which there is a preference over which features should be perturbed when searching for counterfactuals. Formally, we are given a vector of non-negative feature weights (or costs) $\mathbf{w} \in \mathrm{R}_+^{\mathrm{d}}$ with larger values on features that, ideally, should remain invariant. We then modify the regularization term with $R(\delta_w)$, were $\delta_w = \mathbf{w} \odot \delta$ denotes the element-wise product between each feature weight and its corresponding perturbation. We assume $R(\cdot)$ to be a convex regularization function with a minimum in 0. In this setting, the optimization objective in Eq. 3 becomes

$$\min_{\delta:\delta+\mathbf{x}_0 \in X} \max_{\lambda} R(\mathbf{w} \odot \delta) + \lambda \left(f_{\hat{y}_0}(\mathbf{x}_0 + \delta) - \max_{i \in Y \setminus \hat{y}_0} f_i(\mathbf{x}_0 + \delta) \right)_+. \tag{4}$$

We denote the above objective as $\mathrm{wCF}(\delta; \mathbf{w})$. Here $\mathbf{w}$ directly affects the partial derivatives of $R(\cdot)$ w.r.t. each feature perturbation δ_i, $i = 1, \ldots, d$. Features with larger weights are proportionally penalized for having large perturbations, and increasing the cost of a single feature can only maintain or decrease the module of the perturbation of that feature. This is proved and discussed in propositions A.1 and A.2 in Supplementary Material A.2. The cost, or weight vector $\mathbf{w}$ can be obtained by either a cost preference or its converse, an importance feature vector, to provide some guidance where larger values means smaller cost.

3.1 Generating Diverse Counterfactuals

We propose a sequential approach to generate multiple, diverse counterfactuals by leveraging the weighting vector introduced in wCF. Here the core idea is to iteratively recover cost-aware counterfactuals, and use the sequence of previous counterfactuals to update the current weighting vector $\mathbf{w}$. This promotes the generation of counterfactuals that do not modify features that were previously perturbed unless it is necessary. Starting with the uniform weight $\mathbf{w}_0 = \mathbf{1}^d$, we propose the following sequential update

$$\begin{aligned} \delta_{cf,t} &= \arg \min_{\delta:\delta+\mathbf{x}_0 \in X} \mathrm{wCF}(\delta; \mathbf{w}_t), \\ \mathbf{w}_{t+1} &= \Pi_{\Delta^{d-1}}[\mathbf{w}_t + \gamma |\delta_{cf,t}|]. \end{aligned} \tag{5}$$

Here $|\delta_{cf,t}|$ indicates the element-wise absolute value of the perturbations of the t-th counterfactual. $\gamma > 1$ is a factor that scales the aggregation of the previously perturbed features, and $\Pi_{\Delta^{d-1}}[.]$ denotes the euclidean projection operator into the simplex Δ^{d-1}. Note that, as the iterations increases, features that have not been perturbed see their corresponding weight decay to zero, which encourages these features to be perturbed for the subsequent counterfactual perturbation.

The update proposed in Eq 5 can be interpreted as a no-regret algorithm [16] where the adversary is trying to maximize the perturbation cost $\max_{\mathbf{w} \in \Delta^{d-1}} \mathbf{w}^T |\delta_{cf}|$, and a player is trying to minimize the objective from Eq 4. As t increases, the above process converges to a weighting vector $\mathbf{w}_T$ that presents a ranking of the most relevant features to flip the decision of the classifier, and it can be interpreted as an importance feature vector.

4 Finding valid counterfactuals

The goal of the objective presented in Eq. 3 and Eq.4 is to find a proximal counterfactual (i.e., closest to the boundary decision of f), and involves a maximization step on the penalty coefficient λ, needed to achieve a feasible solution, and a minimization step on the perturbation δ. In Appendix A.1 we show an implementation of a standard gradient-based approach to achieve Eq. 3 objective. Here the minimization step on δ requires the gradient estimation of $C_{f,\mathbf{x}_0}(\delta)$ and consequently of f.

We are interested in finding counterfactual examples for black box classifiers. Meaning that we only have access to the probability score predictions. Therefore, we need to estimate the gradient of $C_{f,\mathbf{x}_0}(\delta)$ numerically, or find an update direction when we reach a flat region, $f_i(\mathbf{x}) \quad 0$ for $i \quad Y$, and we are not feasible, $C_{f,\mathbf{x}_0}(\delta) > 0$. Flat regions occur in common model architectures, such as NN classifiers with softmax output layers for inputs that are well into the saturated (i.e., high confidence) region, and on tree-based classification models such as random forests. Recovering update directions in flat regions of the model is a necessary step in finding valid counterfactual examples.

Algorithm 1 gradsearch update direction

Input: $C_{f,\mathbf{x}_0}$ function, $\mathbf{x}_0$ instance, δ_t previous perturbation
Parameter:μ radius scalar, $c > 1$ scaling factor, n samples
Output: $\mathbf{v}_t$ update direction.

1: **if** $C_{f,\mathbf{x}_0}(\delta_t) = 0 \rightarrow$ **return** $\mathbf{0}^d$
2: $\{\mathbf{u}_j \quad U(\mathbf{S}^{d-1})\}_{j=1}^{n}$, $\mathbf{x} = \mathbf{x}_0 + \delta_t, \mathbf{v}_t = \mathbf{0}^d$
3: $\mu_j = \mu$ for $j = 1, ..., n$
4: **while** $||\mathbf{v}_t|| = 0$ **do**
5: $\quad \mu_j = \quad _X[\mu_j],\ j = 1, ..., n$
6: $\quad \mathbf{v}_t = d \quad \sum_{j=1}^{n} \frac{C_{f,\mathbf{x}_0}(\delta_t + \mu_j \mathbf{u}_j) - C_{f,\mathbf{x}_0}(\delta_t)}{n\mu_j} \mathbf{u}_j$,
7: $\quad \mu_j = \mu_j \times c$ for $j = 1, ..., n$,
8: **end while**
9: **return** $\mathbf{v}_t$

We propose gradsearch, an approach that combines numerical gradient estimation, with exploration within flat regions. Whenever our candidate counterfactual is in an infeasible region (i.e., $C_{f,\mathbf{x}_0}(\delta_t) > 0$), we try to estimate the numerical gradient, $C_{f,\mathbf{x}_0}(\delta)$. However, for flat regions, this estimation can be arbitrarily close to 0, indicating a lack of local update direction. We circumvent this limitation by increasing the search radius of the sampling ball (which no longer represents a local gradient) until we find a feasible search direction.

The gradsearch counterfactual perturbation update is described in Algorithm 1. Here, given $C_{f,\mathbf{x}_0}$, we estimate an update direction (see line 6) using a formulation equivalent to the two-point gradient estimator based on random directions uniformly sampled from the sphere, $\mathbf{S}^{d-1}$ in line 2. If μ is sufficiently small this update is equivalent to the numerical gradient estimation used in [17]. If this direction is the null vector we increase the radius by a factor c as shown in line 7 inside the while loop from line 4 to 7. Line 5 is a projection to keep each perturbation inside the feasible ranges for each feature (i.e., remain inside the input space X).

The provided update replaces line 5 in the penalty approach in Algorithm 2 from Appendix A.1. In our experiments, gradsearch proves to be better than the alternatives since it does not suffer from the limitations of numerical gradients and does not require a dataset.

5 Related Work

Counterfactual explanations [6, 18, 15, 19, 20, 21], sometimes referred to (with minor differences in definitions) as *pertinent negative contrastive explanations* [22, 17], are an active area of research. The work in [6] proposed finding counterfactual examples by optimizing for the minimum necessary perturbation for a given instance that modifies the model's prediction. Some of the commonly desired properties of a counterfactual method are validity (i.e., its success rate in providing counterfactuals for a given model), proximity of the provided solutions to the original instance, sparsity across the feature changes, and in-distribution or in-manifold solutions, among other properties [6, 23]. Here we focus on post-hoc black box counterfactual methods which only have access to the model's inference function. Finding valid counterfactuals for a given instance poses a challenge since it involves exploring the input space in search of a sample that changes

the model prediction while still being as close as possible to the original instance (based on some proximity/sparsity criteria).

Numerical gradients (numgrad), as in [17], rely on sampling directions from a uniform distribution over the unit sphere to estimate $C_{f,\mathbf{x}_0}(\delta)$ and obtain an update for δ. As mentioned in Section 4, if we are in an infeasible flat region, the numerical gradient is 0 and we do not have a way to move towards a feasible region. To overcome this issue, prototype guided methods (proto) [21] rely on a given counterfactual data sample, $\mathbf{x}_{proto}$, to include in Eq.3 a term that minimizes the distance between the $\mathbf{x}_{proto}$ and the counterfactual $\mathbf{x}_0 + \delta$. Moreover, if the initialization is set to be $\mathbf{x}_{proto}$ we start from a feasible region. However, the final solution may not have best quality in terms of R.

Purely random sampling methods (random), generate independent instances in the input space, and select the sample with the smallest perturbation (according to R) amongst valid counterfactuals. If a dataset containing counterfactuals is available, one can simply output the instance with the smallest R [24]. An implementation of this approach using kdtree, as well as the random method (see Eq 8) is available in the library provided by [15]. The proposed gradsearch can be seen as a variation of the Growing Spheres approach proposed in [25] since it also relies on sampling form the unit sphere. However, gradsearch approach starts with a small radius sphere μ and is equivalent to an unbiased estimator of the numerical gradient for an L-smooth function [26]. A detailed explanation of the above mentioned methods is available in Appendix A.1.

Method	Validity	$\frac{1}{d}\|\|\delta\|\|_1$	$\frac{1}{d}\|\|\delta\|\|_2$	$\frac{1}{d}\|\|\delta\|\|_0$
UCI Adult - 8 features				
NUMGRAD	0.74	.058±.068	.051±.067	.088±.069
RANDOM	**1.00**	.087±.042	.083±.045	.102±.034
KDTREE	**1.00**	.138±.061	.138±.061	.144±.068
PROTO	**1.00**	.061±.061	.057±.061	.084±.06
GRADSEARCH	**1.00**	**.04±.038**	**.034±.038**	**.067±.034**
German credit - 20 features				
NUMGRAD	0.55	**.01±.008**	**.003±.004**	**.042±.026**
RANDOM	**1.00**	.055±.028	.051±.029	.065±.024
KDTREE	**1.00**	.171±.052	.149±.052	.243±.058
PROTO	**1.00**	.051±.041	.044±.038	.074±.054
GRADSEARCH	**1.00**	.02±.018	.012±.017	.053±.031
Lending Club - 43 features				
NUMGRAD	0.43	**.002±.008**	.001±.006	**.019±.012**
RANDOM	**1.00**	.019±.024	.013±.017	.44±.025
KDTREE	**1.00**	.467±.121	.291±.152	1.0±0.0
PROTO	**1.00**	.05±.043	.033±.032	.109±.078
GRADSEARCH	**1.00**	.003±.01	**.001±.005**	.027±.033

Table 1: Validity (success rate) and counterfactual quality in terms of perturbation norm (L1-Proximity $\frac{1}{d}\|\|\delta\|\|_1$, L2-Proximity $\frac{1}{d}\|\|\delta\|\|_2$ and Sparsity $\frac{1}{d}\|\|\delta\|\|_0$) for the approaches discussed in Section 5 and extended in Appendix A.1. All methods use the softmax output of the FCNN. We present in bold the best results. Note that the quality of the numgrad perturbations is measured over the set of perturbations of valid counterfactuals, which is a considerably smaller set than the other approaches. The proposed gradsearch always succeeded in finding valid counterfactuals, had similar quality to numgrad but with considerably higher success rate (validity).

The problem of computing diverse counterfactuals systematically on linear models was proposed in [18] as an integer optimization problem. [15, 27, 28] considered the problem of computing *diverse* counterfactual explanations on a black box model and proposed several approaches to generate multiple counterfactuals by penalizing their pairwise distance in order to promote diversity. Our cost-aware counterfactual formulation can be seen as a generalization of the weighted counterfactual strategy proposed by [13] that applies a per feature penalty in an $_1$ perturbation. It also satisfies the general definition of the cost functions studied by [7] in the context of recourse for linear models.

6 Experiments

We compare the efficacy of the approaches presented in Section 4 to discover proximal counterfactuals for black box models. We evaluate how these approaches perform even on regions where the black box model has no reliable numerical gradient (i.e., flat regions). We then show how our proposed approach to find cost-aware counterfactuals (presented in Section 3) is able to recover diverse counterfactuals that are proximal to the instance. We also compare to the

available implementations of DiCE [15],[2], which provides multiple counterfactuals. We show that our approach compares favorably, and in most cases outperforms DiCE. In Appendix A.4 we show how cost-aware counterfactuals can incorporate existing feature importance methods such as SHAP and LIME to find counterfactuals.

We experiment on three popular binary classification tabular datasets. These are UCI-Adult for income classification, German-Credit [29] for credit score prediction, and LendingClub, [3], for loan charged-off prediction. We provide a detailed description in Appendix A.3. In all cases, we use the softmax output of a FCNN as the black box model; the FCNN used in these experiments has two hidden layers with 64 neurons each, and uses ReLU activations. For each dataset we train the FCNN model to minimize cross entropy loss using the ADAM optimizer with learning rate 0.01, and batch size 64; we use 80% of the dataset samples for training. Numerical and ordinal features were normalized to the $[0, 1]$ interval, categorical features were converted to their one-hot encoding representation. We consider an elastic norm regularization $R(\delta) = ||\delta||_2 + \beta||\delta||_1$, with $\beta = 0.9$. All experiments are reported over 100 random test instances.

6.1 Finding valid counterfactuals

We compare how the counterfactuals obtain for a black box model with the proposed hybrid approach, `gradsearch`, compares against the existing methods that were discussed in Section 4: `numgrad`, `proto`, `kdtree` and `random`. Here, we only have access to an evaluation function that provides the softmax output of a FCNN. Implementation details are provided in Appendix A.3.

Table 1 show the validity (success rate) and counterfactual quality in terms of the norms of the counterfactual perturbation,[4] (L1-Proximity, L2-Proximity and Sparsity). We can observe that, if we only rely on numerical gradients, the success rate (validity) in finding counterfactuals is considerably lower than other approaches. However, in the cases where `numgrad` succeeds, the quality of the found counterfactuals is high (low perturbation norm, counterfactuals are closer to decision boundary). On the other hand, `random`, `kdtree` and `proto` always succeed (although `random` needs to be given enough samples), but produce lower-quality counterfactuals (i.e., they are not minimal in terms of perturbation norm). Our proposed `gradsearch`, always succeeds, and produces samples of comparable quality to `numgrad` as expected.

	2 counterfactuals			5 counterfactuals			10 counterfactuals		
Method	DivCount	CosDissim	1-IOU	DivCount	CosDissim	1-IOU	DivCount	CosDissim	1-IOU
UCI Adult - 8 features									
DiCEgen	0.61±0.2	0.45±0.24	0.58±0.21	0.62±0.2	0.44±0.25	0.58±0.23	0.65±0.2	0.48±0.25	0.6±0.23
DiCEktree	0.62±0.28	0.49±0.3	0.61±0.29	0.61±0.27	0.48±0.29	0.6±0.28	0.62±0.24	0.47±0.26	0.61±0.25
DiCErand	0.87±0.19	0.71±0.33	0.77±0.27	0.86±0.2	0.72±0.34	0.77±0.29	0.88±0.19	0.74±0.32	0.78±0.27
wCF (ours)	**1.0±0.0**	**0.94±0.23**	**0.92±0.21**	**0.97±0.08**	**0.88±0.26**	**0.87±0.24**	**0.94±0.13**	**0.83±0.29**	**0.83±0.27**
German credit - 20 features									
DiCEgen	0.71±0.19	0.5±0.24	0.52±0.19	0.76±0.15	0.55±0.22	0.57±0.15	0.78±0.13	0.58±0.21	0.59±0.13
DiCEktree	0.76±0.13	0.56±0.21	0.63±0.14	0.76±0.13	0.56±0.21	0.62±0.14	0.76±0.13	0.55±0.2	0.62±0.14
DiCErand	0.91±0.16	0.81±0.29	0.88±0.18	0.92±0.16	0.83±0.29	0.88±0.2	0.92±0.16	0.83±0.28	0.88±0.19
wCF (ours)	**1.0±0.0**	**1.0±0.0**	**0.94±0.11**	**0.97±0.07**	**0.98±0.1**	**0.94±0.13**	**0.97±0.07**	**0.95±0.16**	**0.92±0.15**
Lending Club - 43 features									
DiCEgen	0.84±0.15	0.55±0.22	0.23±0.09	0.87±0.12	0.58±0.22	0.25±0.08	0.88±0.1	0.59±0.22	0.25±0.08
DiCErand	0.12±0.06	0.83±0.3	0.04±0.03	0.12±0.06	0.83±0.32	0.04±0.04	0.12±0.08	0.84±0.31	0.04±0.04
wCF (ours)	**0.97±0.09**	**0.93±0.26**	**0.88±0.27**	**0.98±0.07**	**0.86±0.34**	**0.83±0.33**	**0.98±0.07**	**0.85±0.34**	**0.83±0.33**

Table 2: The above table shows the average and standard deviation of the different metrics used to quantify the diversity between the counterfactuals obtained with the different methods. We report diversity count (`DivCount`), cosine dissimilarity (`CosDissim`), and difference over union (1-`iou`), as previously defined in Eq.17. Larger values indicates high diversity on all metrics. In all scenarios, the proposed sequentially-weighted approach, wCF, achieves the best performance. We omit reporting results for DiCEktree on the Lending Club dataset since the available implementation did not converge in a reasonable time.

[2] https://github.com/interpretml/DiCE

[3] https://www.kaggle.com/datasets/janiobachmann/lending-club-first-dataset

[4] $\delta_{cf} = \mathbf{x}_{cf} - \mathbf{x}_0$

6.2 Multiple Weighted Counterfactuals

We compare the wCF approach, described in Eq.5, to the publicly-available model-agnostic variants of DiCE [15] (DiCErand, DiCEktree and DiCEgen) on the task of generating multiple, diverse counterfactual examples. We report the norms of the counterfactual perturbations to measure proximity of the obtained counterfactuals to the original instance. To quantify diversity, we consider pairwise comparisons and report the average of diversity count (DivCount), cosine dissimilarity (CosDissim), and difference over union (1-IOU). These quantities are described in Appendix A.3, larger values implies more diversity. Additional details about implementation are available in Appendix A.3.

Table 2 compares the diversity of the counterfactuals obtained by the DiCE approaches and wCF using the metrics defined in Eq. 17. In general, wCF is able to provide more diverse counterfactuals. Our weight-based method generates counterfactuals that perturb different set of features as captured in the 1-IOU metric. Table 3 in Appendix A.4 shows the proximity of the generated counterfactuals. We observe that wCF and DiCErand provide counterfactuals closer to the original instance. For the Lending Club dataset, which has considerably more features, the DiCErand counterfactual tend to modify multiple features as shown in the $_0$ norm. From Table 2 and Table 3 we can conclude that wCF with the proposed gradsearch approach tend to produce valid counterfactuals that are more diverse and have a better quality. Additional results are available in Appendix A.4, in particular we provide an analysis of how effective are the importance feature weights of LIME and SHAP methods to generate counterfactuals.

7 Conclusion

In this work we discussed the practical challenges of finding counterfactuals for black box models, especially when numerical gradients of the model are unreliable or near 0, a core practical consideration. We review existing approaches, and propose a novel method that combines numerical gradients and exploration within flat regions without requiring access to a counterfactual dataset. We show that our approach always succeeds, and produces better quality counterfactuals than alternatives.

We additionally analyzed the problem of finding cost-aware counterfactuals, where we have a preference over which features should be perturbed. In this setting we propose a cost-aware counterfactual approach that applies a feature-based penalization on the regularization term. We show that this method can be leveraged to obtain multiple, diverse counterfactuals. Our alternative favorably compares against state of the art alternatives such as DiCE.

References

[1] David Gunning. Explainable artificial intelligence (xai). *Defense advanced research projects agency (DARPA), nd Web*, 2(2):1, 2017.

[2] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

[3] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019.

[4] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In *CCF international conference on natural language processing and Chinese computing*, pages 563–574. Springer, 2019.

[5] Kush R Varshney. Trustworthy machine learning. *Chappaqua, NY, USA: Independently Published*, 2022.

[6] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[7] Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 10–19, 2019.

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

[9] Lloyd S Shapley. A value for n-person games. In *Contributions to the Theory of Games (AM-28), Volume II*, pages 307–318. Princeton University Press, 2016.

[10] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[11] Victor Guyomard, Françoise Fessant, Thomas Guyet, Tassadit Bouadi, and Alexandre Termier. Vcnet: A self-explaining model for realistic counterfactual generation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 437–453. Springer, 2022.

[12] Hangzhi Guo, Thanh H Nguyen, and Amulya Yadav. Counternet: End-to-end training of prediction aware counterfactual explanations. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 577–589, 2023.

[13] Rory Mc Grath, Luca Costabello, Chan Le Van, Paul Sweeney, Farbod Kamiab, Zhao Shen, and Freddy Lecue. Interpretable credit application predictions with counterfactual explanations. *arXiv preprint arXiv:1811.05245*, 2018.

[14] Ramaravind Kommiya Mothilal, Divyat Mahajan, Chenhao Tan, and Amit Sharma. Towards unifying feature attribution and counterfactual explanations: Different means to the same end. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 652–663, 2021.

[15] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 607–617, 2020.

[16] Yoav Freund and Robert E Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the ninth annual conference on Computational learning theory*, pages 325–332, 1996.

[17] Amit Dhurandhar, Tejaswini Pedapati, Avinash Balakrishnan, Pin-Yu Chen, Karthikeyan Shanmugam, and Ruchir Puri. Model agnostic contrastive explanations for structured data. *arXiv preprint arXiv:1906.00117*, 2019.

[18] Chris Russell. Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 20–28, 2019.

[19] Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-objective counterfactual explanations. In *International Conference on Parallel Problem Solving from Nature*, pages 448–469. Springer, 2020.

[20] Dylan Slack, Anna Hilgard, Himabindu Lakkaraju, and Sameer Singh. Counterfactual explanations can be manipulated. *Advances in Neural Information Processing Systems*, 34:62–75, 2021.

[21] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 650–665. Springer, 2021.

[22] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *Advances in neural information processing systems*, 31, 2018.

[23] Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan E Hines, John P Dickerson, and Chirag Shah. Counterfactual explanations and algorithmic recourses for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2020.

[24] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics*, 26(1):56–65, 2019.

[25] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. Inverse classification for comparison-based interpretability in machine learning. *arXiv preprint arXiv:1712.08443*, 2017.

[26] Sijia Liu, Bhavya Kailkhura, Pin-Yu Chen, Paishun Ting, Shiyu Chang, and Lisa Amini. Zeroth-order stochastic variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

[27] Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics*, pages 895–905. PMLR, 2020.

[28] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. Face: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020.

[29] Dheeru Dua, Casey Graff, et al. Uci machine learning repository. 2017.

A Supplementary Material

A.1 Finding Counterfactuals Extension

Algorithm 2 presents a gradient-based approach to achieve Eq. 3 objective. In each iteration we alternate between a gradient descent step in δ, and an increase in λ if the perturbation does not satisfy the counterfactual constraint, and has made no progress in improving our penalty loss, $C_{f,\mathbf{x}_0}(\delta_{t+1}) \geq C_{f,\mathbf{x}_0}(\delta_t)$. Line 5 requires the gradient estimation of f whenever $C_{f,\mathbf{x}_0}(\delta)$ is active; $_{\delta+\mathbf{x}_0 \ X}$ denotes the projection into the feasible input space, [5].

Algorithm 2 Counterfactual Optimization

Input: f model, R regularization, $\mathbf{x}_0$ instance
Parameter:λ initial penalty, T iterations, sc scale factor
Output:$\mathbf{x}_{cf}$ counterfactual

1: $\hat{y}_0 = \arg\max_{i \ Y} f_i(\mathbf{x}_0)$
2: $\delta_0 = \mathbf{0}^d$, $\delta_{cf} = \mathbf{0}^d$,
3: **for** $t = 1, ..., T$ **do**
4: $\quad i_t = \arg\max_{i \ Y \setminus \hat{y}_0} f_i(\mathbf{x}_0 + \delta_t)$
5: $\quad \mathbf{v}_t = \nabla C_{f,\mathbf{x}_0}(\delta_t) = \mathbf{1}_{[C_{f,\mathbf{x}_0}(\delta_t)>0]} \left[\nabla f_{\hat{y}_0}(\mathbf{x}_0 + \delta_t) - \nabla f_{i_t}(\mathbf{x}_0 + \delta_t) \right]$,
6: $\quad \delta_{t+1} = {}_{\delta+\mathbf{x}_0 \ X} \left[\delta_t - \text{lr} \left[\nabla R(\delta_t) - \lambda \mathbf{v}_t \right] \right]$
7: $\quad$ **if** $C_{f,\mathbf{x}_0}(\delta_{t+1}) = 0$ and $R(\delta_{t+1}) < R(\delta_{cf})$ **then**
8: $\quad\quad \delta_{cf} = \delta_{t+1}$
9: $\quad$ **end if**
10: $\quad$ **if** $C_{f,\mathbf{x}_0}(\delta_{t+1}) \geq C_{f,\mathbf{x}_0}(\delta_t)$ and $C_{f,\mathbf{x}_0}(\delta_t) > 0$ **then**
11: $\quad\quad \lambda = \lambda \times \text{sc}$
12: $\quad$ **end if**
13: **end for**
14: **return** $\delta_{cf} + \mathbf{x}_0$

For black box classifiers, Algorithm 2 requires the numerical estimation of $\nabla f_i(\mathbf{x})$ w.r.t. a particular output label $i \ Y$. One option is to do a numerical estimation (numgrad) as described next.

Numerical gradients

We consider the approach taken in [17, 26] where the $\nabla f_i(\mathbf{x})$ is estimated with the empirical approximation of the following expectation

$$\hat{\nabla} f_i(\mathbf{x}) = \mathrm{E}_{\mathbf{u} \ U(\mathbf{S}^{d-1})} \left[d \frac{f_i(\mathbf{x}_0 + \delta_t + \mu\mathbf{u}) - f_i(\mathbf{x}_0 + \delta_t)}{\mu} \mathbf{u} \right]. \quad (6)$$

Note that the directions $\mathbf{u}$ are sampled from a uniform distribution over the unit sphere, $U(\mathbf{S}^{d-1})$. We use numgrad to denote a black box counterfactual explanation approach that uses an empirical estimation of Eq. 6 on Algorithm 2.

Prototype guidance

In situations where no gradient information is available (i.e., flat regions) one can rely on a dataset containing counterfactual examples to provide some guidance by augmenting the loss in Eq.3 with a prototype loss. Given access to $D_{cf} = \{\mathbf{x}_s\}_{s=1}^n$ with $\arg\max_{i \ Y} f_i(\mathbf{x}_s) = \arg\max_{j \ Y} f_j(\mathbf{x}_0)$ one can generate a counterfactual prototype [21], $\mathbf{x}_{proto} = q(D_{cf})$ (e.g.,$\mathbf{x}_{proto} = \arg\min_{\mathbf{x} \ D_{cf}} R(\mathbf{x} - \mathbf{x}_0)$), and augment the objective in Eq. 3 as follows

$$\min_{\delta:\delta+\mathbf{x}_0 \ X} \max_{\lambda} R(\delta) + \lambda C_{f,\mathbf{x}_0}(\delta) + \beta d(\mathbf{x}_{proto}, \mathbf{x}_0 + \delta). \quad (7)$$

Here $d(\mathbf{x}_{proto}, \mathbf{x}_0 + \delta)$ is the distance between the prototype and the objective. If the initial perturbation is set to be $\delta_0 = \mathbf{x}_{proto} - \mathbf{x}_0$, then the method starts from a feasible solution, and

[5] $_{\delta+\mathbf{x}_0 \ X}[\delta] := \arg\min_{\delta' :\delta'+\mathbf{x}_0 \ X} d_\delta(\delta, \delta')$ for some distance function d_δ

$R(\delta)$ is the term that pushes for the counterfactual to be closer to the original instance. This approach is able to provide valid counterfactuals even without gradients for f. However, the final solutions may not have the best quality in terms of R. We denote this approach as `proto`.

Random Search

Instead of optimizing a counterfactual objective, one can naively explore the input space in search of counterfactuals by random sampling (`random`). Given a distribution whose support is X, denoted as P_X, one can sample independent instances $\mathbf{x}_s$, and select the sample with the smallest perturbation amongst valid counterfactuals. This is summarized as follows, [6]

$$\mathbf{x}_{cf} = \arg \min_{\mathbf{x} \ \{\mathbf{x}_s \ \ P_X : f(\mathbf{x}_s) = f(\mathbf{x}_0)\}_{s=1}^{n}} R(\mathbf{x} - \mathbf{x}_0) \tag{8}$$

In our experiments, we consider P_X to be the uniform distribution on the feasible range of values. This corresponds to the least informative prior over the dataset.

Dataset Guidance

In cases where a dataset containing counterfactual samples is available, $D_{cf} = \{\mathbf{x}_s\}_{s=1}^{n}$ with $f(\mathbf{x}_s) = f(\mathbf{x}_0)$ for all $s = 1, ..., n$. One can simply output the instance in the counterfactuals dataset that is closer in terms of R,

$$\mathbf{x}_{cf} = \arg \min_{\mathbf{x} \ D_{cf}} R(\mathbf{x} - \mathbf{x}_0). \tag{9}$$

This is equivalent to the random search approach, where the counterfactual dataset is provided, rather than sampled. An implementation of the above approach using `kdtree`, as well as the `random` method (see Eq 8) is available in the library provided by [15].

A.2 Additional Analysis on cost-aware Counterfactuals

Proposition A.1 shows that, for Eq. 4, features with larger weights are proportionally penalized for having larger perturbations. Proposition A.2 shows that increasing the weight of a single feature can only maintain or decrease the module of the perturbation of that feature. Proposition A.3 shows as an example how in the binary linear classification setting, and if we consider the l2 regularization, the perturbation for each feature is inversely proportional to the weight/cost of the feature.

Proposition A.1. *Consider $R : \mathrm{R}^d \to \mathrm{R}_+$ a convex function with a minimum in $\mathbf{0}^{\mathbf{d}}$ that, for any δ R^d and any coordinate pair i, j $\{1, ..., d\}$ such that $|\delta_i| \geq |\delta_j|$ satisfies $|\frac{\partial R(\delta)}{\partial \delta_i}| \geq |\frac{\partial R(\delta)}{\partial \delta_j}|$, with equality only if $|\delta_i| = |\delta_j|$. Then, given a weight vector $\mathbf{w} : w_i > w_j > 0$ and δ R^d such that $\delta_i = \delta_j$ we have that $|\frac{\partial R(\mathbf{w} \ \delta)}{\partial \delta_i}| > |\frac{\partial R(\mathbf{w} \ \delta)}{\partial \delta_j}|$ if $\delta_j = 0$.*

Proof. $|\frac{\partial R(\mathbf{w} \ \delta)}{\partial \delta_i}| = |w_i \frac{\partial R(\mathbf{w} \ \delta)}{\partial w_i \delta_i}| > |w_j \frac{\partial R(\mathbf{w} \ \delta)}{\partial w_j \delta_j}| = |\frac{\partial R(\mathbf{w} \ \delta)}{\partial \delta_j}|$ since $|w_i \delta_i| = |w_j \delta_i|$ and $w_i > w_j > 0$. □

Proposition A.2. *Consider $R : \mathrm{R}^d \to \mathrm{R}_+$ a convex function with a minimum in $\mathbf{0}^{\mathbf{d}}$ that can be expressed as $\mathrm{R}(\delta) = \sum_{k=1}^{d} r(\delta_k)$ where $r : \mathrm{R} \to \mathrm{R}_+$ is a real function that satisfies $r(\delta_1) > r(\delta_2)$ if $|\delta_1| > |\delta_2|$. Given $\mathbf{w}, \mathbf{w}$ R_+^d such that $w_i > w_i$ and $w_j = w_j$ for all $j = i$, the solutions $\delta^w = \arg\min_{\delta} wCF(\delta; \mathbf{w})$ and $\delta^w = \arg\min_{\delta} wCF(\delta; \mathbf{w})$ (solutions to Eq. 4) satisfy that $|\delta_i^w| \leq |\delta_i^w|$.*

Proof. Given the above assumptions we can write the following

$$\begin{aligned} R(\mathbf{w} \quad \delta^w) &= R(\mathbf{w} \quad \delta^w) - r(w_i \times \delta_i^w) + r(w_i \times \delta_i^w) \\ &\geq R(\mathbf{w} \quad \delta^w) - r(w_i \times \delta_i^w) + r(w_i \times \delta_i^w) \\ &\geq R(\mathbf{w} \quad \delta^w) - r(w_i \times \delta_i^w) + r(w_i \times \delta_i^w). \end{aligned} \tag{10}$$

[6] Here we abuse notation and $f(\mathbf{x})$ indicates the decision instead of the probability score

If $|\delta_i^w| > |\delta_i^w|$ then we would have that $r(w_i \times \delta_i^w) > r(w_i \times \delta_i^w)$ leading to the contradiction of $R(\mathbf{w} \quad \delta^w) > R(\mathbf{w} \quad \delta^w)$ where δ^w could not be the minimizer for $wCF(\delta, \mathbf{w})$.

□

Proposition A.3. *We consider Eq.4 in a binary linear classification setting where* $f(\mathbf{x}) = \frac{1}{1+e^{-h(\mathbf{x})}}$ *and* $h(\mathbf{x}) = \mathbf{v}^T\mathbf{x} + b$. *Without loss of generality we consider* $f(\mathbf{x}_0) > 0.5$, *then Eq.4 can be expressed as*

$$\min_{\delta} \max_{\lambda \geq 0} L(\delta, \lambda)$$
$$L(\delta, \lambda) = R(\mathbf{w} \quad \delta) + \lambda \left(\frac{1}{1+e^{-h(\mathbf{x}_0+\delta)}} - \frac{1}{1+e^{h(\mathbf{x}_0+\delta)}} \right) \tag{11}$$

and the solution with $R(\mathbf{w} \quad \delta) = ||\mathbf{w} \quad \delta||_2^2 + \beta||\mathbf{w} \quad \delta||_1$ *is*

$$\delta \quad = \{ \frac{-(\mathbf{v}^T\mathbf{x}_0 + b)}{||\frac{\mathbf{v}}{2\mathbf{w}}||_2^2} \frac{v_i}{4w_i^2} \}_{i=1}^d$$

Proof. Lets consider the KKT conditions.

$$\begin{aligned} &\frac{\partial L}{\partial \lambda} = \frac{1}{1+e^{-h(\mathbf{x}_0+\delta \quad)}} - \frac{1}{1+e^{h(\mathbf{x}_0+\delta \quad)}} = 0 \\ &\text{then } h(\mathbf{x}_0 + \delta \quad) = 0 \rightarrow \mathbf{v}^T\delta \quad = -(\mathbf{v}^T\mathbf{x}_0 + b). \end{aligned} \tag{12}$$

With the above result we can compute

$$\begin{aligned} &\frac{\partial L}{\partial \delta_i} = 2w_i^2\delta_i + \lambda\frac{1}{2}v_i = 0 \rightarrow \delta_i = -\frac{\lambda v_i}{4w_i^2}, \\ &\mathbf{v}^T\delta \quad = -\sum_{i=1}^d \frac{\lambda v_i^2}{4w_i^2} = -(\mathbf{v}^T\mathbf{x}_0 + b), \\ &\lambda = \frac{(\mathbf{v}^T\mathbf{x}_0 + b)}{\sum_{i=1}^d \frac{v_i^2}{4w_i^2}} \geq 0. \end{aligned} \tag{13}$$

Then $\delta_i = \frac{(\mathbf{v}^T\mathbf{x}_0 + b)}{||\frac{\mathbf{v}}{2\mathbf{w}}||_2^2} \frac{v_i}{4w_i^2}$ for $i = 1, ..., d$

□

A.3 Experimental Details

UCI-Adult This is a binary classification tabular dataset, available in [29]. It contains census information for more than 26k individuals. The goal is to predict whether or not the annual income of a person is above 50k. We followed the same pre-processing described in [15], where 8 features (2 numerical and 6 categorical) are considered for the income prediction. The trained FCNN classifier achieved and accuracy of 85.4% and 83.1% on the train and test partitions respectively.

German-Credit This is a tabular dataset, available in [29], that contains personal information from a set of 1k individuals that took on a bank loan. The goal is to predict, for each individual, a binary variable corresponding to the credit score (good or bad). We utilized all 20 features, 12 numerical or ordinal, and 8 strictly categorical. The trained FCNN classifier achieved and accuracy of 100% and 71.5% on the train and test partitions respectively.

LendingClub This tabular dataset,[7] contains pertinent information from individuals that requested a loan in the years between 2007 and 2015. The goal is to predict a binary variable indicating whether the loan was fully paid or charged off. We utilized 43 features, 38 corresponding to numerical or ordinal variables, and 5 to categorical ones. The dataset has nearly 40k samples after data cleaning. The trained FCNN classifier achieved an accuracy of 99.5% and 99.6% on the train and test partitions respectively.

[7] https://www.kaggle.com/datasets/janiobachmann/lending-club-first-dataset

Counterfactual Methods

numgrad: Uses Algorithm 2 with $\lambda = 0.01$, $T = 2000$, sc = 1.25, and lr = 0.1 with a polinomial decay,[8] of power 0.5 and end learning rate of 1e-7. The numerical gradient of $f_i(\mathbf{x})$ in step 5 from Algorithm 2 is computed with the following empirical approximation of Eq. 6

$$\begin{aligned} \hat{} f_i(\mathbf{x}) &= \frac{d}{n\mu} \sum_{j=1}^{n} f_i(\mathbf{x} + \mu \mathbf{u}_j) - f_i(\mathbf{x}) \mathbf{u}_j, \quad i \quad Y \\ \mathbf{u}_j &= \frac{\hat{\mathbf{u}}_j}{||\hat{\mathbf{u}}_j||_2}, \quad \hat{\mathbf{u}}_j \quad N(\mathbf{0}, \mathbf{I}), j = 1, ..., n. \end{aligned} \tag{14}$$

We used $\mu = 0.005$ and $n = 50$ samples. d are the dimensions of the input features and is data dependent. The regularization loss we are using is $R(\delta) = ||\delta||_2^2 + 0.9||\delta||_1$, and we incorporate the projected FISTA from [22]. Therefore, we consider the slack variable $\mathbf{z}_t$ such that $\mathbf{z}_0 = \delta_0$ and line 6 in Algorithm 2 becomes

$$\begin{aligned} \delta_{t+1} &= \min_{\delta + \mathbf{x}_0 \ X} S_\beta \left(\mathbf{z}_t - \text{lr} \, 2\mathbf{z}_t - \lambda \mathbf{v}_t \right), \\ \mathbf{z}_{t+1} &= \delta_{t+1} + \frac{t}{t+3}(\delta_{t+1} - \delta_t). \end{aligned} \tag{15}$$

Where $S_\beta : \mathrm{R}^d \rightarrow \mathrm{R}^d$ is an element-wise shrinkage-thresholding function

$$S_\beta(\delta)_i = (\delta_i - \beta)\mathbf{1}_{[\delta_i > \beta]} + (\delta_i + \beta)\mathbf{1}_{[\delta_i < -\beta]} + 0\mathbf{1}_{[|\delta_i| < \beta]} \tag{16}$$

where $i = 1, ..., d$.

proto: Same as numgrad except for the following points. The loss is augmented to include a proto distance term as shown in Eq. 7. We set $\mathbf{x}_{proto}$ to be the closest sample in the training dataset to $\mathbf{x}_0$ in terms of $R(\mathbf{x}_{proto} - \mathbf{x}_0) = ||\mathbf{x}_{proto} - \mathbf{x}_0||_2^2 + 0.9||\mathbf{x}_{proto} - \mathbf{x}_0||_1$. Moreover, the prototype distance loss is $d(\mathbf{x}_{proto}, \mathbf{x}) = ||\mathbf{x}_{proto} - \mathbf{x}||_2^2$, and β was initialized in 0.1 and increased under the same conditions as λ in line 11 from Algorithm 2. We start with a feasible solution by setting $\delta_0 = \mathbf{x}_{proto} - \mathbf{x}_0 = \delta_{cf}$ in step 2 from Algorithm 2 .

gradsearch: Same as numgrad but we use Algorithm 1 to get the direction update of line 5 in Algorithm 2. In Algorithm 1 initial radius is $\mu = 0.005$, $n = 50$ and $c = 2$.

kdtree and random: We use the implementation available in the DiCE library [15] ,[9].

DiCErand, DiCEktree and DiCEgen : DiCE provides three model agnostic implementations to find counterfactuals. One is random sampling of the input space (DiCErand), to find counterfactuals and then choose those that satisfy a proximity condition. The second approach is a k-NN implementation (DiCEktree) that requires access to a dataset with valid counterfactuals, and outputs those that are closer to a given instance. Finally, they provide a genetic algorithm approach (DiCEgen) whose goal is to find a compromise between proximal and diverse counterfactuals. Diversity is encouraged by penalizing pairwise distance between counterfactuals. We used the suggested parameters for all three approaches and increased either the number of samples or iterations in cases where the original setting was insufficient. For DiCEgen, we sweep the coefficient that promotes diversity and report the results with best diversity. We set the maximum number of iterations to 10K and run for diversity weight equal to 5, 10, 50 and 100. However, we did not see a major improvement in terms of diversity, in all cases we reported the one that achieved the best performance. In all three cases we use the implementation available in the DiCE library [15].

wCF: Given a feature weight vector $\mathbf{w}$ (see Eq. 4) we used gradsearch to find the counterfactual. For multiple counterfactuals we tried $\gamma \quad \{10, 50, 100\}$ (see Eq.5) and reported the one with best performance in terms of diversity. Reported results for UCI Adult and German credit use γ=50, and γ=100 for Lending Club.

[8] https://keras.io/api/optimizers/learning_rate_schedules

[9] https://github.com/interpretml/DiCE

Quantify counterfactuals diversity

To quantify diversity, we consider pairwise comparisons and report the average of diversity count (DivCount), cosine dissimilarity (CosDissim), and difference over union (1-IOU). Given k counterfactuals, these metrics are

$$\begin{aligned}
\text{DivCount:} \quad & \frac{1}{C_k^2} \bigsqcup_{\substack{i=1\ldots,k \\ j=i+1,\ldots,k}} \bigsqcup_{l=1}^{d} \frac{||\delta_{cf,i} - \delta_{cf,j}||_0}{(\delta^l_{cf,i}=0) \quad (\delta^l_{cf,j}=0)}, \\
\text{CosDissim:} \quad & 1 - \frac{1}{C_k^2} \bigsqcup_{\substack{i=1\ldots,k \\ j=i+1,\ldots,k}} \frac{<\delta_{cf,i}, \delta_{cf,j}>}{||\delta_{cf,i}||||\delta_{cf,j}||}, \\
\text{1-IOU:} \quad & 1 - \frac{1}{C_k^2} \bigsqcup_{\substack{i=1\ldots,k \\ j=i+1,\ldots,k}} \frac{\bigsqcup_{l=1}^{d} (\delta^l_{cf,i}=0) \quad (\delta^l_{cf,j}=0)}{\bigsqcup_{l=1}^{d} (\delta^l_{cf,i}=0) \quad (\delta^l_{cf,j}=0)}.
\end{aligned} \tag{17}$$

Here C_k^2 represents the total combinations of k choose 2 elements, and and represent the 'and' and 'or' operators.

A.4 Additional Results

	2 counterfactuals			5 counterfactuals			10 counterfactuals		
Method	$\frac{1}{d}\|\|\delta\|\|_1$	$\frac{1}{d}\|\|\delta\|\|_2$	$\frac{1}{d}\|\|\delta\|\|_0$	$\frac{1}{d}\|\|\delta\|\|_1$	$\frac{1}{d}\|\|\delta\|\|_2$	$\frac{1}{d}\|\|\delta\|\|_0$	$\frac{1}{d}\|\|\delta\|\|_1$	$\frac{1}{d}\|\|\delta\|\|_2$	$\frac{1}{d}\|\|\delta\|\|_0$
UCI Adult - 8 features									
DiCEgen	.171±.067	.171±.067	.183±.067	.171±.064	.17±.064	.188±.062	.169±.064	.168±.065	.189±.062
DiCEktree	.147±.065	.147±.065	.154±.071	.16±.072	.16±.072	.167±.077	.17±.072	.169±.072	.178±.077
DiCErand	.086±.04	.081±.042	.102±.032	.091±.043	.086±.046	**.105±.035**	**.092±.043**	**.087±.046**	**.107±.036**
wCF(ours)	**.059±.053**	**.055±.054**	**.078±.047**	**.089±.073**	**.085±.073**	.106±.071	.093±.07	.088±.07	.11±.069
German credit- 20 features									
DiCEgen	.176±.054	.156±.054	.268±.054	.174±.049	.153±.05	.267±.05	.173±.051	.152±.052	.265±.051
DiCEktree	.182±.054	.158±.054	.256±.058	.183±.053	.159±.054	.259±.057	.186±.053	.162±.053	.264±.056
DiCErand	.055±.026	.051±.028	.065±.022	.057±.026	.053±.028	.066±.022	.057±.027	.053±.029	**.068±.023**
wCF(ours)	**.028±.029**	**.02±.028**	**.057±.036**	**.042±.042**	**.036±.041**	**.065±.047**	**.048±.04**	**.042±.038**	.071±.046
Lending Club- 43 features									
DiCEgen	.122±.039	.077±.035	.485±.038	.124±.038	.077±.034	.48±.035	.126±.039	.079±.035	.481±.033
DiCErand	.018±.02	.012±.015	.441±.023	.017±.019	.011±.015	.441±.024	**.018±.023**	**.012±.017**	.442±.025
wCF (ours)	**.005±.02**	**.003±.014**	**.029±.045**	**.014±.043**	**.009±.033**	**.047±.081**	.019±.05	.013±.039	**.054±.089**

Table 3: Average and standard deviation of the three different norms used to quantify the magnitude of the counterfactuals' perturbation for the different methods. Here, a smaller value indicates that the counterfactual is closer to the instance. We do not report DiCEktree for Lending Club since the available implementation did not converge in a reasonable time.

Analysis of multi counterfactual updates

We analyze how the multi counterfactual updates in Eq.5 behave as we increase the number of counterfactual examples. Figure 1 shows how the cosine similarity between consecutive weight updates, $\text{cossim}(\mathbf{w}_{cf}, \mathbf{w}_{cf-1})$, evolves across iterations for different γ parameters. We observe that, as the number of counterfactuals increases, the consecutive weights become more similar. This is evident on the UCI Adult and German datasets at 10 counterfactuals, and is an expected behavior, since increasing the number of diverse counterfactuals will eventually perturb most of the features that are able to flip the model's decision. We also show how the similarity between a perturbation and its corresponding weight evolves through subsequent iterations by measuring the ration between cosine similarities $\frac{\text{cossim}(\mathbf{w}_{cf}, \delta_{cf})}{\text{cossim}(\mathbf{w}_{cf}, \delta_{cf-1})}$. This quantity is expected to be lower than 1, as shown in Figure 1, since $\mathbf{w}_{cf}$ takes higher values on features that should not be perturbed considering δ_{cf-1}.

Analysis of importance features explainers

The proposed wCF approach can be used to evaluate the effectiveness at generating counterfactuals of importance feature weights provided by other explanation methods such as SHAP or LIME. To evaluate this, we assume that we are given a vector of non-negative feature importance weights $I_F \quad \mathbb{R}_+^d$, and define a weight vector $\mathbf{w}_F$ that assigns a lower cost on features with

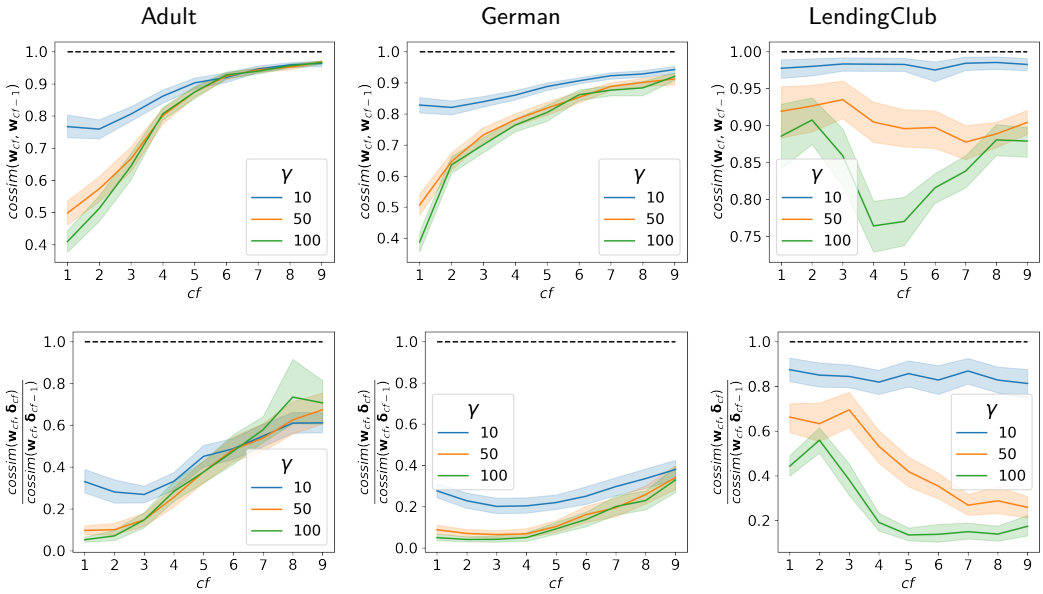

Figure 1: First row shows cosine similarity between consecutive weight updates for the proposed wCF approach (Eq.5). Second row shows the ratio of cosine similarity between the current weight and the recovered perturbation and cosine similarity between the current weight and the previous perturbation. For cosine similarity, lower values are preferred, while for the ratio metric, higher values are better.

higher importance. To build this weight vector, we first negate I_F and define the cost-aware counterfactual weights $\mathbf{w}_F$ as follows

$$\begin{aligned} \mathbf{w}_F &= \frac{nI_F}{||nI_F||_1}\gamma + 1, \\ \text{with } nI_F &= -I_F + \max(I_F). \end{aligned} \tag{18}$$

We utilize this weight vector $\mathbf{w}_F$ in wCF to generate counterfactuals that directly relate to the I_F produced by other explainers.

We evaluate how the quality/cost of the counterfactuals change if we consider the importance feature vector (I_F) provided by local and global SHAP (LSHAP-wCF and GSHAP-wCF), as well as LIME (LLIME-wCF and GLIME-wCF) in our cost-aware counterfactual formulation. We compare the cost-aware counterfactuals obtained when using I_F to generate weights against a uniform weighting $\mathbf{w} = \mathbf{1}^d$ (uCF) where no feature preference is provided. The quantitative comparison is done using the metrics described below.

cossim$(I_F, |\delta|)$**:** Cosine Similarity between the importance feature vector I_F and the absolute feature-wise perturbation associated to the obtained counterfactual $|\delta|$.

$$\text{cossim}(I_F, |\delta|) = \frac{< I_F, |\delta| >}{||I_F||_2||\delta||_2}. \tag{19}$$

The higher $\text{cossim}(I_F, |\delta|)$ is the more aligned the perturbation of the recovered counterfactual is with the importance features provided. We expect $\text{cossim}(I_F, |\delta|)$ to be larger when the I_F is provided as a weight in our cost-aware formulation (LSHAP-wCF, GSHAP-wCF,LLIME-wCF, GLIME-wCF) in comparison to providing uniform preference (uniform cost weights, uCF). If the counterfactual perturbation obtained with uniform preference (uCF) aligns with I_F it means that the low cost counterfactuals are naturally found in a direction closer to the I_F provided by the importance feature explainer.

rank$_{I_F}$**:** Smallest index of the descending sorted ordering of I_F that covers all perturbed features from δ. Given π_i with $i = 1, ..., F$ a descending sorting of the importance feature vector I_F,

$I_{F,\pi_i} \geq I_{F,\pi_{i+1}}$, rank_{I_F} is defined as follows

$$k = \min_{i=1,\dots,F} i \quad s.t., |\delta_{\pi_j}| = 0, \quad \pi_j > \pi_i$$
$$\text{rank}_{I_F} = 100 \quad k/F \tag{20}$$

This evaluation metric indicates that the set of perturbed features in the obtained counterfactual are in the top-rank_{I_F} of the I_F vector. If this metric is small, the most important features in I_F tend to be sufficient to produce a valid counterfactual.

cossim(I_F, I_{cf})**:** Cosine similarity between a global importance feature vector based on the counterfactual perturbations, I_{cf} and the corresponding I_F obtained with SHAP or LIME. We consider I_{cf} as a global importance feature vector based on the counterfactual perturbations on N random instances on the dataset, $I_{cf} = \frac{1}{N} \sum_{n=1}^{N} |\delta_n|$.

$$\text{cossim}(I_F, I_{cf}) = \frac{< I_F, I_{cf} >}{||I_F||_2 ||I_{cf}||_2}. \tag{21}$$

The higher $\text{cossim}(I_F, I_{cf})$ is, the more aligned the features perturbed by the counterfactual explanation are with the importance features vector obtained with the other explainers.

Proximity and Sparsity: $\frac{1}{d}||\delta||_1$ is the L1-Proximity, $\frac{1}{d}||\delta||_2$ the L2-Proximity and $\frac{1}{d}||\delta||_0$ the Sparsity of the obtained counterfactuals. As presented before, δ is the feature-wise difference vector between the counterfactual and the original instance $\delta = x_{cf} - X_0$.

Table 4 show a comparison of the global and local SHAP- and LIME-derived counterfactuals, as well as the uniform-weighted counterfactual (uCF). We observe that, as a general rule, the uCF counterfactuals, tend to align better with SHAP I_F than with LIME, in terms of both rank and cossim. This is more prominent when we compare the cossim between the SHAP I_F, and those derived from our counterfactuals I_{cf}. When we impose a weight that is consistent with I_F (wCF) the similarity between the obtained counterfactuals and I_F increases, and the perturbation norm is relatively low. Still, directions provided by SHAP tend to produce counterfactuals that are better aligned with their I_F than LIME. However, note that LIME and SHAP importance features do not perfectly align with the generated counterfactuals, even when we promote their correlation through the cost-aware formulation. This is consistent with the analysis done by [14], where they mask out features based on the ranking of the importance features provided by SHAP and LIME.

method	$\text{cossim}_{I_F,\lvert\delta\rvert}$	rank_{I_F} (%)	$\frac{1}{d}\lvert\lvert\delta\rvert\rvert_1$	$\frac{1}{d}\lvert\lvert\delta\rvert\rvert_2$	$\frac{1}{d}\lvert\lvert\delta\rvert\rvert_0$	$\text{cossim}_{I_F,I_{cf}}$
UCI Adult - 8 features						
LSHAP - wCF	**.49±.19**	**42.6±32.3**	.076±.042	.057±.047	.051±.048	**.886**
LSHAP - uCF	.34±.21	47.1±30.9	**.067±.034**	**.041±.039**	**.035±.04**	.777
GSHAP- wCF	**.42±.11**	**31.0±25.4**	.077±.052	.05±.055	.042±.054	**.841**
GSHAP - uCF	.33±.11	42.6±30.3	**.067±.034**	**.041±.039**	**.035±.04**	.771
LLIME - wCF	**.38±.14**	**35.9±25.0**	.078±.05	.054±.052	.047±.052	**.839**
LLIME - uCF	.27±.14	47.9±29.8	**.067±.034**	**.041±.039**	**.035±.04**	.754
GLIME - wCF	**.36±.1**	**31.7±14.1**	.073±.045	.05±.048	.043±.047	**.806**
GLIME - uCF	.31±.09	36.9±19.7	**.067±.034**	**.041±.039**	**.035±.04**	.755
German credit- 20 features						
LSHAP - wCF	**.51±.17**	**18.8±23.3**	**.042±.022**	.024±.019	.018±.019	**.849**
LSHAP - uCF	.36±.22	34.2±27.2	.053±.031	**.02±.018**	**.012±.017**	.789
GSHAP - wCF	**.5±.09**	**10.8±12.8**	**.046±.023**	.02±.02	.013±.018	.711
GSHAP - uCF	.36±.15	29.0±22.9	.053±.031	**.02±.018**	**.012±.017**	**.786**
LLIME - wCF	**.28±.14**	**33.1±24.3**	**.043±.02**	.024±.019	.018±.019	**.544**
LLIME - uCF	.12±.14	57.1±26.7	.053±.031	**.02±.018**	**.012±.017**	.352
GLIME - wCF	**.27±.13**	**32.5±19.3**	**.046±.025**	.024±.021	.017±.021	**.521**
GLIME - uCF	.13±.11	47.4±18.1	.053±.031	**.02±.018**	**.012±.017**	.352
Lending Club - 43 features						
LSHAP - wCF	**.19±.26**	**16.0±11.1**	**.02±.011**	**.002±.005**	**.001±.003**	**.808**
LSHAP - uCF	.13±.19	22.5±19.6	.027±.033	.003±.01	.001±.005	.83
GSHAP - wCF	**.23±.2**	**12.9±3.1**	**.019±.009**	**.002±.004**	**.001±.003**	.689
GSHAP - uCF	.18±.15	19.2±19.0	.027±.033	.003±.01	.001±.005	**.837**
LLIME - wCF	**.55±.11**	**3.2±5.4**	**.022±.015**	**.002±.005**	**.001±.003**	.664
LLIME - uCF	.52±.15	10.0±22.5	.027±.033	.003±.01	.001±.005	**.67**
GLIME - wCF	**.6±.08**	**4.0±10.7**	**.022±.014**	**.002±.005**	**.001±.003**	.595
GLIME - uCF	.59±.12	9.3±22.9	.026±.03	.003±.009	.001±.005	**.683**

Table 4: We compare global and local, SHAP- and LIME-derived counterfactuals (denoted with the wCF postfix) against the uniform-weighted counterfactual (uCF). All weighted counterfactuals use a scaling factor $\gamma = 100$, as described in Section A.4.