The Skyline of Counterfactual Explanations for Machine Learning Decision Models

Yongjie Wang
yongjie002@e.ntu.edu.sg
School of Computer Science and Engineering, Nanyang Technological University, Singapore

Qinxu Ding
qding001@e.ntu.edu.sg
Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University, Singapore

Ke Wang
wangk@cs.sfu.ca
School of Computing Science, Simon Fraser University, Canada

Yue Liu
ly228308@alibaba-inc.com
Alibaba Group, China

Xingyu Wu
zhuyang.wxy@alibaba-inc.com
Alibaba Group, China

Jinglong Wang
jinglong.wjl@alibaba-inc.com
Alibaba Group, China

Yong Liu
stephenliu@ntu.edu.sg
Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University, Singapore

Chunyan Miao*
ASCYMiao@ntu.edu.sg
School of Computer Science and Engineering, Nanyang Technological University, Singapore

ABSTRACT

Counterfactual explanations are minimum changes of a given input to alter the original prediction by a machine learning model, usually from an undesirable prediction to a desirable one. Previous works frame this problem as a constrained cost minimization, where the cost is defined as L_1/L_2 distance (or variants) over multiple features to measure the change. In real-life applications, features of different types are hardly comparable and it is difficult to measure the changes of heterogeneous features by a single cost function. Moreover, existing approaches do not support interactive exploration of counterfactual explanations. To address above issues, we propose the *skyline counterfactual explanations* that define the skyline of counterfactual explanations as all *non-dominated* changes. We solve this problem as multi-objective optimization over actionable features. This approach does not require any cost function over heterogeneous features. With the skyline, the user can interactively and incrementally refine their goals on the features and magnitudes to be changed, especially when lacking prior knowledge to express their needs precisely. Intensive experiment results on three real-life datasets demonstrate that the skyline method provides a friendly way for finding interesting counterfactual explanations, and achieves superior results compared to the state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Philosophical/theoretical foundations of artificial intelligence**; **Machine learning**.

KEYWORDS

Counterfactual explanations; Multi-objective optimization; Skyline; Interactive query

*corresponding author

ACM Reference Format:
Yongjie Wang, Qinxu Ding, Ke Wang, Yue Liu, Xingyu Wu, Jinglong Wang, Yong Liu, and Chunyan Miao. 2021. The Skyline of Counterfactual Explanations for Machine Learning Decision Models. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482397

1 INTRODUCTION

Machine learning algorithms with millions of parameters are increasingly deployed in real-life applications (e.g., finance, autopilot, security, medical) for automatic decision-making. Despite the impressive performance, the large volume of trainable parameters makes it difficult to understand how a prediction is made by a machine learning model, which is essential for high-stake applications. Various machine learning explanation methods [2, 14, 26, 33, 36] have been proposed to explain the opaque behaviors of machine learning models. Counterfactual explanation [39] is a method for answering the question "what minimum changes are needed for an input instance to flip its bad prediction outcome by the model (e.g., a high risk of disease), into a good one (e.g., a low risk of disease)." Therefore, it has wide applications [18] in healthcare (altering an unhealthy situation to a healthy one), finance (improving loan approval rate), education (improving school work), marketing (customer retention or improving sales), etc .

In particular, given a data point $\boldsymbol{x}$ and a machine learning decision model f learned from a dataset, [39] finds an improved example $\boldsymbol{c}$ that is close to $\boldsymbol{x}$ in the feature space and has a prediction $f(\boldsymbol{c})$ close to a desired target y,

$$arg\min_{\boldsymbol{c}} \max_{\lambda \leq \Lambda} \lambda \ell(f(\boldsymbol{c}), y) + d(\boldsymbol{x}, \boldsymbol{c}) \tag{1}$$

where ℓ is the loss function between the desired target y and the current prediction $f(\boldsymbol{c})$, d is the distance/cost function measuring the change between the input $\boldsymbol{x}$ and $\boldsymbol{c}$, Λ is a hyperparameter and maximization over λ is achieved via searching $\boldsymbol{c}$ iteratively and increasing λ until a sufficient small loss is found.

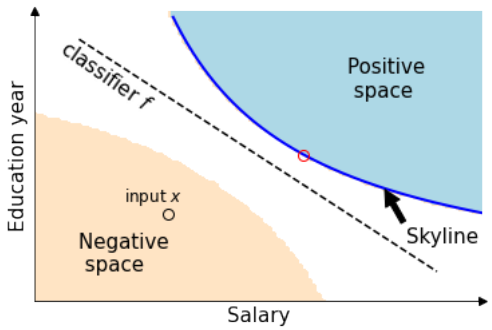

Figure 1: The differences between previous works and the proposed method. Using 2 features "Education year" and "Salary" as an example, positive and negative space are separated by the classifier f. With a given cost function, existing methods often return a single explanation as the red circle. Our method returns the skyline of counterfactuals shown as the blue curve. The skyline of counterfactuals forms the database and user can query the database interactively.

1.1 Limitations of Existing Works

Following [39], much research goes into adding various constraints (e.g., diversity [31, 35], reliability [15, 32], actionability [37], sparsity [11], causality [15, 19, 20]), and introducing different searching strategies (e.g., gradient descent [15, 39], FISTA [11, 38], integer program [37], mixed integer program [35]). However, almost all methods [11, 15–17, 20, 23, 24, 31, 38, 39] require a determinate cost function d to measure the change of $\boldsymbol{c}$ from $\boldsymbol{x}$ in the feature space. For example, [23, 39] introduce the L_2 distance on features scaled to the same range; [11, 15, 31] adopt L_1 or L_2 distances weighted by the inverse median absolute deviation (MAD) on heterogeneous features; [16] uses the Mahalanobis distance; [17, 38] combine the mixture of above distance functions; [8, 31] allow users to specify the relative feature weights and define a weighted distance. Instead of minimizing the distance on the input space, [32] proposes a scalar objective on the latent space via a variational autoencoder.

However, existing works suffer from two major limitations:

Difficulty of defining a cost function d over heterogeneous features. It is hard to compare changes of different features and define the cost of change by a single cost function d. For example, it is unclear how to trade-off between 1 year increase in "Education year" and 2 hour increase in "Work hour". Similarly, it is hard to compare changes between a numerical feature and a categorical feature. In addition, the cost of change often depends on the current feature values of $\boldsymbol{x}$ [18]. For example, improving the exam score from 50 to 60 is easier than from 90 to 100.

Lack of support for specifying constraints. It is hard for the user to specify the constraints precisely [39], especially when lacking domain knowledge and alternatives available. For example, after knowing either 1 year increase in "Education year" or 2 hours increase in "Work hour" can flip the bad prediction, the user may change the initial constraint that no work hour is increased and prefer the latter. This solution cannot be found if the user does not know that it is available.

1.2 Our Approach

We propose a *skyline counterfactual explanation* framework to mitigate the above issues. First, we formulate this problem as finding counterfactual explanations with minimum changes under the multi-objective optimization (MOO) [12, 30]. Instead of finding a single or a few counterfactual explanations determined by the minimum cost, our approach finds a set of all non-dominated counterfactual explanations, called the skyline [3] or Pareto front [30], where a counterfactual explanation $\boldsymbol{c}'$ is dominated by another $\boldsymbol{c}$ if the change of $\boldsymbol{c}'$ is no smaller than that of $\boldsymbol{c}$ on every feature and is larger on at least one feature. Figure 1 shows the skyline of counterfactual explanations. Importantly, the skyline computation does not use any cost function d that must trade-off between different features. For example, consider changes in (Education year, Salary): (+2, 0) is dominated by (+1, 0), whereas (+1, 0) and (0, +5000) are non-dominated.

The skyline also provides a solution to the lack of prior knowledge in specifying user's constraints. Initially, the only constraints on the skyline are the basic requirements for non-dominated and valid counterfactuals, which leave the maximum set of alternative counterfactual explanations for further exploration. With the skyline being available, the user can refine his criteria on counterfactual explanations by "querying" the skyline interactively and incrementally, where the results of previous queries provide the context and prior knowledge for formulating the next query. With a few queries and certain ranking criteria, the user quickly identifies several desired counterfactual explanations. To the best of our knowledge, this is the first work to use non-dominated counterfactual explanations to deal with the difficulty of specifying the cost function and constraints.

1.3 Contributions

The main contributions of our work are four folds:

- Section 2: We review related work and discuss the differences of our work.
- Section 3: We formulate the counterfactual explanation problem as finding the skyline of valid counterfactual explanations in a given feature space. This skyline serves as a counterfactual database for exploratory discovery of desired counterfactual explanations. We propose a query template for this discovery process.
- Section 4: We propose a sample-directed method, named skyline counterfactual algorithm (SkylineCF), to search for the skyline of counterfactual explanations. The efficiency of this method is provided by considering the search space defined by straight lines from the input $\boldsymbol{x}$ to valid samples with the target prediction y.
- Section 5: We evaluate the skyline approach using real-life datasets by common evaluation metrics, and we also showcase the power of querying the skyline to discover interesting counterfactual explanations.

Notations. Frequently used notations are given in Table 1. An instance $\boldsymbol{x}$ can be divided into actionable features $\boldsymbol{x}^{act}$ and immutable

features $\boldsymbol{x}^{imt}$, denoted as $\boldsymbol{x} = (\boldsymbol{x}^{act}, \boldsymbol{x}^{imt})$. Immutable features cannot be changed while actionable features can be changed. Note that some actionable features are semi-actionable, which can be changed only in one direction, such as age or education degree.

Table 1: Frequently used notations

Symbol	Description
$\boldsymbol{x}$	A query instance to be explained
$\boldsymbol{c}$	A counterfactual explanation of $\boldsymbol{x}$
y	The desired target
f	The pretrained machine learning model
$\boldsymbol{x}^j$,$\boldsymbol{c}^j$	The feature j of $\boldsymbol{x}$, $\boldsymbol{c}$
$\boldsymbol{x}^{imt}$	The immutable features of $\boldsymbol{x}$
$\boldsymbol{x}^{act}$	The actionable features of $\boldsymbol{x}$
$\mathcal{F}_{imt}$	The set of immutable features
$\mathcal{F}_{act}$	The set of actionable features

2 RELATED WORK

The counterfactual explanation was firstly formulated as an optimization problem by [39] with the objective function in Eq. (1). This problem is also studied under terminologies such as recourse [37], inverse classification [22, 23], and contrastive explanation [11].

To guarantee *plausibility* of counterfactual explanations, [15] formulates the problem using the following objective function,

$$\boldsymbol{c}' = \underset{\boldsymbol{c}|p(\boldsymbol{c})>\gamma}{\arg\min}\, d(\boldsymbol{x}, \boldsymbol{c}), \quad s.t.\ f(\boldsymbol{c}') = y \tag{2}$$

where $p(\boldsymbol{c})$ tells the probability of the explanation $\boldsymbol{c}$ following the data distribution, and γ describes the threshold. Similarly, [10] measures the distance to the nearest observed data points, [11, 15, 32] adopt the reconstruction loss of an autoencoder or a variational autoencoder, [16] uses an out-of-distribution detector, called local outlier factor [4]. With a variational autoencoder, [32] minimizes the L_2 norm over the latent space and avoids the distance measure for heterogeneous data simultaneously.

To provide *diverse* explanations, [31] formulates the problem by defining the objective as follows,

$$C = \underset{\boldsymbol{c}_1,\ldots,\boldsymbol{c}_k}{\arg\min} \frac{1}{k}\sum_{i=1}^{k} \ell(f(\boldsymbol{c}_i), y) + \frac{\lambda_1}{k}\sum_{i=1}^{k} d(\boldsymbol{x}, \boldsymbol{c}_i) - \lambda_2\ \text{dpp\_diversity}(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_k) \tag{3}$$

where dpp_diversity$(\cdot)$ describes the diversity of the counterfactual set. In [35], the author generates the diverse set by prohibiting produced counterfactuals. The multiple initializations in [39] also have the diverse effect. [37] considers *actionability* of counterfactual explanations. [11, 38] enforce *sparsity* by adding L_0 or L_1 norm terms to penalize the changes over many features. [15, 19, 20, 28] consider *causal* relationships between features.

Most of the above works require a scalar cost function $d(\boldsymbol{x}, \boldsymbol{c})$ for measuring the changes over multiple features in the input space. [32] optimizes the scalar distance in the latent space, but the minimum explanation in latent space is not equivalent to the minimum changes in the input space [7], and their experiments reveal a higher degree of cost. Besides, minimizing the latent space distance can alter the semi-actionable features in an impossible direction, e.g., reducing the age. However, our skyline approach does not require a scalar cost function to measure the changes over different features.

How to set user constraints for an appropriate counterfactual remains a challenge [39] for above methods because users often have trouble expressing their preferences without knowing what alternative solutions are available [34]. Two solutions in the literature are: (1) a user tries several trails [39] using some imprecise constraints, then tunes some trade-off factors and stops until a final explanation is found. This approach requires running an algorithm multiple times, which is time-consuming; (2) directly offering a diverse set of counterfactuals [31, 35], but such counterfactuals do not necessarily meet the user preferences. Our skyline provides all alternative solutions, i.e., valid counterfactuals with minimum changes. With the skyline as the starting point, the user will formulate her preferences *interactively* through specifying the next query based on the examination of the results of previous queries to the skyline. The user does not need to have a clear choice of preferences from the start.

3 OVERVIEW

We assume that the followings are given: a machine learning model f, an input instance $\boldsymbol{x}$ having undesirable prediction by f, and the desired target prediction y. The goal is to identify promising counterfactual explanations $\boldsymbol{c}$ that convert the undesirable prediction to the target y with minimum cost measured by changes of features. Previous works (i.e., Eqs. (1), (2) and (3)) need to specify a scalar cost function over changes of multiple features, which can be difficult as discussed in Section 1.

To avoid specifying a scalar cost function over multiple features, our approach has two steps. The first step considers each actionable feature as an objective and finds all explanations with minimum changes under multi-objective optimization [12, 30]. The result is the set of "non-dominated" solutions over incomparable objectives, a.k.a. the skyline. In the second step, we propose a query interface to the skyline to help the user locate preferred counterfactuals interactively.

3.1 The Skyline Approach

For two solutions $\boldsymbol{c}$ and $\boldsymbol{c}'$, $\boldsymbol{c}$ is said to *dominate* $\boldsymbol{c}'$ on objectives $\boldsymbol{h}$, denoted by $\boldsymbol{c} \succ_{\boldsymbol{h}} \boldsymbol{c}'$, if $\boldsymbol{c}$ is no larger than $\boldsymbol{c}'$ in each objective and $\boldsymbol{c}$ is smaller than $\boldsymbol{c}'$ in at least one objective. A solution is *non-dominated* if it is not dominated by any other solution.

Consider the set of actionable features $\mathcal{F}_{act}$. For a given input $\boldsymbol{x}$ and an explanation $\boldsymbol{c}$, the change of $\boldsymbol{c}$ on feature j is defined by,

$$h_j(\boldsymbol{c}, \boldsymbol{x}) = |\boldsymbol{c}^j - \boldsymbol{x}^j|, \quad j \in \mathcal{F}_{act}. \tag{4}$$

The changes of $\boldsymbol{c}$ with respect to $\boldsymbol{x}$ are given by $h(\boldsymbol{c}, \boldsymbol{x}) = (h_1(\boldsymbol{c}, \boldsymbol{x}), \ldots, h_m(\boldsymbol{c}, \boldsymbol{x}))$, where $m = |\mathcal{F}_{act}|$. For a given search space $\boldsymbol{C}$ of counterfactuals and an input point $\boldsymbol{x}$, we define the *$\boldsymbol{x}$-skyline* of $\boldsymbol{C}$ as the set of $\boldsymbol{c}$ in $\boldsymbol{C}$ such that $h(\boldsymbol{c}, \boldsymbol{x})$ is not dominated by any $h(\boldsymbol{c}', \boldsymbol{x})$ for $\boldsymbol{c}'$ in $\boldsymbol{C}$.

In the following definition, p denotes the probability of following the data distribution, and $r(\boldsymbol{c}, \boldsymbol{x})$ returns 1 if the changes of $\boldsymbol{c}$ from $\boldsymbol{x}$ are in the valid directions, otherwise, returns 0. For example,

semi-actionable feature "working years" can be altered in only one direction. Any $\boldsymbol{c}$ reducing the "working years" will have $r(\boldsymbol{c}, \boldsymbol{x}) = 0$.

Definition 1. (*Skyline counterfactual explanation problem*) For a given search space $\boldsymbol{C}$ and an input $\boldsymbol{x}$, we want to find the $\boldsymbol{x}$-skyline of $\boldsymbol{C}$, subject to the constraints,

$$S = \arg\min_{\boldsymbol{c} \in \boldsymbol{C}} (h_1(\boldsymbol{c}, \boldsymbol{x}), h_2(\boldsymbol{c}, \boldsymbol{x}), \ldots, h_{|\mathcal{F}_{act}|}(\boldsymbol{c}, \boldsymbol{x})) \quad (5)$$

$$s.t. \quad \ell(f(\boldsymbol{c}), y) \leq \varepsilon \quad (6)$$

$$p(\boldsymbol{c}) > \gamma \quad (7)$$

$$r(\boldsymbol{c}, \boldsymbol{x}) = 1 \quad (8)$$

Eq. (6) ensures the closeness to the target prediction; Eq. (7) ensures that $\boldsymbol{c}$ follows the data distribution; Eq. (8) ensures the change in the allowed directions. These constraints ensure that S contains only valid changes. $\arg\min(\cdot)$ returns the $\boldsymbol{x}$-skyline of all satisfying counterfactuals, i.e., the non-dominated counterfactuals on the objectives $h_1, .., h_{|\mathcal{F}_{act}|}$.

The loss function ℓ is a general loss covering both regression and classification models. For example, for a regression model f, $\ell(f(\boldsymbol{c}) - y) = (f(\boldsymbol{c}) - y)^2$. For a classification model f, let $f_y(\boldsymbol{c})$ denote the probability of $\boldsymbol{c}$ predicted to have the class y. There should be no penalty when $f_y(\boldsymbol{c})$ is greater than a specified threshold [31], which can be specified using $\varepsilon = 0$ and the following hinge-loss function,

$$\ell(f(\boldsymbol{c}), y) = \max(0, \text{threshold} - f_y(\boldsymbol{c})). \quad (9)$$

3.2 Querying the Skyline

S contains all non-dominated counterfactual explanations satisfying the basic requirements in Eqs. (6)-(8). The size of S can still be large, and not all such counterfactual explanations are interesting to users. On the other hand, users may not be always clear about their preferences from the start, especially before knowing what alternatives are available. In this context, the skyline S provides exemplary alternatives for users to formulate their preferences. To this end, we propose the following interface to allow users to interactively explore what alternatives are available in the skyline,

SELECT TOP <an integer> <features or *>
FROM S
WHERE <constraints>
ORDER BY <ranking criteria> **ASC|DESC**

where the **ASC** and **DESC** specify the rank in ascending or descending order. This query returns the top <integer> counterfactuals in the skyline, ordered by <ranking criteria>, which satisfy <constraints> on the changes on actionable features. <features or *> specifies the features returned.

For example, suppose that there are three actionable features: "Education_Year", "Credit_Points", and "Salary". Instead of having exact preferences on the changes on these features, initially the user knows only that her salary can be increased by at most $1000. Then she queries the skyline using the above query with the constraint "Salary ≤ $1000". From the returned counterfactuals, the user notes that some solutions have no changes on "Salary" but have increases on "Credit_Points". Feeling that it is easier to increase credit points than increasing salary, the user queries the skyline second time with the constraint "Salary = 0 AND Credit_Point ≤ 100" and the ranking criteria on the number of unchanged features , in descending order. Suppose that the top 3 results returned are "Credit_Points=100", "Education_Year = 2", and "Credit_Points=50 AND Education_Year = 1". Finally, the user makes a decision from these alternatives based on her estimated difficulty of increasing credit points and increasing education years.

In this example, the user does not have precise and clear preferences from the start and uses the query interface to interactively formulate her preferences by recognizing what alternatives are available. The basic idea is similar to query refinement for search engine queries [29, 34], and is consistent with the cognition theory that recognition is easier than description [1].

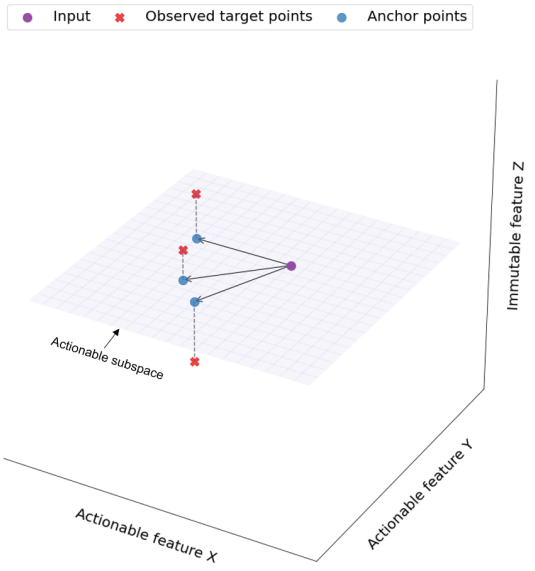

Figure 2: Purple color represents the input point x. Red colors represent the observed points o with the target prediction. Blue colors represent the projection o' of o onto the actionable subspace. Black arrows from input to anchor points specify the search space and directions.

4 SKYLINE COUNTERFACTUAL ALGORITHM

We now present our *skyline counterfactual algorithm (SkylineCF)* to solve the problem in Definition 1. The solution S depends on the search space $\boldsymbol{C}$. On one hand, $\boldsymbol{C}$ should be large enough to contain good solutions, on the other hand, it should allow efficient search. In the following, we first propose several choices of $\boldsymbol{C}$ and then present an algorithm for solving the problem in Definition 1 for a given $\boldsymbol{C}$.

4.1 The Search Space $\boldsymbol{C}$

A good counterfactual $\boldsymbol{c}$ should be close to the input $\boldsymbol{x}$ to have a small change, whereas any observed point $\boldsymbol{o}$ with the target y being the true label and predicted label in the data set is always a valid counterfactual for $\boldsymbol{x}$. Therefore, to find a good counterfactual $\boldsymbol{c}$,

Function 1 $AnchorSet(O, \boldsymbol{x})$

Input: A set of observed target points O, an input point $\boldsymbol{x}$.
Output: A set of non-dominated anchor points A.

$A' = \{\ \}$
for $\boldsymbol{o} \in O$ **do**
  $\boldsymbol{o}' = (\boldsymbol{o}^{act}, \boldsymbol{x}^{imt})$ ▷ Project $\boldsymbol{o}$ onto the actionable subspace
  if $p(\boldsymbol{o}') \geq \gamma$ & $\ell(f(\boldsymbol{o}'), y) \leq \varepsilon$ & $r(\boldsymbol{o}', \boldsymbol{x}) = 1$ **then**
    $A' = A' \bigcup \{\boldsymbol{o}'\}$ ▷ Add valid counterfactuals $\boldsymbol{o}'$ into A'
  end if
end for
$A = Skyline(A', \boldsymbol{x})$ ▷ Prune dominated anchors
Return A

we can move from $\boldsymbol{x}$ towards $\boldsymbol{o}$ and sample the points along the path at some step size, and eventually we will convert the current prediction to the target prediction. A short travel in this move is along the straight line from $\boldsymbol{x}$ to $\boldsymbol{o}$. Based on this idea, C could be the set of sampled points on the straight lines from $\boldsymbol{x}$ to $\boldsymbol{o}$. To support the straight line search, we assume that a categorical feature j is encoded using one-hot encoding in $\mathcal{F}_{act}$.

The question is how to choose the observed target points $\boldsymbol{o}$. In general, more and diverse observed points would lead to a larger search space, but also increase the search time. For a small dataset, we can consider all observed target points $\boldsymbol{o}$ with the target y being the true label and predicted label in the data set. For a large data set, we can group observed target points into clusters and consider the cluster centers as $\boldsymbol{o}$. Also, we continue referring to $\boldsymbol{o}$ as an observed point although it could be a cluster center. Note that a clustering algorithm may require a distance function, but this step is external to our method that is free of distance function. Other choices of $\boldsymbol{o}$ include several diverse observed points that are far from each other, nearest target neighbors around $\boldsymbol{x}$, or several observed target points that serve as the prototypes. Let O denote the set of chosen observed target points $\boldsymbol{o}$. Our algorithm below takes $\boldsymbol{O}$ as input but is independent of how $\boldsymbol{O}$ is chosen.

An observed target point $\boldsymbol{o}$ may have different values from $\boldsymbol{x}$ on immutable features, in which case the sampled points on the straight line from $\boldsymbol{x}$ to $\boldsymbol{o}$, except for $\boldsymbol{x}$, will change the values of immutable features, which are not valid counterfactual explanations. To avoid such changes, we replace $\boldsymbol{o}$ with another point $\boldsymbol{o}'$ obtained by replacing the immutable features of $\boldsymbol{o}$ with those of $\boldsymbol{x}$ while keeping all actionable features unchanged. Intuitively, $\boldsymbol{o}'$ is the projection of $\boldsymbol{o}$ onto the actionable subspace of $\boldsymbol{x}$, as shown in Figure 2, where the actionable subspace is determined by the immutable features of the input. To ensure $\boldsymbol{o}'$ is a valid counterfactual explanation, we require $p(\boldsymbol{o}') > \gamma$, $\ell(f(\boldsymbol{o}'), y) \leq \varepsilon$, and $r(\boldsymbol{o}', \boldsymbol{x}) = 1$. Only valid counterfactual explanations $\boldsymbol{o}'$ will serve as the search "destination", a.k.a. *anchor points* (blue points in Figure 2).

For the given collection of observed target points O, Function 1 computes the set of anchor points, as discussed above. Considering some anchor points $\boldsymbol{o}'$ dominate other anchor points, we only keep the non-dominated points for efficiency by applying the skyline operator $Skyline(\cdot)$, implemented as in [3, 5, 6, 13], for example.

Finally, we define our search space C as the set of sampled points on the straight lines from the input $\boldsymbol{x}$ to the anchor points in the anchor set returned by Function 1. Importantly, as we shall see in Section 4.2, there is no need to materialize the sampled points in C; instead, these points are implicitly represented by the straight lines from the input $\boldsymbol{x}$ to the anchor points and the step size for sampling.

Algorithm 1 Skyline Counterfactual Algorithm (SkylineCF)

Input: A set of observed target points O, an input point $\boldsymbol{x}$, and the number of sampled points on each line s.
Output: A skyline of counterfactual explanations.

$A \leftarrow AnchorSet(O, \boldsymbol{x})$ ▷ Obtain the anchor set
$S' \leftarrow LineSearch(A, \boldsymbol{x}, s)$ ▷ Find closest counterfactuals
$S \leftarrow Skyline(S', \boldsymbol{x})$ ▷ Return non-dominated counterfactuals
return S

Function 2 $LineSearch(A, \boldsymbol{x}, s)$

Input: Anchor set A, an input point $\boldsymbol{x}$, and the number of sampled points on each line s.
Output: A set of counterfactuals.

$S' = \{\}$
for $\boldsymbol{o}' \in A$ **do**
  for $i \leftarrow 1, s$ **do**
    $\boldsymbol{t} = \frac{s-i}{s} * \boldsymbol{x} + \frac{i}{s} * \boldsymbol{o}'$ ▷ i-th point on the line
    if $p(\boldsymbol{t}) \geq \gamma$ & $\ell(f(\boldsymbol{t}), y) \leq \varepsilon$ **then**
      $S' = S' \bigcup \{\boldsymbol{t}\}$ ▷ Add valid counterfactual $\boldsymbol{t}$
      Break ▷ Stop once the nearest one is found
    end if
  end for
end for
Return S'

4.2 SkylineCF

The main algorithm, SkylineCF, is given in Algorithm 1. The inputs contain a set of observed target points O, an instance $\boldsymbol{x}$, and the number of sampled points on each line s. $AnchorSet(O, \boldsymbol{x})$ finds the anchor set. $LineSearch(A, \boldsymbol{x}, s)$ in Function 2 searches over the straight line from $\boldsymbol{x}$ to every anchor point until it finds the first (i.e., nearest) valid counterfactual. Note that checking $r(\boldsymbol{t}, \boldsymbol{x}) = 1$ in $LineSearch(A, \boldsymbol{x}, s)$ is not needed because the anchor point $\boldsymbol{o}'$ satisfies $r(\boldsymbol{o}', \boldsymbol{x}) = 1$, which ensures that all sampled points on the line also satisfy this condition. On each line, s equally spaced points are searched. Exactly one counterfactual will be found for each anchor point in A because the anchor point is a valid solution. $Skyline(S', \boldsymbol{x})$ returns the $\boldsymbol{x}$-skyline of the counterfactuals in S'.

The number of sampled points s serves as the trade-off between granularity of search and efficiency of search. A smaller s allows a more efficient search but has a higher chance of missing the nearest valid counterfactual. Assume there is a neighborhood around $\boldsymbol{o}'$ in which the condition $p(\boldsymbol{t}) \geq \gamma$ & $\ell(f(\boldsymbol{t}), y) \leq \varepsilon$ is satisfied. So once we enter this neighborhood along the line search from $\boldsymbol{x}$ to $\boldsymbol{o}'$, this condition will remain satisfied for the remaining sampled points. Therefore, even if we miss the nearest valid counterfactual on this line, the next searched point is guaranteed to be a solution because it will satisfy the above condition, and this solution is at most one search step away from the nearest solution. The same idea also underlies the nearest neighbor classification [9] where

nearest neighbors are used to estimate the class label for a query point.

Our method is orthogonal to the choices of the implementations for testing $p(\boldsymbol{x}) \geq \gamma$ and for computing the skyline. As obtaining the prior probability distribution is challenging, we use out-of-distribution (OOD) detectors, e.g., [4, 25, 27], to test $p(\boldsymbol{x}) \geq \gamma$. In our experiments, we choose the tree-based Isolation Forest [25] by default because it avoids the use of distance functions. The $p(\cdot)$ can be replaced arbitrarily as long as it is free of distance functions. The skyline algorithms such as [3, 5, 6, 13] can be used for the $Skyline(\cdot)$ operator. We use Block-Nested-Loops (BNL) algorithm [3] because of its good efficiency as discussed in [13].

4.3 Complexity of Algorithm 1

$AnchorSet(O, \boldsymbol{x})$ of Function 1 iterates over each observed target point $\boldsymbol{o}$ in O to define the corresponding anchor point $\boldsymbol{o'}$. Checking the condition in line 4 takes constant time, given the pre-trained functions f and p. $Skyline(A', \boldsymbol{x})$ in line 8 when implemented by the BNL skyline algorithm [3] has the worst case complexity of $O(m * |A'|^2)$ and the average case complexity of $O(m * |A'|)$ [13], where m is the number of objectives (i.e., the number of actionable features). $LineSearch(A, \boldsymbol{x}, s)$ of Function 2 has the complexity of $O(|A| * s)$ because at most s points on each line are searched and there is one line for each anchor point in A. For $Skyline(S', \boldsymbol{x})$, from the discussion above, the worst case complexity is $O(m * |S'|^2)$ and the average case complexity is $O(m * |S'|)$.

To sum up, we note that $|A'| \leq |O|$ and $|S'| \leq |A| \leq |O|$. So the worst case complexity of Algorithm 1 is $O(m * |O|^2 + |O| * s)$, and the average case complexity is $O(m * |O| + |O| * s)$.

5 EXPERIMENTS

In this section, we assess the proposed method on three public datasets, comparing the results with state-of-the-art methods. After that, we present a case study to demonstrate how the user narrows down the candidates using the query interface to explore the skyline of counterfactual explanations interactively and incrementally.

5.1 Datasets

We consider the following datasets widely used in existing works [31, 32, 35] for experimental evaluation.

UCI Adult Dataset [21]. This dataset contains 48, 842 records of the 1994 US census database with 14 features (6 numerical, 1 ordinal, and 7 nominal features) describing the personal information. The target variable indicates whether personal income is above \$50, 000 or not. The yearly income of $\geq$ \$50, 000 is the positive class while the opposite is the negative class. The 37, 155 records is below \$50, 000. We fill missing values with mean values for numerical features and mode values for categorical features. A categorical feature is blocked into several coarse categories as in [40], then encoded into one-hot vector. We treat "capital-loss", "capital-gain", "occupation", "hours-per-week", and "workclass" as actionable features, and the rest as immutable features.

Give Me Some Credit (GMSC)[1]. This dataset was used to predict the probability that someone will experience financial distress in the next two years. It contains financial and demographic information of 150, 000 applicants where 139, 974 applicants are labeled as "good" (positive class) and 10, 026 applicants as "bad" (negative class). All 10 features are numerical. We applied the preprocessing [2] to fill the missing values, remove outliers, delete irrelevant features, etc. As the dataset is imbalanced, we select the first 10, 026 "good" applicants and all the 10, 026 "bad" applicants to form the final set. We treat "Age" and "NumberofDependents" as immutable features following [32].

HELOC Dataset[3]. This dataset was collected for the FICO explainable machine learning challenge to predict whether a user will repay her HELOC account over a two-year period with 23 numerical features. The predicted target "RiskPerformance" is binary. There are 5,000 "good" records (positive class) and 5, 459 "bad" records (negative class). We treat "ExternalRiskEstimate", "MSinceOldestTradeOpen" and "AverageMInFile" as immutable features following [32].

We normalize all features by the min-max scaler into the range $[0, 1]$ and randomly split the records into the train/test sets at the ratio of $4 : 1$ as in [31], and apply the 5-fold cross validation on the train set for tuning hyper-parameters. Then, we train a 3-layer multilayer perception (MLP) model with two hidden layer sizes of 20 and 10, using the Adam optimizer and 10^{-4} learning rate. The test accuracy for the three datasets is 85.03%, 76.85%, 72.81%, respectively. We use this MLP model as the black box f. As for the input instances $\boldsymbol{x}$ for producing counterfactual explanations for UCI Adult and GMSC, we consider the first 1, 000 true negative samples in the test set for saving time, and for Heloc, all 808 true negative samples. We report the average evaluation score of the selected input instances $\boldsymbol{x}$.

5.2 Evaluation Methods

We evaluate our method through *quantitative evaluation* and *use case evaluation*. For the quantitative evaluation, we consider the following evaluation metrics. The use case evaluation is through the exploration of the skyline using the query interface in Section 3.2.

Sparsity. The sparsity [31] is defined as the percentage of actionable features not changed,

$$Sparsity(\boldsymbol{x}, \boldsymbol{c}) = \frac{1}{|\mathcal{F}_{act}|} \sum_{j \in \mathcal{F}_{act}} 1_{\boldsymbol{x}^j = \boldsymbol{c}^j}. \tag{10}$$

$\boldsymbol{c}$ with a larger sparsity is preferred due to changes on fewer features.

Average Percentile Shift (APS). The change of numerical features is defined by the *average percentile shift* [32, 37]. Let $Q_j(\cdot)$ denote the percentile rank of a value relative to all the values of feature j of the whole dataset. $\mathcal{F}_{num}$ denotes the set of actionable numerical features. The APS is defined as follows,

$$APS(\boldsymbol{x}, \boldsymbol{c}) = \frac{1}{|\mathcal{F}_{num}|} \sum_{j \in \mathcal{F}_{num}} |Q_j(\boldsymbol{x}^j) - Q_j(\boldsymbol{c}^j)|. \tag{11}$$

Non-dominated Ratio (NR). Recall that a counterfactual represents valid minimum changes that flip the prediction in our search

[1] https://www.kaggle.com/c/GiveMeSomeCredit/overview

[2] https://www.kaggle.com/nicholasgah/eda-credit-scoring-top-100-on-leaderboard

[3] https://community.fico.com/s/explainable-machine-learning-challenge

space, whereas for each dominated counterfactual, there is a non-dominated counterfactual that has smaller changes. We define NR to measure the ability of returning only non-dominated counterfactuals. Note that our skyline algorithm has the maximum NR of 1 because it returns only non-dominated counterfactuals.

Rule-based Score (RBS). An important consideration for useful counterfactuals is whether the change matches with prior knowledge on the relation between the change and the target prediction. In particular, for some features, increasing (↑) the feature values will increase (↑) or decrease (↓) the target probability. We can represent such prior knowledge by rules and use them to check the feasibility of counterfactual explanations. A rule is generally written as,

$$\text{feature} \uparrow \rightarrow \text{target} \uparrow \text{ or } \downarrow$$

For a pre-selected set of rules R that models the prior knowledge, let $R(\boldsymbol{x}, \boldsymbol{c})$ denote the set of rules in R whose feature on the left-hand-side has a non-zero change in $\boldsymbol{c}$, and let $M(\boldsymbol{x}, \boldsymbol{c})$ denote the set of rules in $R(\boldsymbol{x}, \boldsymbol{c})$ that are satisfied by the relation between the target and the changes in $\boldsymbol{c}$. RBS is defined by,

$$\text{RBS}(\boldsymbol{x}, \boldsymbol{c}) = \frac{|M(\boldsymbol{x}, \boldsymbol{c})|}{|R(\boldsymbol{x}, \boldsymbol{c})|} \tag{12}$$

For the UCI Adult data, we create 4 rules with prior knowledge:

- R1: "Capital-gain" ↑ → "income" ↑.
- R2: "Capital-loss" ↑ → "income" ↓.
- R3: "Occupation" ↑ → "income" ↑, "Service" < "Admin" < "Blue-Collar" < "Sales" < "Other" < "Military" < "Professional" < "White-Collar".
- R4: "Workclass" ↑ → "income" ↑, "Other/Unknown" < "Private" < "Government" < "Self-Emplo-yed".

R1 and R2 are common sense. R3 is a rule for the categorical feature "Occupation" where the categories are ascendingly ordered by their corresponding average income on the whole dataset. Similarly, we create R4 for "Workclass", ordering the categories of "Workclass" by their average income.

For the GMSC data, "RevolvingUtilizationOfUnsecuredLines", "NumberOfTime30-59DaysPastDueNotWorse", "NumberOfTimes90-DaysLate", "NumberOfTime60-89DaysPastDueNotWorse", and "DebtRatio" are positively related to the "Bad" prediction, whereas "MonthlyIncome" are negatively related to the "Bad" prediction. Therefore, we have 6 rules for the GMSC data. For the Heloc data, we use the 16 rules discussed in the link [4]. These features are monotonically decreasing or increasing with respect to the "Bad" probability.

5.3 Baselines

We compare the proposed SkylineCF algorithm with state-of-the-art methods. Our method returns a skyline of variable size, we select top-k explanations from the skyline ranked by the above four evaluation metrics respectively. If the size of the skyline is less than k, we report all counterfactuals of skyline. For a fair comparison, we also generate k explanations from the baseline methods.

PlainCF [39]. This approach frames the counterfactual explanation as the objective in Eq. (1) and searches for the solution with gradient descent from a random initialization. The L_1 distance weighted by the inverse median absolute deviation (MAD) is chosen as the cost function in Eq. (1). We take k solutions obtained from k different random initializations.

DiCE [31]. This is one of the most popular methods released on GitHub. It finds k diverse counterfactuals based on the objective in Eq. (3), with L_1 distance weighted by MAD as the cost function.

Growing Spheres (GS) [23]. This algorithm searches the counterfactuals from random samples in a close sphere neighborhood of the input instance. The sphere grows until a counterfactual is found. A postprocessing step is adopted to enforce the sparsity. Because of the random sampling in the neighborhood, we possibly obtain different explanations in each run. We repeat this algorithm with different random samples until k counterfactuals are found.

C-CHVAE [32]. It maps the input $\boldsymbol{x}$ to the latent space using a variational autoencoder and searches the closest counterfactual from a neighborhood of the latent representation $\boldsymbol{z}$ like **GS** [23]. We also try several trails as **GS** to find k explanations.

For our SkylineCF algorithm, we set O as all true positive observations in the training set for covering as much as good solutions. We set the number of sampled points on each line s as 20, and the threshold in Eq. (9) as 0.7. The OOD threshold γ of our method is determined automatically as discussed in the original paper [25]. On the first two datasets, we set $r(\boldsymbol{x}, \boldsymbol{c}) = 1$ (see Eq. (8)) for all counterfactuals $\boldsymbol{c}$ for simplicity, and on the third dataset, we set $r(\boldsymbol{x}, \boldsymbol{c})$ such that the features "NumTradesOpeninLast12M", "MSinceMostRecentInqexcl7days" and "NumInqLast6M", which represent user's historical information, cannot be reduced.

5.4 Quantitative Evaluation

Figure 3 reports the results on the four quantitative evaluation metrics with the x-axis representing the number of counterfactuals, k, and y-axis representing the averaged metrics value of the k counterfactuals for all input instances considered. Our SkylineCF achieves the highest RBS and NR, almost the lowest APS and competitive sparsity over all k values considered on the three datasets.

The lower APS means that SkylineCF achieves the desirable outcome with smaller perturbations on continuous features. This result is reasonable as our algorithm finds the first (i.e., nearest) solution in the line search and filters dominated solutions. For sparsity, SkylineCF falls only behind GS as expected because GS [23] adopts the post-processing to enforce sparsity while SkylineCF does not optimize sparsity directly. SkylineCF guarantees the highest NR because it returns only non-dominated solutions while the other methods do not enforce the "non-dominated" requirement on returned solutions. GS's NR drops quickly for a larger k. This may result from its postprocessing on sparsity, which tends to return counterfactuals with changes on similar features, leading to dominated solutions. Our method has a better balance on sparsity and NR. For RBS, we note that the anchor point usually follows the relation specified by the predefined rules, so do the points on the straight line from the input instance to the anchor point.

The skyline of counterfactuals returned by SkylineCF provides a diverse set of candidate solutions for user's consideration. The user has the flexibility to select counterfactuals from such candidates to meet additional preferences. This aspect is evaluated next.

[4] https://github.com/Trusted-AI/AIX360/blob/master/examples/tutorials/HELOC.ipynb

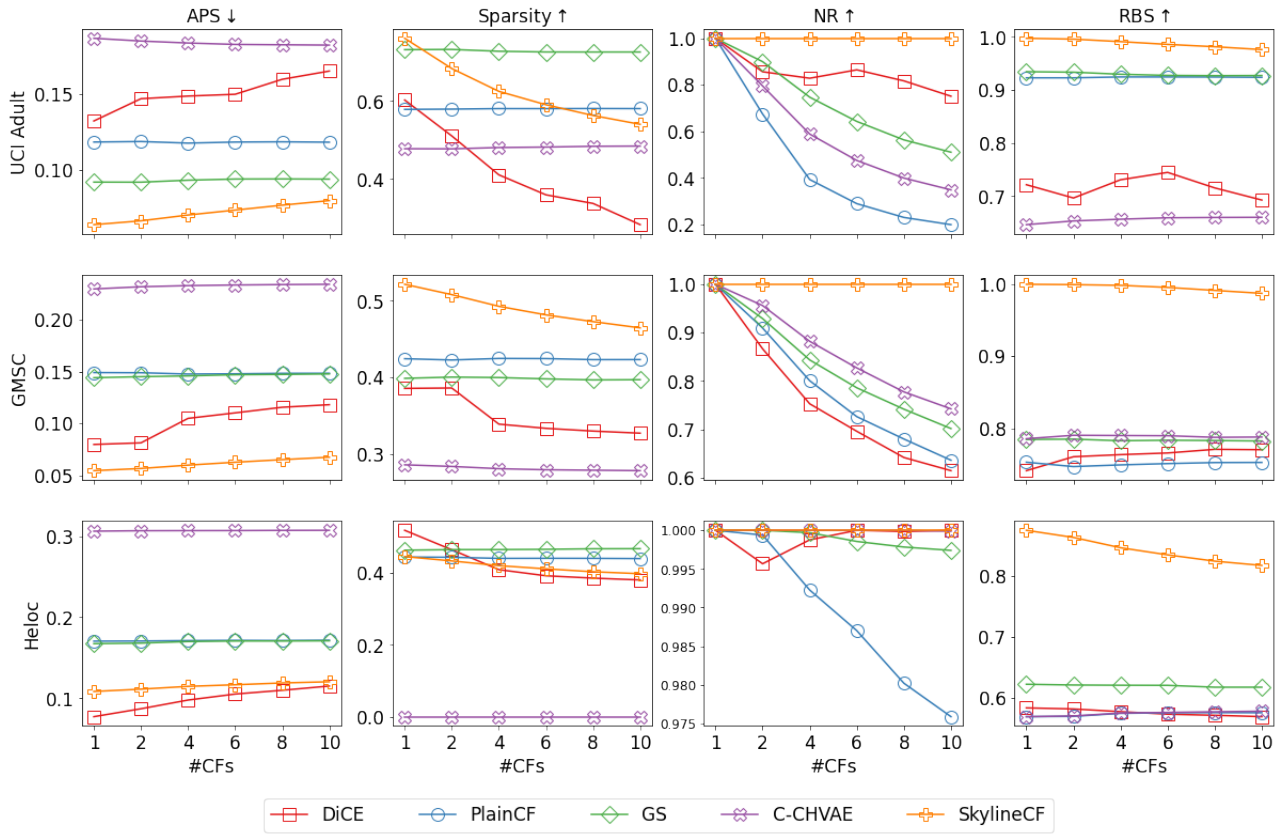

Figure 3: Each row represents the results on a dataset. Each column represents the results evaluated by a metric. The ↑/↓ represents that a higher/lower score is better.

5.5 Use Case Evaluation

In this section, we use the UCI Adult Dataset to illustrate how a user obtains satisfying counterfactual explanations by exploring the skyline interactively with our query interface. The yearly income of the user is $< \$50,000$ and is also predicted by the model f as $< \$50,000$. She wants some recommendation of counterfactuals that can increase her income to $> \$50,000$. As the background knowledge, the user knows that the actionable features are capital-gain, capital-loss, hours-per-week, workclass, and occupation. She may also know that some features are easier to be changed than others, for example, increasing hours-per-week might be hard but not impossible, so if there are other options, she would prefer not increasing hours-per-week. Importantly, while she does have some preferences, such preferences may not be specified precisely from the start without knowing the alternatives available. For this reason, it would be very hard for baseline methods to model user preferences through constrained optimization because they do not offer all available alternatives to the user.

Now, let consider how the user could use our skyline-query interface to find satisfying counterfactual explanations. The user's input instance and the query process are shown in Figure 4. The top table shows the feature values of the input instance. The first table on the second row shows the entire skyline of 58 counterfactual explanations (showing actionable features only), all can increase her income to $> \$50,000$. This skyline is produced from the 7,592 true positive training samples (recall $> \$50,000$ is the positive class). The skyline operator has removed all non-dominated counterfactuals. Since the size of skyline is still large, the user applies her preferences to narrow down the candidates using our query interface.

First query Q1: knowing that there are many alternatives available, the user applies her first preference of not increasing hours-per-week by issuing the first query Q1 with the constraint "hours-per-week<=40". 16 results are retrieved after this query.

Second query Q2: with 16 results retrieved, the user applies her next preference of not increasing capital-gain and capital-loss too much, say no more than 5,000 and 0, respectively. 3 results are retrieved.

Final selection: at this time, since only 3 results are remaining, the user can examine each closely. The user notices that the final selection will be a trade-off between an increase in capital-gain and a change of occupation. For example, the first result has a smaller increase in capital-gain but requires changing occupation to White-collar. Since increasing capital-gain is an easier option, the user selects the second and third results as the final candidates.

Discussion. This example highlights several interesting properties of our approach that stand out from existing works. First,

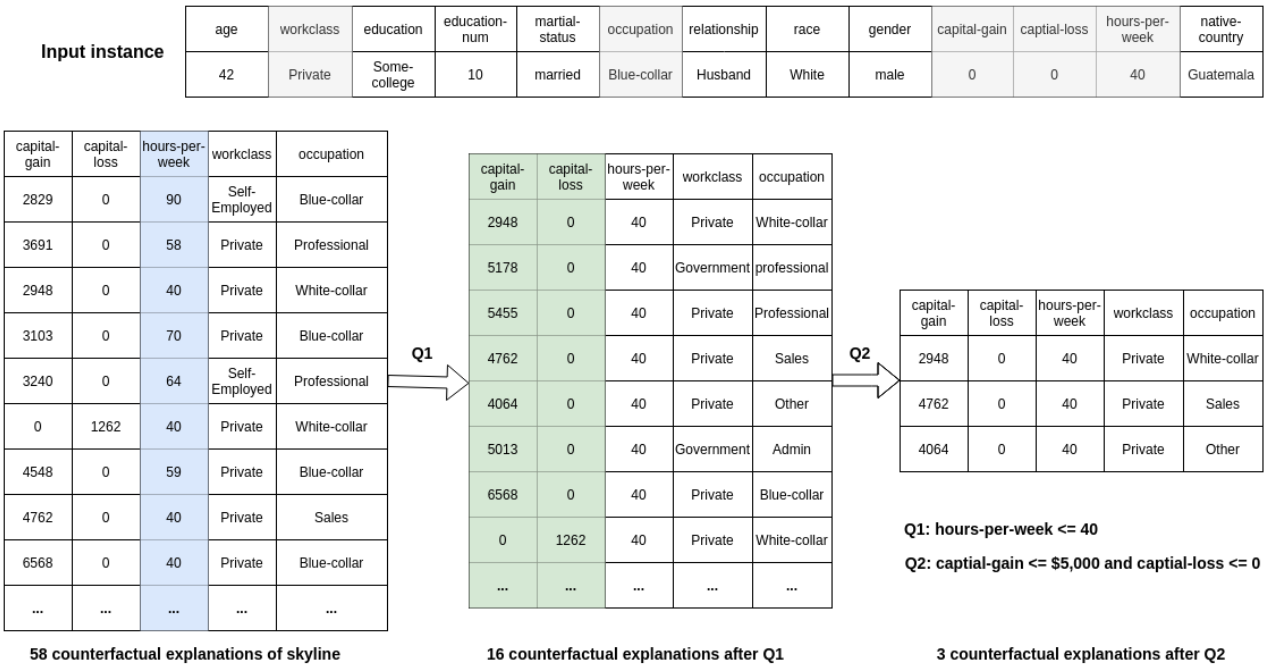

Figure 4: The use case of an input instance in UCI Adult Dataset. The shaded features in the input instance are actionable features. The query process starts with 58 counterfactual explanations in the skyline. With two queries Q1 and Q2, the user reaches a small set of candidates that can be looked at closely. Other contains the jobs of protective-services and tech-support following [40].

the skyline contains all and only valid counterfactuals that have minimum changes in our search space. Therefore, with the skyline as the starting point, the user has access to all candidate solutions. Existing works do not have this property. Second, the user does not need to have a clear and precise notion of preferences from the start, instead, the user will interactively formulate and prioritize her preferences "on-the-fly" based on the result of previous queries and impose such preferences through the next query. If many results are returned by the previous query, the user could tight up her preferences, and if no or few results are returned, she could give up some preferences. Third, while the user interactively formulates and explores her preferences, the SkylineCF is only performed once. In contrast, existing works require rerunning the search algorithm each time for refining her preferences.

6 CONCLUSION

Previous work minimizes the changes of counterfactual explanations through a scalar cost function, which is problematic because changes of different features are incomparable. To address this issue, we formulated the minimum change problem as finding the skyline of changes under the multi-objective optimization framework. We proposed a novel solution to this problem to find the skyline of changes. To our knowledge, this is the first work that finds the full set of valid counterfactual explanations with minimum changes. This completeness provides the user with a set of all possible candidates. We also presented a query interface to help the user narrow down suitable candidates through interactively and incrementally formulating her preferences based on previous query results. Our current work has not considered causal relations between the target and features, and feature correlations. In addition, our work for high-dimensional dataset is less efficient and effective because of the computation issue of skyline and too many explanations returned. Investigation into these issues will be our future work.

ACKNOWLEDGMENTS

This research is supported, in part, by Alibaba Group through Alibaba Innovative Research (AIR) Program and Alibaba-NTU Singapore Joint Research Institute (JRI), Nanyang Technological University, Singapore. This research is also supported, in part, by the National Research Foundation, Prime Minister's Office, Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019-003) and under its NRF Investigatorship Programme (NRFI Award No. NRF-NRFI05-2019-0002). The work of Ke Wang is partially supported by a Discovery Grant from Natural Sciences and Engineering Research Council of Canada. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

REFERENCES

[1] John Robert Anderson. 2000. *Learning and memory: An integrated approach*. John Wiley & Sons Inc.

[2] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015).

[3] Stephan Borzsony, Donald Kossmann, and Konrad Stocker. 2001. The skyline operator. In *Proceedings 17th international conference on data engineering*. IEEE, 421–430.

[4] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 93–104.

[5] Chee-Yong Chan, HV Jagadish, Kian-Lee Tan, Anthony KH Tung, and Zhenjie Zhang. 2006. Finding k-dominant skylines in high dimensional space. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. 503–514.

[6] Chee-Yong Chan, HV Jagadish, Kian-Lee Tan, Anthony KH Tung, and Zhenjie Zhang. 2006. On high dimensional skylines. In *International Conference on Extending Database Technology*. Springer, 478–495.

[7] Xingyu Chen, Chunyu Wang, Xuguang Lan, Nanning Zheng, and Wenjun Zeng. 2021. Neighborhood Geometric Structure-Preserving Variational Autoencoder for Smooth and Bounded Data Sources. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[8] Furui Cheng, Yao Ming, and Huamin Qu. 2020. DECE: Decision Explorer with Counterfactual Explanations for Machine Learning Models. *IEEE Transactions on Visualization and Computer Graphics* (2020).

[9] Thomas Cover and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13, 1 (1967), 21–27.

[10] Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. 2020. Multi-Objective Counterfactual Explanations. *arXiv preprint arXiv:2004.11165* (2020).

[11] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems*. 592–603.

[12] Michael TM Emmerich and André H Deutz. 2018. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing* 17, 3 (2018), 585–609.

[13] Parke Godfrey, Ryan Shipley, and Jarek Gryz. 2007. Algorithms and analyses for maximal vector computation. *The VLDB Journal* 16, 1 (2007), 5–28.

[14] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)* 51, 5 (2018), 1–42.

[15] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. 2019. Towards realistic individual recourse and actionable explanations in black-box decision making systems. *arXiv preprint arXiv:1907.09615* (2019).

[16] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Hiroki Arimura. 2020. DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, Christian Bessiere (Ed.). International Joint Conferences on Artificial Intelligence Organization*. 2855–2862.

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

[18] Amir-Hossein Karimi, Gilles Barthe, Bernhard Schölkopf, and Isabel Valera. 2020. A survey of algorithmic recourse: definitions, formulations, solutions, and prospects. *arXiv preprint arXiv:2010.04050* (2020).

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

[20] Amir-Hossein Karimi, Julius von Kügelgen, Bernhard Schölkopf, and Isabel Valera. 2020. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. *Advances in Neural Information Processing Systems* 33 (2020).

[21] Ron Kohavi. 1996. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid.. In *Kdd*, Vol. 96. 202–207.

[22] Michael T Lash, Qihang Lin, Nick Street, Jennifer G Robinson, and Jeffrey Ohlmann. 2017. Generalized inverse classification. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 162–170.

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

[24] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. 2019. The dangers of post-hoc interpretability: Unjustified counterfactual explanations. *arXiv preprint arXiv:1907.09294* (2019).

[25] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 413–422.

[26] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*. 4768–4777.

[27] Junshui Ma and Simon Perkins. 2003. Time-series novelty detection using one-class support vector machines. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, Vol. 3. IEEE, 1741–1745.

[28] Divyat Mahajan, Chenhao Tan, and Amit Sharma. 2019. Preserving causal constraints in counterfactual explanations for machine learning classifiers. *arXiv preprint arXiv:1912.03277* (2019).

[29] Gary Marchionini. 2006. Exploratory search: from finding to understanding. *Commun. ACM* 49, 4 (2006), 41–46.

[30] R Timothy Marler and Jasbir S Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization* 26, 6 (2004), 369–395.

[31] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 607–617.

[32] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. 2020. Learning Model-Agnostic Counterfactual Explanations for Tabular Data. In *Proceedings of The Web Conference 2020*. 3126–3132.

[33] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?". In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/2939672.2939778

[34] Tuukka Ruotsalo, Giulio Jacucci, Petri Myllymäki, and Samuel Kaski. 2014. Interactive intent modeling: Information discovery beyond search. *Commun. ACM* 58, 1 (2014), 86–92.

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

[36] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 3319–3328.

[37] Berk Ustun, Alexander Spangher, and Yang Liu. 2019. Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 10–19.

[38] Arnaud Van Looveren and Janis Klaise. 2019. Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584* (2019).

[39] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. arXiv:1711.00399 [cs.AI]

[40] Haojun Zhu. 2016. *Predicting Earning Potential using the Adult Dataset*. Retrieved December 5, 2016 from https://rpubs.com/H_Zhu/235617