Counterfactual Explanation Trees: Transparent and Consistent Actionable Recourse with Decision Trees

Kentaro Kanamori
Hokkaido University

Takuya Takagi
Fujitsu Limited

Ken Kobayashi
Fujitsu Limited
Tokyo Institute of Technology

Yuichi Ike
The University of Tokyo

Abstract

Counterfactual Explanation (CE) is a post-hoc explanation method that provides a perturbation for altering the prediction result of a classifier. An individual can interpret the perturbation as an "action" to obtain the desired decision results. Existing CE methods focus on providing an action, which is optimized for a given single instance. However, these CE methods do not address the case where we have to assign actions to multiple instances simultaneously. In such a case, we need a framework of CE that assigns actions to multiple instances in a transparent and consistent way. In this study, we propose Counterfactual Explanation Tree (CET) that assigns effective actions with decision trees. Due to the properties of decision trees, our CET has two advantages: (1) Transparency: the reasons for assigning actions are summarized in an interpretable structure, and (2) Consistency: these reasons do not conflict with each other. We learn a CET in two steps: (i) compute one effective action for multiple instances and (ii) partition the instances to balance the effectiveness and interpretability. Numerical experiments and user studies demonstrated the efficacy of our CET in comparison with existing methods.

1 INTRODUCTION

Complex machine learning models, such as deep neural networks and tree ensembles, have been applied to critical decision-making tasks in the real world (e.g., medical diagnoses, hiring decisions, and loan approvals). While these models have achieved high prediction accuracy, they often lack *explainability* (Doshi-Velez and Kim, 2017). Consequently, several post-hoc methods for extracting local explanations from each prediction of these models have been attracting much attention over the last few years (Ribeiro et al., 2016; Lundberg and Lee, 2017; Koh and Liang, 2017). These explanations assist human-users in understanding the decisions given by complex models and accepting them with confidence (Goodman and Flaxman, 2017).

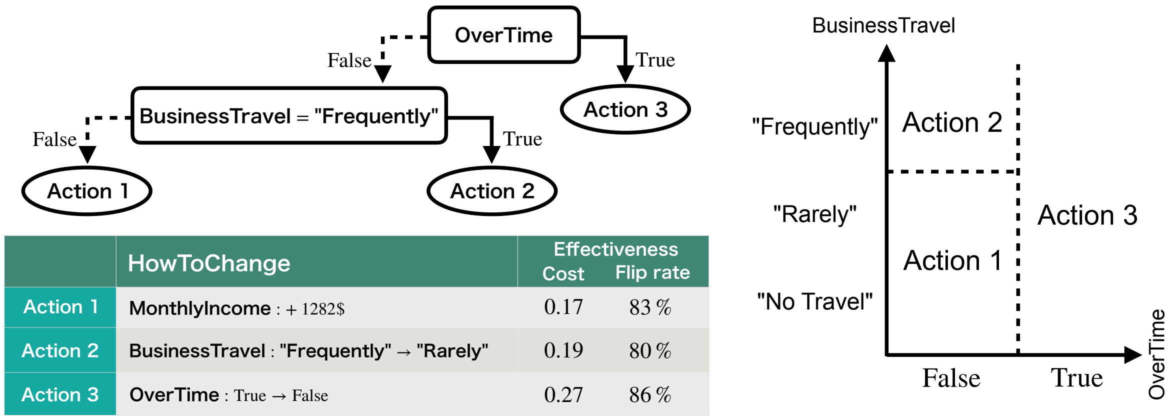
To provide users with better insights, post-hoc local explanations need to show both why the undesirable predictions are given and how users should act to obtain the desired prediction results (Miller, 2019). *Counterfactual Explanation (CE)* (Wachter et al., 2018) is a post-hoc explanation method that satisfies these requirements. For a given classifier $f\colon \mathcal{X} \to \mathcal{Y}$, a target class $y^* \in \mathcal{Y}$, and an instance $x \in \mathcal{X}$ such that $f(x) \neq y^*$, CE attempts to provide a perturbation vector a that flips the prediction result into the desired outcome, i.e., $f(x+a) = y^*$. The individual x can execute the perturbation a as an "*action*" for obtaining the desired decision result y^* from the classifier f (e.g., reducing the BMI and blood glucose levels to reduce the risk of diabetes). To achieve this, most of the existing studies consider the following optimization problem (Karimi et al., 2020b):

$$\underset{a \in \mathcal{A}}{\text{minimize}} \quad c(a \mid x) \quad \text{subject to} \quad f(x+a) = y^*,$$

where $\mathcal{A}$ is a set of feasible actions, i.e., perturbation vectors, and c is a cost function that measures the required efforts of executing an action a. Recently, this procedure of providing individuals with actions is also known as *Actionable Recourse* (Ustun et al., 2019).

In the practical deployment of machine learning models, however, actions are not necessarily executed by the individuals themselves, and are often assigned to multiple instances simultaneously, which contrast with the standard assumption of existing CE (Karimi et al., 2020b). For example, let us consider the case where a company uses a classifier to predict employee attrition,

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).



	HowToChange	Effectiveness	
		Cost	Flip rate
Action 1	MonthlyIncome : + 1282$	0.17	83 %
Action 2	BusinessTravel : "Frequently" → "Rarely"	0.19	80 %
Action 3	OverTime : True → False	0.27	86 %

Figure 1: Example of our CET learned on the IBM HR Analytics Employee Attrition dataset (Kaggle, 2017).

and attempts to assign an action to each employee for reducing one's attrition risk (Kaggle, 2017). In this case, these actions are executed by the company rather than each employee, e.g., promoting one or increasing one's salary. Such actions may result in changing the personnel or payroll systems and affect not only the individual but also the entire employees. In such a case, to assign actions to the entire employees, the company need to satisfy the following requirements:

1. *Transparency*: The company should explain how the actions are determined for the entire employees. That is, the company needs to ensure transparency by providing the reason why the action is assigned to each employee. If these reasons are unclear, the employees may doubt the validity of each assigned action, e.g., unjustified disparity among employees (Rawal and Lakkaraju, 2020).

2. *Consistency*: The company should provide consistent reasons for the assigned actions. For example, an employee who is transferred as the assigned action for the reason of "age $>$ 40" may complain to the company if another employee over 40 is promoted. To avoid such conflicts between employees, the company needs to provide the reasons that are consistent with each assigned action (Rudin and Shaposhnik, 2019).

Existing CE frameworks fail to satisfy these requirements because they only focus on giving an action for a single input instance and thus cannot take the entire input space into account, as with other local explanation methods (Ribeiro et al., 2018; Gao et al., 2021).

To satisfy these requirements, we aim to partition the input space in an interpretable manner and assign an appropriate action to each subspace. For that purpose, we introduce *Counterfactual Explanation Tree (CET)*, a new framework of CE for assigning actions to multiple instances with a decision tree $h \colon \mathcal{X} \to \mathcal{A}$. A decision tree is a popular model that consists of a set of exclusive if-then-else rules expressed as a binary tree (Breiman et al., 1984). Since it is interpretable for humans (Rudin, 2019) and has high potential representability (Lee and Jaakkola, 2020; Vidal and Schiffer, 2020), decision trees have been used not only in supervised learning, but also in other practical decision-making (see, e.g., Bertsimas et al. (2019); Elmachtoub et al. (2020); Lakkaraju and Rudin (2017); Silva et al. (2020)). Due to the characteristics of decision trees, our CET has the following advantages:

1. For any instance $x \in \mathcal{X}$, our CET can provide a reason for an assigned action as a form of a *rule*, i.e., subspace of $\mathcal{X}$ where the action is assigned. Since these reasons are summarized as an interpretable tree structure, we can easily understand how the actions are assigned to the instances over the entire input space $\mathcal{X}$ (Guidotti et al., 2018).

2. Our CET guarantees to assign a unique pair of an action and the reason to any instance since it partitions the input space $\mathcal{X}$ into distinct subspaces and only one leaf is determined for any input (Freitas, 2014). It ensures that there is no conflict of reasons for assigned actions between instances.

Our strategy to learn a CET is the following two steps: (i) assign the most effective single action to multiple instances in the sense of its required cost and rate of flipping the predictions into the desired result, and (ii) construct a decision tree partitioning a set of instances to balance a trade-off between the effectiveness and interpretability. We formulate these problems and propose an efficient algorithm to solve them.

1.1 Our Contributions

Our contributions are summarized as follows:

1. We introduce CET, a decision tree that assigns an effective action to an input instance over the input space. By taking advantage of decision trees, our CET (1) provides a transparent process of assigning actions, and (2) assigns a unique pair of an action and the reason to any instance.

2. We formulate the task of learning a CET from a given set of instances as an optimization problem, and propose an algorithm to solve it with stochastic local search. Our algorithm can utilize a pruning strategy based on an objective bound.

3. We conduct experiments on public datasets to evaluate the efficacy of our CET by comparison with the existing methods. Additionally, our user studies showed that our CET is easy for human-users to understand.

Figure 1 presents an example of our CET on the IBM HR Analytics Employee Attrition dataset (Kaggle, 2017). The task is to predict the attrition risks of the individual employees. We trained a CET on the dataset and a LightGBM classifier (Ke et al., 2017). Figure 1 demonstrates that our CET is (1) easy to understand how each action is assigned to instances over the input space, and (2) able to assign a unique pair of an action and the reason as a form like "this action is effective for 86% of the employees who work overtime" to any instance. We note that our CET can use various existing cost functions (e.g., max percentile shift (Ustun et al., 2019)), and quickly assign actions for unseen instances once it is trained, like amortized explanation (Chen et al., 2018; Jethani et al., 2021).

1.2 Related Work

Several post-hoc methods for extracting local explanations from complex models have been proposed (Ribeiro et al., 2016; Koh and Liang, 2017; Lundberg and Lee, 2017). *Counterfactual Explanation (CE)*, also referred to as *Actionable Recourse*, is one of the post-hoc explanation methods that have attracted increasing attention in recent years (Karimi et al., 2020b). Most of the existing CE methods focus on providing a single action (Wachter et al., 2018; Karimi et al., 2020a; Kanamori et al., 2020; Schut et al., 2021) or multiple diverse actions (Ustun et al., 2019; Mothilal et al., 2020) for a given single instance.

While various local explanation methods have been proposed, recent studies have pointed out some issues with them, e.g., the lack of robustness (Ghorbani et al., 2019; Dombrowski et al., 2019) and improper assumptions (Barocas et al., 2020; Venkatasubramanian and Alfano, 2020; Karimi et al., 2021). One of the critical problems is that the region where the explanations can be applied is unclear (Ribeiro et al., 2018; Rudin and Shaposhnik, 2019). Due to this issue, local explanation methods often fail to explain the global behavior of complex models (Rudin, 2019), and have a potential risk of malicious manipulation (Aivodji et al., 2019).

To address the above issue, several frameworks for a global summary of local explanations have been proposed (Pedreschi et al., 2019). Our CET is most closely related to AReS (Rawal and Lakkaraju, 2020), which is a global summary of the actions expressed by a two-level rule set. AReS assists users to globally understand the recourse, i.e., actions, which are extracted from complex classifiers. However, AReS cannot ensure transparency and consistency in the process of assigning actions due to the properties of rule sets; that is, (1) it does not necessarily cover the entire input space, and (2) it may assign multiple actions to a single input instance (Freitas, 2014). MAME (Ramamurthy et al., 2020) and GIME (Gao et al., 2021) are global summaries of general local explanation methods (e.g., LIME (Ribeiro et al., 2016)), which are expressed by hierarchical clustering and interpretable topic modeling, respectively. However, to the best of our knowledge, there is no existing method that explicitly achieves both transparency and consistency.

1.3 Notation and Setting

For a positive integer $n \in \mathbb{N}$, we write $[n] \coloneqq \{1, \dots, n\}$. For a proposition ψ, $\mathbb{I}[\psi]$ denotes the indicator of ψ; that is, $\mathbb{I}[\psi] = 1$ if ψ is true, and $\mathbb{I}[\psi] = 0$ if ψ is false.

Throughout this paper, we consider a *binary classification problem* as a prediction task, which is sufficient for CE. We can reduce a multi-class classification problem to a binary classification problem between the target class and other classes. We denote input and output domains $\mathcal{X} \subseteq \mathbb{R}^D$ and $\mathcal{Y} = \{-1, +1\}$, respectively. We call a vector $x = (x_1, \dots, x_D) \in \mathcal{X}$ an *instance*, and a function $f : \mathcal{X} \to \mathcal{Y}$ a *classifier* to be explained.

2 GROUP-WISE COUNTERFACTUAL EXPLANATION

We first review an existing individualized CE framework (Ustun et al., 2019; Karimi et al., 2020b). We then give its naive extension to group-wise setting and point out its drawback. Finally, we present our proposed formulation and show its theoretical properties.

2.1 Individualized CE and Its Naive Extension to Group-wise CE

Individualized CE For an instance $x \in \mathcal{X}$, we define an *action* as a perturbation vector $a \in \mathbb{R}^D$ such that $x+a \in \mathcal{X}$. As with existing CE methods (e.g., Ustun et al. (2019); Kanamori et al. (2020)), we assume that we are given a set of feasible actions $\mathcal{A}(x) \subseteq \mathbb{R}^D$ such that $\mathbf{0} \in \mathcal{A}(x)$ and $\mathcal{A}(x) \subseteq \{a \in \mathbb{R}^D \mid x+a \in \mathcal{X}\}$.

For an instance $x \in \mathcal{X}$ and an action $a \in \mathcal{A}(x) \subseteq \mathbb{R}^D$, a *cost function* $c \colon \mathcal{X} \times \mathbb{R}^D \to \mathbb{R}_{\geq 0}$ measures the required effort of a with respect to x. To appropriately evaluate the required effort among actions, several useful cost functions have been proposed, such as the max percentile shift (Ustun et al., 2019). Throughout this paper, we assume $c(\mathbf{0} \mid x) = 0$, which is satisfied by most of the existing cost functions.

For a given classifier $f \colon \mathcal{X} \to \mathcal{Y}$ and an instance $x \in \mathcal{X}$ such that $f(x) \neq +1$, the aim of *Counterfactual Explanation (CE)* is to find an action a that alters the prediction result into $f(x+a) = +1$ and minimizes its cost $c(a \mid x)$. This task can be formulated as follows:

$$\underset{a \in \mathcal{A}(x)}{\text{minimize}} \; c(a \mid x) \quad \text{subject to} \quad f(x+a) = +1. \tag{1}$$

In the rest of this paper, we fix f and c and omit them if it is clear from the context. Note that our framework presented in Section 3 can be applied to any f and c.

Naive Extension and Drawback By extending the above individualized CE, we consider *group-wise CE* as a problem of assigning a single effective action to a given set of N instances $X \subseteq \mathcal{X}$, where $f(x) \neq +1$ for any $x \in X$. A naive formulation for this task is to find an action a that satisfies $f(x+a) = +1$ for any $x \in X$ and minimizes the sum of costs $c(a \mid x)$ over X. Let $\mathcal{A}(X) \coloneqq \bigcap_{x \in X} \mathcal{A}(x)$ be the set of feasible actions for X. Then, we can formulate the task as follows:

$$\begin{aligned} a^* = \; & \underset{a \in \mathcal{A}(X)}{\arg\min} && \textstyle\sum_{x \in X} c(a \mid x) \\ & \text{subject to} && f(x+a) = +1, \quad \forall x \in X. \end{aligned} \tag{2}$$

However, actions that alter the prediction results of all the given instances tend to be costly. To demonstrate this issue, we consider the same settings as Ustun et al. (2019). Let f be a linear classifier $f(x) = \operatorname{sgn}(w^\top x)$ with a parameter $w \in \mathbb{R}^D$. We assume $\mathcal{A}(x) = \mathbb{R}^D$ for any $x \in \mathcal{X}$ and $c(a \mid x) = c_x \cdot \|a\|$, where $c_x > 0$ is a constant depending on x. In the following proposition, we show an upper bound on the gap of the optimal costs between individualized and group-wise CE.

Proposition 1 (Upper Bound on Cost Gap)**.** *For a set of instances $X \subseteq \mathcal{X}$, let a^* be an optimal solution to problem* (2)*. For an instance $x \in X$, let $c^*(x)$ be the optimal value of problem* (1)*. Then, we have $c(a^* \mid x) - c^*(x) \leq c_x \cdot \|x - x^\circ\|$, where $x^\circ = \arg\min_{x \in X} w^\top x$.*

Proposition 1 implies that an upper bound on the cost gap between individualized and group-wise CE depends on the farthest instance $x^\circ \in X$ from the decision boundary of f. Let us show an example where the cost in (2) achieves that upper bound. Assume that we have many instances near the decision boundary and only one instance x° far from the boundary. The required cost for flipping all the results depends on x°, which results in a higher cost value. In other words, for an instance $x \in X$, an optimal action a^* of (2) may be unrealistic in the sense of its cost $c(a^* \mid x)$ depending on other instances in X. This is because the constraint in (2) that alters all the output $f(x)$ of $x \in X$ is too strict. These results suggest a risk that the constraints $f(x+a) = +1$ for all $x \in X$ prevents us from obtaining an effective action.

2.2 Proposed Formulation of Group-wise CE

To obtain an effective action, we balance a trade-off between the cost $c(a \mid x)$ and constraint $f(x+a) = +1$ over $x \in X$. We relax the hard constraint $f(x+a) = +1$ for all $x \in X$ as with Wachter et al. (2018), and define an *invalidity* score $i_\gamma \colon \mathcal{X} \times \mathbb{R}^D \to \mathbb{R}_{\geq 0}$ as follows:

$$i_\gamma(a \mid x) \coloneqq c(a \mid x) + \gamma \cdot l_{01}(f(x+a), +1),$$

where $\gamma > 0$ is a trade-off parameter, and $l_{01}(\hat{y}, y) = \mathbb{I}[\hat{y} \neq y]$ is the 0–1 loss function. We use the invalidity $i_\gamma(a \mid x)$ as a measure for evaluating the effectiveness of an action a with respect to an instance x. Then, we consider the following optimization problem.

Problem 1 (CE for Multiple Instances)**.** Given a set of N instances $X \subseteq \mathcal{X}$ such that $\forall x \in X \colon f(x) \neq +1$, a set of actions $\mathcal{A}(X) = \bigcap_{x \in X} \mathcal{A}(x)$, and a parameter $\gamma > 0$, find an action $a^* \in \mathcal{A}(X)$ that is an optimal solution for the following problem:

$$\underset{a \in \mathcal{A}(X)}{\text{minimize}} \quad g_\gamma(a \mid X) \coloneqq \textstyle\sum_{x \in X} i_\gamma(a \mid x).$$

Optimization It is unclear whether existing CE methods can be directly extended to solve Problem 1. By extending existing CE methods based on *mixed-integer linear optimization (MILO)* (Ustun et al., 2019; Kanamori et al., 2020; Parmentier and Vidal, 2021), we can formulate Problem 1 as an MILO problem for several types of classifiers. We present MILO formulations for linear classifiers, tree ensembles, and deep ReLU networks in the supplementary materials.

Monotonicity of g_γ For the objective function $g_\gamma(a \mid X)$, we show its monotonicity with respect to a binary partition of X in the following proposition.

Proposition 2 (Monotonicity of g_γ)**.** *For a set of instances $X \subseteq \mathcal{X}$, we write $a^*_X \coloneqq \arg\min_{a \in \mathcal{A}(X)} g_\gamma(a \mid X)$. Let $X_1, X_2 \subset X$ be distinct subsets of X such that $X_1 \cup X_2 = X$ and $X_1 \cap X_2 = \emptyset$. Then, we have*

$$g_\gamma(a^*_X \mid X) \geq g_\gamma(a^*_{X_1} \mid X_1) + g_\gamma(a^*_{X_2} \mid X_2).$$

Proposition 2 indicates that we can assign more effective actions to the instances $x \in X$ in terms of their objective value $g_\gamma(a \mid X)$, i.e., cost $c(a \mid x)$ and loss $l_{01}(f(x+a), +1)$, by partitioning X into two distinct subsets and optimizing an action for each of the subsets. However, it is undesirable to increase the total number of subsets, i.e., assigned actions, because it reduces the interpretability of the entire process of assigning actions. In the following sections, we discuss how to partition a given set of instances X into distinct subsets such that a more effective action is assigned to each of the subsets, and how to balance the trade-off between the effectiveness and interpretability.

3 PROBLEM STATEMENT

Following the formulation of Problem 1, we introduce our *Counterfactual Explanation Tree (CET).*

3.1 Counterfactual Explanation Tree

For a set of feasible actions $\mathcal{A} \subseteq \mathbb{R}^D$, a CET is a *decision tree* $h \colon \mathcal{X} \to \mathcal{A}$ assigning an action for an input instance $x \in \mathcal{X}$. It consists of a set of exclusive if-then-else rules expressed as a binary tree structure. For x, a CET h assigns an action $a_l \in \mathcal{A}$ corresponding to the leaf $l \in \mathcal{L}(h)$ that x reaches, where $\mathcal{L}(h)$ is the set of leaves in h. The leaf l is determined by traversing the tree from the root depending on the *branching rule* of each internal node, which is expressed as a statement, e.g., $x_d \leq b$ with a feature $d \in [D]$ and threshold $b \in \mathbb{R}$.

For a CET h, let $r_l \subseteq \mathcal{X}$ be the subspace corresponding to a leaf $l \in \mathcal{L}(h)$. Each subspace r_l is determined by the branching rules on the path from the root to l, which can be interpreted as a *rule* for the assigned action a_l (Guidotti et al., 2018). The set of such subspaces $\{r_l \mid l \in \mathcal{L}(h)\}$ gives a partition of the input space $\mathcal{X}$ (Freitas, 2014). It implies that a CET h assigns a unique pair of an action a_l and a rule r_l to any input instance $x \in \mathcal{X}$, and can be expressed as $h(x) = \sum_{l \in \mathcal{L}(h)} a_l \cdot \mathbb{I}[x \in r_l]$. These two properties ensure that CET satisfies transparency and consistency. Figure 1 shows an example of a CET and its partition.

3.2 Problem Definition

Now we formulate a learning problem of a CET h from a given set of instances X. To guarantee the feasibility of assigned actions, let $\mathcal{H}$ be a set of CETs h satisfying $h(x) \in \mathcal{A}(x)$ for any $x \in X$. Then, we formulate a task of learning a CET $h \in \mathcal{H}$ from X as follows.

Problem 2 (Learning CET)**.** Given a set of N instances $X \subseteq \mathcal{X}$ such that $\forall x \in X \colon f(x) \neq +1$ and parameters $\gamma, \lambda > 0$, find a CET $h^* \in \mathcal{H}$ that is an optimal solution for the following optimization problem:

$$\begin{aligned} \underset{h \in \mathcal{H}}{\text{minimize}} \quad & o_{\gamma,\lambda}(h \mid X) \\ & := \tfrac{1}{N} \textstyle\sum_{x \in X} i_\gamma(h(x) \mid x) + \lambda \cdot |\mathcal{L}(h)|. \end{aligned}$$

While the first term of the learning objective $o_{\gamma,\lambda}(h \mid X)$ evaluates the average invalidity $i_\gamma(a \mid x)$, i.e., cost $c(a \mid x)$ and loss $l_{01}(f(x+a), +1)$, of the assigned actions $a = h(x)$ for $x \in X$, the second term $|\mathcal{L}(h)|$ is the total number of the leaves, i.e., actions, included in h. By tuning the parameter λ, we can adjust the trade-off between the effectiveness of the actions assigned by a CET h and the interpretability of h. Again note that the framework of our CET can be applied to any classifier f and cost function c of existing CE methods.

4 OPTIMIZATION FRAMEWORK

In this section, we propose a learning algorithm for a CET. As with the learning problems for standard decision trees, Problem 2 is a combinatorial optimization problem and finding an exact optimal solution is challenging. Moreover, unlike standard decision trees, we need to optimize both branching rules of internal nodes and actions of leaves simultaneously. It implies that Problem 2 includes Problem 1 as its subproblems.

A naive algorithm is a greedy top-down partitioning strategy like CART (Breiman et al., 1984) that recursively determines a branching rule of each internal node to most improve the objective value after splitting. However, to determine a branching rule of each internal node, we need to solve Problem 1 for the number of candidate branching rules, which is computationally infeasible. In preliminary experiments, we also observed that it did not yield a decision tree of good quality in terms of the invalidity score i_γ. This is because the approach often chose an inappropriate branching rule at a node near the root and thus failed to partition input instances X such that they are assigned an effective action in each leaf they reach.

From the above observations, we propose an algorithm based on the stochastic local search. The stochastic local search has been shown to be suitable for learning non-standard rule models (Wang, 2019; Pan et al., 2020). Our algorithm consists of two steps: determine branching rules of internal nodes with the stochastic local search strategy, and optimize an action assigned to each leaf by solving Problem 1 independently.

4.1 Leaf Size Bound

Before we describe the details of our algorithm, we derive an upper bound on the optimal size of CETs to prune the search space. We show an upper bound on the leaf size $|\mathcal{L}(h^*)|$ of an optimal CET h^* as follows.

Algorithm 1 Stochastic Local Search for CET.

```
Input: set of instances $X$, trade-off parameters $\gamma, \lambda$, set of candidate branching rules $\mathcal{R}$, maximum number of iterations $T$, and accept condition Accept.
Output: CET $h^*$.
$h^{(0)} \leftarrow$ generate an initial solution randomly;
$h^* \leftarrow h^{(0)}$;
for $t = 1, 2, \dots, T$ do
    $\delta \sim$ random();
    if $\delta \leq 1/3$ and $|\mathcal{L}(h^{(t-1)})| < (\gamma + \lambda)/\lambda$ then
        $h^{(t)} \leftarrow$ insert a node with a rule $r \in \mathcal{R}$ to $h^{(t-1)}$ randomly;
    else if $\delta \leq 2/3$ then
        $h^{(t)} \leftarrow$ delete a node from $h^{(t-1)}$ randomly;
    else
        $h^{(t)} \leftarrow$ replace the rule of a node in $h^{(t-1)}$ with another rule $r \in \mathcal{R}$ randomly;
    end if
    for $l \in \mathcal{L}(h^{(t)})$ do
        $a_l^{(t)} \leftarrow \arg\min_{a \in \mathcal{A}(X_l^{(t)})} g_\gamma(a \mid X_l^{(t)})$;
    end for
    if Accept$(t, h^{(t-1)}, h^{(t)})$ is False then
        $h^{(t)} \leftarrow h^{(t-1)}$;
    end if
    $h^* \leftarrow \arg\min_{h \in \{h^*, h^{(t)}\}} o_{\gamma,\lambda}(h \mid X)$;
end for
return $h^*$;
```

Theorem 1 (Leaf Size Bound)**.** *Let h^* be an optimal solution for Problem 2, i.e., $h^* = \arg\min_{h \in \mathcal{H}} o_{\gamma,\lambda}(h \mid X)$. Then, we have $|\mathcal{L}(h^*)| \leq \frac{\gamma+\lambda}{\lambda}$.*

Theorem 1 indicates that the optimal leaf size $|\mathcal{L}(h^*)|$ is upper bounded by some constant determined by the trade-off parameters γ and λ. It also suggests to us how to determine the trade-off parameters γ and λ.

4.2 Stochastic Local Search Algorithm

We present an algorithm for Problem 2 based on the stochastic local search. We assume a set of candidate branching rules $\mathcal{R}$. An element $r \in \mathcal{R}$ is a statement with respect to an instance x, e.g., $x_d \leq b$ for continuous features or $x_d = b$ for categorical features, where $d \in [D]$ and $b \in \mathbb{R}$. We can obtain $\mathcal{R}$ by some discretization techniques, as with recent studies on learning decision trees (Hu et al., 2019; Aglin et al., 2020).

For a CET $h \in \mathcal{H}$, let $X_l = \{x \in X \mid x \in r_l\}$ be the set of instances that reach a leaf $l \in \mathcal{L}(h)$. Since $\{r_l \mid l \in \mathcal{L}(h)\}$ is a partition of the input space $\mathcal{X}$, it gives a partition of X; that is, $\bigcup_{l \in \mathcal{L}(h)} X_l = X$ and $X_l \cap X_{l'} = \emptyset$ for any $l, l' \in \mathcal{L}(h)$. Therefore, we can rewrite our learning objective $o_{\gamma,\lambda}(h)$ as follows:

$$o_{\gamma,\lambda}(h) = \frac{1}{N} \sum\nolimits_{l \in \mathcal{L}(h)} g_\gamma(a_l \mid X_l) + \lambda \cdot |\mathcal{L}(h)|.$$

It suggests that if branching rules of the internal nodes in h, i.e., a partition of X, is determined, then we can optimize an action $a_l \in \mathcal{A}(X_l)$ assigned to each of the leaves $l \in \mathcal{L}(h)$ by solving Problem 1 independently.

Algorithm 1 presents our proposed algorithm. In Algorithm 1, we first randomly generate an initial solution $h^{(0)}$, and then iteratively update it until the number of iteration reaches a given maximum number $T \in \mathbb{N}$. Each iteration $t \in [T]$ consists of two update steps. First, we update the previous solution $h^{(t-1)}$ to $h^{(t)}$ with the stochastic local search strategy. The update is done by three edit operations with approximately equal probabilities: (1) *insert* an internal node with a randomly selected rule $r \in \mathcal{R}$ into a random position of $h^{(t-1)}$, (2) *delete* a node of $h^{(t-1)}$ randomly, and (3) *replace* the rule of a randomly selected node of $h^{(t-1)}$ with another rule $r \in \mathcal{R}$ randomly. Note that we prune the search space by excluding the insert operation when the leaf size $|\mathcal{L}(h^{(t-1)})|$ exceeds the upper bound of Theorem 1. Second, we optimize an action $a_l^{(t)}$ of each leaf $l \in \mathcal{L}(h^{(t)})$ by solving Problem 1 for the instances $X_l^{(t)}$ that reach l. In each iteration, the update is accepted depending on a given accept condition Accept$(t, h^{(t-1)}, h^{(t)})$. Following previous studies (Wang, 2019; Pan et al., 2020), we accept it with probability $p(t) = \exp\left(\frac{o_{\gamma,\lambda}(h^{(t-1)}) - o_{\gamma,\lambda}(h^{(t)})}{C_0^{1-\frac{t}{T}}}\right)$, where C_0 is a base temperature of simulated annealing. The probability $p(t)$ gradually decreases with iterations t.

5 EXPERIMENTS

To investigate the efficacy and interpretability of our CET, we conducted numerical experiments and user studies. All the code was implemented in Python 3.7[1]. All the experiments were conducted on 64-bit macOS Catalina 10.15.6 with Intel Core i9 2.4 GHz CPU and 64 GB memory.

5.1 Experimental Settings

Cost Function As a cost function c, we used the *Max Percentile Shift (MPS)* (Ustun et al., 2019) defined as

$$c(a \mid x) = \max_{d \in [D]} |Q_d(x_d + a_d) - Q_d(x_d)|,$$

where Q_d is the cumulative distribution function (CDF) with respect to a feature d. Note that the value of MPS is bounded by $[0, 1]$. Compared to other existing cost functions, MPS is suitable for evaluating actions assigned to all the instances due to two advantages (Ustun et al., 2019): (i) MPS is scale-invariant

[1] All the code and scripts for our experiments are available at `https://github.com/kelicht/cet`.

Table 1: Results of 10-fold cross validation.

(a) LightGBM classifier (Ke et al., 2017)

Dataset	**Method**	**Train**			**Test**		
		Cost	**Loss**	**Invalidity**	**Cost**	**Loss**	**Invalidity**
Attrition	Clustering	0.055 ± 0.04	0.951 ± 0.03	1.0 ± 0.01	0.047 ± 0.05	0.958 ± 0.06	1.01 ± 0.02
	AReS	0.436 ± 0.06	0.435 ± 0.07	0.871 ± 0.04	0.45 ± 0.08	0.298 ± 0.09	0.748 ± 0.09
	CET	0.349 ± 0.1	0.4 ± 0.11	0.749 ± 0.05	0.383 ± 0.12	0.318 ± 0.19	0.701 ± 0.12
German	Clustering	0.034 ± 0.02	0.915 ± 0.05	0.949 ± 0.04	0.039 ± 0.02	0.917 ± 0.05	0.956 ± 0.04
	AReS	0.452 ± 0.09	0.232 ± 0.05	0.683 ± 0.11	0.467 ± 0.12	0.265 ± 0.08	0.732 ± 0.14
	CET	0.103 ± 0.01	0.301 ± 0.07	0.404 ± 0.07	0.107 ± 0.02	0.276 ± 0.11	0.384 ± 0.1

(b) TabNet classifier (Arik and Pfister, 2021)

Dataset	**Method**	**Train**			**Test**		
		Cost	**Loss**	**Invalidity**	**Cost**	**Loss**	**Invalidity**
Attrition	Clustering	0.509 ± 0.16	0.487 ± 0.24	0.996 ± 0.23	0.515 ± 0.16	0.513 ± 0.23	1.03 ± 0.23
	AReS	0.161 ± 0.1	0.235 ± 0.1	0.396 ± 0.1	0.164 ± 0.1	0.281 ± 0.16	0.445 ± 0.18
	CET	0.312 ± 0.12	0.198 ± 0.14	0.51 ± 0.19	0.296 ± 0.13	0.258 ± 0.21	0.554 ± 0.23
German	Clustering	0.01 ± 0.01	0.981 ± 0.02	0.99 ± 0.01	0.009 ± 0.01	0.99 ± 0.01	0.999 ± 0.01
	AReS	0.18 ± 0.08	0.405 ± 0.14	0.585 ± 0.2	0.188 ± 0.09	0.427 ± 0.17	0.614 ± 0.22
	CET	0.121 ± 0.08	0.242 ± 0.21	0.362 ± 0.19	0.12 ± 0.08	0.234 ± 0.21	0.355 ± 0.19

and able to reflect the distribution on $\mathcal{X}$ unlike norm-based cost functions, and (ii) MPS is easy to interpret and compare between multiple instances since it represents the difficulty of a for x as the maximal change in percentile $|Q_d(x_d + a_d) - Q_d(x_d)|$ over $d \in [D]$.

Comparison Baselines We compare our **CET** with two baseline methods: (1) **Clustering**, which first partitions a given set of instances into a few clusters by K-means clustering and then assigns an action to each of the clusters by solving Problem 1, and (2) **AReS** (Rawal and Lakkaraju, 2020), a rule set assigning actions for input instances. To adapt AReS to our setting, we modified its submodular optimization framework to handle MPS as a cost measure, and tuned its hyperparameters to minimize the average invalidity i_γ on validation datasets. The implementation details are provided in the supplementary materials.

Datasets and Classifiers We used Attrition ($N = 1470, D = 44$) (Kaggle, 2017) and German ($N = 1000, D = 40$) (Dua and Graff, 2017) datasets. We transformed each categorical feature into a one-hot encoded vector. The task for the Attrition (resp. German) dataset is to predict whether employees will leave their company (resp. customers will default on their loan). As complex classifiers f, we used LightGBM (Ke et al., 2017) and TabNet (Arik and Pfister, 2021), which are renowned as state-of-the-art models for tabular datasets. We trained the baseline methods and our **CET** on training instances that had received undesirable predictions, such as “high risk of attrition.” We also used the ℓ_2-regularized logistic regression classifiers to analyze the trade-off between the effectiveness and interpretability of each method.

Evaluation Criteria For examining assigned actions a, we measured the average values of the cost $c(a \mid x)$, loss $l_{01}(f(x+a), y^*)$, and invalidity $i_\gamma(a \mid x)$. To make it easier to compare the values of the cost and loss, we set $\gamma = 1.0$. We measured these criteria not only on training instances but also on test instances to evaluate the generalization performance of each method (Gao et al., 2021). We also measured the total number of actions and running times.

5.2 Experimental Results

5.2.1 Performance Comparison

First, we compared the performance of our **CET** with baselines. For interpretability, we set the number of clusters for **Clustering** and the maximum number of recourse-rules, i.e., pairs of a rule and corresponding action, for **AReS** to 8 (Miller, 1956). We chose $\lambda = 0.02$ and $T = 3000$ for **CET** based on hold-out validation. We report the average values of each evaluation criterion measured on the train and test datasets.

Table 1 presents the results of 10-fold cross validation. The average accuracy of LightGBM and TabNet on the Attrition dataset (resp. German dataset) were 85.3% and 80.5% (resp. 69.5% and 70.6%), respectively. The average number of instances to train the baseline methods and our **CET** on the Attrition dataset (resp. German dataset) were 229.1 for LightGBM and 274.1 for TabNet (resp. 310.1 for LightGBM and 505.4 for TabNet), respectively. One example of our CET learned on the Attrition dataset is shown in Figure 1. Owing to the page limitation, the detailed results are provided in the supplementary materials. From these results, we observe the following findings:

- Our **CET** achieved lower invalidity for all the datasets and classifiers than **Clustering**, while **Clustering** optimizes the same invalidity i_γ as ours. This result indicates the efficacy of our Algorithm 1 that partitions instances based on the invalidity of the action assigned to each leaf.
- Compared to **AReS**, our **CET** achieved lower cost and loss for most of the datasets and classifiers ($11/16 = 68.75\%$). Notably, the average number of leaves in our **CET**, i.e., the total number of actions, was 6.38, whereas **AReS** contained 8 actions. These results indicate that our **CET** succeeded in assigning effective actions to instances with fewer actions than **AReS**.

Transparency and Consistency The average ratios of the instances (1) that were assigned no recourse-rule and (2) that were assigned multiple recourse-rules by **AReS** were 20.87% and 27.15%, respectively. These results indicate that **AReS** failed in ensuring transparency and consistency, i.e., assigning unique pairs of an action and the reason over the entire input space. In contrast, we note that our **CET** always succeeded due to the properties of decision trees.

From these findings, our **CET** could assign more effective actions in terms of cost and loss than **AReS**, while **CET** ensured transparency and consistency.

Computational Complexity Regarding the computational time, the average running time of **Clustering** was $1,585$ seconds since it only executes the K-means clustering and assigns an action to each cluster by solving Problem 1. Conversely, **AReS** and **CET** were much slower than **Clustering**, and the average running times were $15,099$ and $12,696$ seconds, respectively. We observed that **AReS** often spent a significant amount of time on preprocessing to generate candidate recourse-rules, especially for the German dataset. We can control the computational complexity of **CET** by tuning the maximum number of iteration T of Algorithm 1. The convergence analyses of Algorithm 1 are presented in the supplementary materials.

5.2.2 Trade-off Analysis

Next, we analyze a trade-off between the effectiveness and interpretability. We randomly split each dataset into training (75%) and test (25%) instances, and trained ℓ_2-regularized logistic regression classifiers. Then, we trained the baseline methods and our **CET** on the training instances by varying the number of actions for the baselines and λ for **CET**, respectively. We measured the cost and loss of the assigned actions to the test instances by each methods.

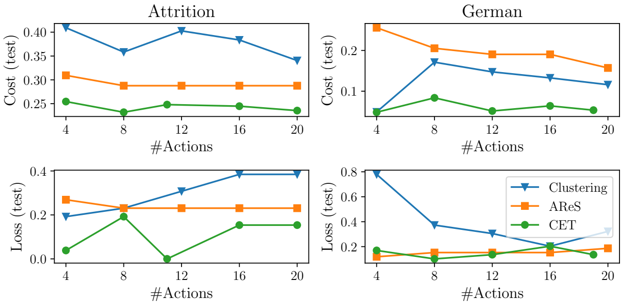


Figure 2: Analyses of trade-off between effectiveness and interpretability.

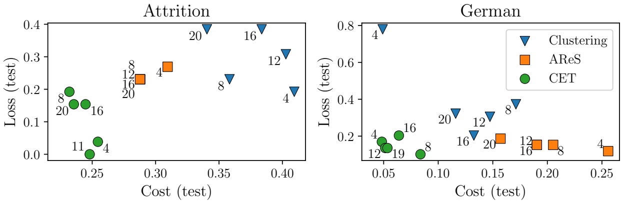


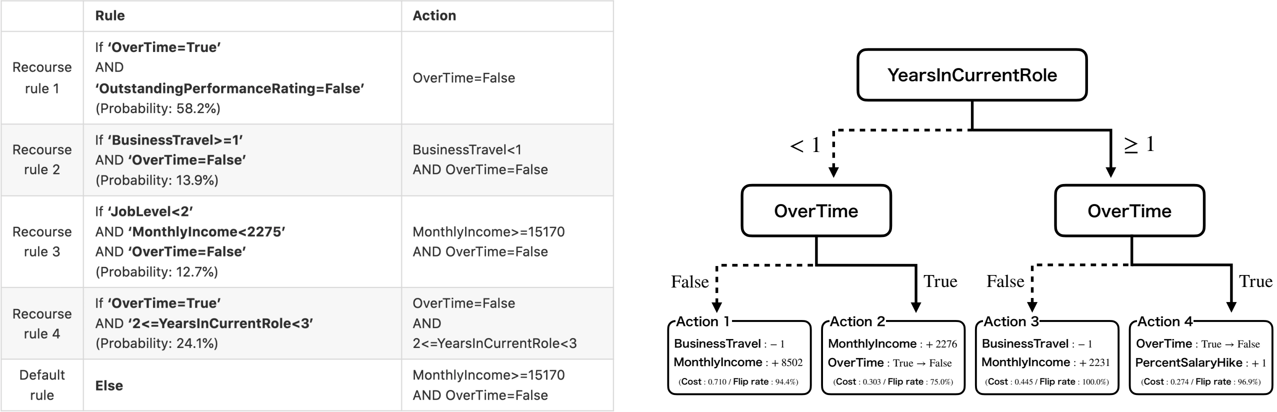
Figure 3: Scatter plot of Cost and Loss, where the number displayed at each point indicates #Actions.

Figure 2 presents the trade-off between the number of actions (#Actions) and the average values of cost and loss. Note that increasing the number of actions does not necessarily decrease the average values of both cost and loss on test instances since each method optimizes the sum of them on training instances as its learning objective. The results on training instances are presented in Figure 5 of the supplementary materials. Figure 3 shows the scatter plots of the cost and loss, where the number displayed at each point indicates #Actions. From these results, we confirmed that our **CET** stably outperformed the baselines. It indicates that our **CET** could assign more effective actions than the baselines at different levels of the interpretability.

5.3 User Studies

Finally, to investigate whether our **CET** is easy for human-users to understand, we conducted user studies with 35 participants. Each participant work in research and development departments related to artificial intelligence in a private company.

Settings We used the Attrition dataset and trained



	Rule	**Action**
Recourse rule 1	If **‘OverTime=True’** AND **‘OutstandingPerformanceRating=False’** (Probability: 58.2%)	OverTime=False
Recourse rule 2	If **‘BusinessTravel>=1’** AND **‘OverTime=False’** (Probability: 13.9%)	BusinessTravel<1 AND OverTime=False
Recourse rule 3	If **‘JobLevel<2’** AND **‘MonthlyIncome<2275’** AND **‘OverTime=False’** (Probability: 12.7%)	MonthlyIncome>=15170 AND OverTime=False
Recourse rule 4	If **‘OverTime=True’** AND **‘2<=YearsInCurrentRole<3’** (Probability: 24.1%)	OverTime=False AND 2<=YearsInCurrentRole<3
Default rule	**Else**	MonthlyIncome>=15170 AND OverTime=False

(a) AReS (Rawal and Lakkaraju, 2020)

(b) CET (ours)

Figure 4: Examples of learned AReS and CET shown to the participants in user study.

Table 2: Experimental results of user study.

Method	**User Acc.**	**Confidence**	**Time [s]**
Clustering	68.75%	2.16 ± 1.06	1079.0 ± 368
AReS	95.12%	3.32 ± 0.60	784.8 ± 202
CET	100.0%	3.16 ± 1.06	674.0 ± 392

LightGBM as a classifier f. We trained **Clustering**, **AReS**, and **CET** on the instances predicted as high risk of attrition by f. Our user studies were carried out in the following steps: (1) randomly assign a participant to one of the methods, (2) show an explanation of the assigned method; show the action and center of each cluster for **Clustering**, the rule set for **AReS**, and the decision tree for **CET**, (3) show an instance predicted as high risk of attrition by f, and (4) show five actions and ask the participant to answer which action is given by the assigned method for the instance. We also asked participants to rate their confidence level on a scale of 1 (not confident at all) to 5 (very confident). For each participant, we provided four instances to answer. We compare the three methods in terms of (i) the user accuracy, i.e., the proportion of the correct answers, (ii) the average confidence, and (iii) the average time to answer. The participants whose answer time exceeded one hour were excluded when we calculated the average time. See the supplementary materials for examples of our questions.

Results The numbers of participants who answered at least one question were 8 for **Clustering**, 11 for **AReS**, and 16 for **CET**, respectively. Examples of **AReS** and **CET** shown to the participants are presented in Figure 4. Table 2 summarizes their results, where "User Acc.," "Confidence," and "Time" show the proportion of the correct answers, average confidence, and average time to answer, respectively. We find that the user accuracy of **CET** was the highest out of all the methods, while the number of participants of **CET** was the largest. The average confidence of **CET** was smaller than **AReS**, but the difference was comparatively small. Notably, the average time for **CET** was the fastest among all the methods. From these results, we confirmed that the behavior of our **CET** was easily understood by human-users.

6 CONCLUSION

We proposed Counterfactual Explanation Tree (CET) that assigns effective actions to input instances by a decision tree. Our CET achieves transparency and consistency in the process of assigning actions to the entire population of an input space. To learn a CET, we first introduced a framework of group-wise CE for assigning a single action to given multiple instances. Then, we proposed an algorithm for learning a CET based on the stochastic local search with a theoretical pruning strategy. Our experiments and user studies demonstrated the efficacy and interpretability of our CET by comparing it with existing methods.

In future work, we plan to develop a more efficient learning algorithm to handle large datasets. Because the computational time of Algorithm 1 in our experiments mainly depended on that of solving Problem 1 by MILO at each node, we expect the scalability issue would be alleviated if we can solve Problem 1 more efficiently. Furthermore, we will extend our model to deal with interactions among multiple instances, as mentioned by Karimi et al. (2020b). It is also interesting future work to incorporate existing individualized CE methods that provide practical actions, e.g., multiple diverse actions (Mothilal et al., 2020), ordered actions (Kanamori et al., 2021), and causal interventions (Karimi et al., 2021), into our CET.

Acknowledgements

We wish to thank Hiroki Arimura for making a number of valuable suggestions. We also thank the anonymous reviewers for their insightful comments. This work was supported in part by JSPS KAKENHI Grant-in-Aid for JSPS Research Fellow 20J20654, Early-Career Scientists 21K17817, JST ACT-X JPMJAX2108, and JST CREST JPMJCR18K3.

References

G. Aglin, S. Nijssen, and P. Schaus. Learning optimal decision trees using caching branch-and-bound search. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 3146–3153, 2020.

U. Aivodji, H. Arai, O. Fortineau, S. Gambs, S. Hara, and A. Tapp. Fairwashing: the risk of rationalization. In *Proceedings of the 36th International Conference on Machine Learning*, pages 161–170, 2019.

S. Õ. Arik and T. Pfister. TabNet: Attentive interpretable tabular learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 6679–6687, 2021.

S. Barocas, A. D. Selbst, and M. Raghavan. The hidden assumptions behind counterfactual explanations and principal reasons. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 80–89, 2020.

D. Bertsimas, J. Dunn, and N. Mundru. Optimal prescriptive trees. *INFORMS Journal on Optimization*, 1(2):164–183, 2019.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

J. Chen, L. Song, M. Wainwright, and M. Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *Proceedings of the 35th International Conference on Machine Learning*, pages 883–892, 2018.

Z. Cui, W. Chen, Y. He, and Y. Chen. Optimal action extraction for random forests and boosted trees. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 179–188, 2015.

A.-K. Dombrowski, M. Alber, C. Anders, M. Ackermann, K.-R. Müller, and P. Kessel. Explanations can be manipulated and geometry is to blame. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 13589–13600, 2019.

F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv*, arXiv:1702.08608, 2017.

D. Dua and C. Graff. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2017. Accessed Feb. 21st, 2022.

A. Elmachtoub, J. C. N. Liang, and R. Mcnellis. Decision trees for decision-making under the predict-then-optimize framework. In *Proceedings of the 37th International Conference on Machine Learning*, pages 2858–2867, 2020.

A. A. Freitas. Comprehensible classification models: A position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10, 2014.

J. Gao, X. Wang, Y. Wang, Y. Yan, and X. Xie. Learning groupwise explanations for black-box models. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pages 2396–2402, 2021.

A. Ghorbani, A. Abid, and J. Y. Zou. Interpretation of neural networks is fragile. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 3681–3688, 2019.

B. Goodman and S. Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57, 2017.

R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):1–42, 2018.

X. Hu, C. Rudin, and M. Seltzer. Optimal sparse decision trees. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7265–7273, 2019.

N. Jethani, M. Sudarshan, Y. Aphinyanaphongs, and R. Ranganath. Have we learned to explain?: How interpretability methods can learn to encode predictions in their interpretations. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, pages 1459–1467, 2021.

Kaggle. IBM HR Analytics Employee Attrition & Performance. `https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset`, 2017. Accessed Feb. 21st, 2022.

K. Kanamori, T. Takagi, K. Kobayashi, and H. Arimura. DACE: Distribution-aware counterfactual explanation by mixed-integer linear optimization. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 2855–2862, 2020.

K. Kanamori, T. Takagi, K. Kobayashi, Y. Ike, K. Uemura, and H. Arimura. Ordered counterfactual explanation by mixed-integer linear optimization. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 11564–11574, 2021.

A.-H. Karimi, G. Barthe, B. Balle, and I. Valera. Model-agnostic counterfactual explanations for consequential decisions. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 895–905, 2020a.

A.-H. Karimi, G. Barthe, B. Schölkopf, and I. Valera. A survey of algorithmic recourse: definitions, formulations, solutions, and prospects. *arXiv*, arXiv:2010.04050, 2020b.

A.-H. Karimi, B. Schölkopf, and I. Valera. Algorithmic recourse: From counterfactual explanations to interventions. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 353–362, 2021.

G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 3149–3157, 2017.

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1885–1894, 2017.

H. Lakkaraju and C. Rudin. Learning Cost-Effective and Interpretable Treatment Regimes. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 166–175, 2017.

G.-H. Lee and T. S. Jaakkola. Oblique decision trees from derivatives of relu networks. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4765–4774, 2017.

G. A. Miller. The magical number seven, plus or minus two : Some limits on our capacity for processing information. *The Psychological Review*, 63(2):81–97, 1956.

T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

R. K. Mothilal, A. Sharma, and C. Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617, 2020.

D. Pan, T. Wang, and S. Hara. Interpretable companions for black-box models. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 2444–2454, 2020.

A. Parmentier and T. Vidal. Optimal counterfactual explanations in tree ensembles. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8422–8431, 2021.

D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, and F. Turini. Meaningful explanations of black box AI decision systems. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 9780–9784, 2019.

K. N. Ramamurthy, B. Vinzamuri, Y. Zhang, and A. Dhurandhar. Model agnostic multilevel explanations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 5968–5979, 2020.

K. Rawal and H. Lakkaraju. Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 12187–12198, 2020.

M. T. Ribeiro, S. Singh, and C. 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*, pages 1135–1144, 2016.

M. T. Ribeiro, S. Singh, and C. Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1527–1535, 2018.

C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206–215, 2019.

C. Rudin and Y. Shaposhnik. Globally-consistent rule-based summary-explanations for machine learning models: Application to credit-risk evaluation. In *Proceedings of INFORMS 11th Conference on Information Systems and Technology*, pages 1–19, 2019.

L. Schut, O. Key, R. Mc Grath, L. Costabello, B. Sacaleanu, M. Corcoran, and Y. Gal. Generating interpretable counterfactual explanations by implicit minimisation of epistemic and aleatoric uncertainties. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, pages 1756–1764, 2021.

T. Serra, C. Tjandraatmadja, and S. Ramalingam. Bounding and counting linear regions of deep neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4558–4566, 2018.

A. Silva, M. Gombolay, T. Killian, I. Jimenez, and S.-H. Son. Optimization methods for interpretable differentiable decision trees applied to reinforcement learning. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 1855–1865, 2020.

B. Ustun, A. Spangher, and Y. Liu. Actionable recourse in linear classification. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, pages 10–19, 2019.

S. Venkatasubramanian and M. Alfano. The philosophical basis of algorithmic recourse. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 284–293, 2020.

T. Vidal and M. Schiffer. Born-again tree ensembles. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9743–9753, 2020.

S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 31:841–887, 2018.

T. Wang. Gaining free or low-cost interpretability with interpretable partial substitute. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6505–6514, 2019.

Supplementary Material:
Counterfactual Explanation Trees: Transparent and Consistent Actionable Recourse with Decision Trees

A OMITTED PROOFS

A.1 Proof of Proposition 1

As mentioned in the main paper, we are given (i) a linear classifier $f(x) = \operatorname{sgn}(w^\top x)$, where $w \in \mathbb{R}^D$ is a coefficient vector, (ii) a feasible action set $\mathcal{A}(x) = \mathbb{R}^D$ for any $x \in \mathcal{X}$, and (iii) a cost function $c(a \mid x) = c_x \cdot \|a\|$, where $c_x > 0$ is a constant depending on x. To prove Proposition 1, we first show the following three lemmas.

Lemma 1 (Optimal Cost for Individualized CE (Ustun et al., 2019))**.** *For $x \in \mathcal{X}$ with $f(x) \neq +1$, let $c^*(x)$ be the optimal value for the problem in Eq. (1), i.e.,*

$$c^*(x) = \min_{a \in \mathcal{A}(x)} c(a \mid x) \quad \text{subject to} \quad f(x+a) = +1.$$

Then, we have $c^*(x) = c_x \cdot \frac{|w^\top x|}{\|w\|}$.

Lemma 2. *For a coefficient vector w of f, let $x, x' \in \mathcal{X}$ be instances such that $w^\top x \leq w^\top x'$ and $f(x) = f(x') = -1$. Then, we have $f(x+a) = +1 \implies f(x'+a) = +1$.*

Proof. From the definition of f, $f(x+a) = +1$ implies $w^\top(x+a) \geq 0$. Since $w^\top x \leq w^\top x'$, we have

$$w^\top(x'+a) = w^\top x' + w^\top a \geq w^\top x + w^\top a = w^\top(x+a) \geq 0.$$

Since $w^\top(x'+a) \geq 0$, we obtain $f(x'+a) = +1$. □

Lemma 3. *For a given set of N instances $X = \{x^{(1)}, \dots, x^{(N)}\} \subset \mathcal{X}$ such that $f(x) \neq +1$ for any $x \in X$, let $x^\circ = \arg\min_{x \in X} w^\top x$. Then, we consider the problem in Eq. (1) for x°, i.e.,*

$$\underset{a \in \mathcal{A}(x^\circ)}{\text{minimize}} \;\; c(a \mid x^\circ) \quad \text{subject to} \quad f(x^\circ + a) = +1. \tag{3}$$

We also consider the problem in Eq. (2) for X, i.e.,

$$\underset{a \in \mathcal{A}(X)}{\text{minimize}} \;\; \sum_{x \in X} c(a \mid x) \quad \text{subject to} \quad f(x+a) = +1, \forall x \in X, \tag{4}$$

where $A(X) = \bigcap_{x \in X} \mathcal{A}(x)$. Then, optimal solutions for the problems in Eq. (3) and (4) are equivalent.

Proof. Let $\mathcal{A}_{x^\circ} = \{a \in \mathcal{A}(x^\circ) \mid f(x^\circ + a) = +1\}$ and $\mathcal{A}_X = \{a \in \mathcal{A}(X) \mid \forall x \in X \colon f(x+a) = +1\}$ be the set of feasible solutions for the problems in Eq. (3) and Eq. (4), respectively. From the definition of c, we have $\sum_{x \in X} c(a \mid x) = C \cdot \|a\|$, where $C := \sum_{x \in X} c_x$. By ignoring constants in the objective function, we find that the problem in Eq. (4) is equivalent to an unconstrained optimization problem: $\min_{a \in \mathcal{A}_X} \|a\|$. Similarly, the problem in Eq. (3) is equivalent to $\min_{a \in \mathcal{A}_{x^\circ}} \|a\|$.

By the definitions of $\mathcal{A}_{x^\circ}$ and $\mathcal{A}_X$, we immediately have $\mathcal{A}_X \subseteq \mathcal{A}_{x^\circ}$. From Lemma 2, we also have $f(x+a) = +1$ for any $x \in X$ and $a \in \mathcal{A}_{x^\circ}$, which implies $\mathcal{A}_{x^\circ} \subseteq \mathcal{A}_X$. By combining the above results, we obtain $\mathcal{A}_{x^\circ} = \mathcal{A}_X$. Since both their feasible solutions and objective functions are respectively equivalent, optimal solutions for the problems in Eq. (3) and (4) are equivalent. □

Now, we give a proof of Proposition 1.

Proof of Proposition 1. From Lemma 3, we have

$$a^* = \min_{a \in \mathcal{A}(x)} c(a \mid x^\circ) \quad \text{subject to} \quad f(x^\circ + a) = +1.$$

By Lemma 1, we also have $c^*(x) = c_x \cdot \frac{|w^\top x|}{\|w\|}$ and $c(a^* \mid x) = c_x \cdot \frac{|w^\top x^\circ|}{\|w\|}$. Using the Cauchy–Schwarz inequality, we obtain

$$\begin{aligned} c(a^* \mid x) - c^*(x) &= c_x \cdot \frac{|w^\top x^\circ|}{\|w\|} - c_x \cdot \frac{|w^\top x|}{\|w\|} \\ &= \frac{c_x}{\|w\|} \cdot (w^\top x - w^\top x^\circ) && (\because w^\top x^\circ \leq w^\top x \leq 0) \\ &= \frac{c_x}{\|w\|} \cdot w^\top (x - x^\circ) \\ &\leq \frac{c_x}{\|w\|} \cdot \|w\| \cdot \|x - x^\circ\| = c_x \cdot \|x - x^\circ\|. \end{aligned}$$

□

A.2 Proof of Proposition 2

Proof of Proposition 2. By the definitions of g_γ, X_1, and X_2, we have

$$g_\gamma(a \mid X) = \sum_{x \in X} i_\gamma(a \mid x) = \sum_{x \in X_1} i_\gamma(a \mid x) + \sum_{x \in X_2} i_\gamma(a \mid x) = g_\gamma(a \mid X_1) + g_\gamma(a \mid X_2).$$

Since $\mathcal{A}(X) \subseteq \mathcal{A}(X_1)$ and $\mathcal{A}(X) \subseteq \mathcal{A}(X_2)$, we also have $a^*_X \in \mathcal{A}(X_1)$ and $a^*_X \in \mathcal{A}(X_2)$. Thus, it holds that $g_\gamma(a^*_X \mid X_1) \geq g_\gamma(a^*_{X_1} \mid X_1)$ and $g_\gamma(a^*_X \mid X_2) \geq g_\gamma(a^*_{X_2} \mid X_2)$, respectively. Therefore, we obtain

$$g_\gamma(a^*_X \mid X) = g_\gamma(a^*_X \mid X_1) + g_\gamma(a^*_X \mid X_2) \geq g_\gamma(a^*_{X_1} \mid X_1) + g_\gamma(a^*_{X_2} \mid X_2).$$

□

A.3 Proof of Theorem 1

To obtain a bound of the optimal leaf size, we first show an upper bound on the optimal value for Problem 1.

Lemma 4. *For a set of N instances $X \subset \mathcal{X}$ such that $f(x) \neq +1$ for any $x \in X$, we have* $\min_{a \in \mathcal{A}(X)} g_\gamma(a \mid X) \leq \gamma \cdot N$.

Proof. Since $f(x) \neq +1$ and $c(\mathbf{0} \mid x) = 0$ for any $x \in X$, we have $g_\gamma(\mathbf{0} \mid X) = \sum_{x \in X} c(\mathbf{0} \mid x) + \gamma \cdot \sum_{x \in X} l(f(x + \mathbf{0}), +1) = \gamma \cdot N$. Since $\mathbf{0} \in \mathcal{A}(X)$, we have $\min_{a \in \mathcal{A}(X)} g_\gamma(a \mid X) \leq g_\gamma(\mathbf{0} \mid X)$. Therefore, we obtain $\min_{a \in \mathcal{A}(X)} g_\gamma(a \mid X) \leq \gamma \cdot N$. □

Using Lemma 4, we give a proof of Theorem 1 as follows.

Proof of Theorem 1. Let h_0 be a CET that returns a single action $a^* = \arg\min_{a \in \mathcal{A}(X)} g_\gamma(a \mid X)$ for any input, i.e., $h_0(x) = a^*$ for any $x \in \mathcal{X}$. Since $a^* \in \mathcal{A}(x)$ for any $x \in X$, we have $h_0 \in \mathcal{H}$ and $o_{\gamma,\lambda}(h^*) \leq o_{\gamma,\lambda}(h_0)$. From Lemma 4, we also have

$$\begin{aligned} o_{\gamma,\lambda}(h_0) &= \frac{1}{N} \sum_{x \in X} i_\gamma(h_0(x) \mid x) + \lambda \cdot |\mathcal{L}(h_0)| \\ &= \frac{1}{N} \sum_{x \in X} i_\gamma(a^* \mid x) + \lambda \cdot 1 \\ &= \frac{1}{N} \min_{a \in \mathcal{A}(X)} g_\gamma(a \mid X) + \lambda \leq \gamma + \lambda. \end{aligned}$$

Since $o_{\gamma,\lambda}(h^*) \geq \lambda \cdot |\mathcal{L}(h^*)|$, we obtain

$$\lambda \cdot |\mathcal{L}(h^*)| \leq o_{\gamma,\lambda}(h^*) \leq o_{\gamma,\lambda}(h_0) \leq \gamma + \lambda \iff |\mathcal{L}(h^*)| \leq \frac{\gamma + \lambda}{\lambda}.$$

□

B MILO FORMULATION FOR GROUP-WISE CE

In this section, we formulate Problem 1 as a mixed-integer linear optimization (MILO) problem. Our formulations can be applied to the cases where we use linear classifiers (e.g., logistic regression), tree ensembles (e.g., gradient boosted trees (Ke et al., 2017)), and deep ReLU networks. We can obtain an optimal solution for the MILO problem by off-the-shelf MILO solvers, such as CPLEX[2], and easily recover the optimal solution for Problem 1 from it. We also present a model-agnostic formulation based on linear approximation by LIME (Ribeiro et al., 2016).

B.1 Common Ideas

As with existing methods based on MILO (Ustun et al., 2019; Kanamori et al., 2020), we assume that each coordinate A_d of a given feasible action set $\mathcal{A}(X) = A_1 \times \cdots \times A_D$ is finite and discretized; that is, we assume $A_d = \{a_{d,1}, \dots, a_{d,I_d}\}$, where $I_d = |A_d|$.

For simplicity, we also assume that the cost function c can be expressed as the following linear form:

$$c(a \mid x) = \sum_{d=1}^{D} c_d(a_d \mid x_d),$$

where $c_d \colon A_d \to \mathbb{R}_{\geq 0}$ is a cost measure of the feature d that represents the effort to change x_d to $x_d + a_d$. It includes several existing cost functions, such as the total-log percentile shift (Ustun et al., 2019) and the weighted ℓ_1-norm based on the inverse of median absolute deviation (Wachter et al., 2018). Note that our formulations can be extended to handle existing non-linear cost functions, such as the max percentile shift (Ustun et al., 2019), the Mahalanobis distance, and local outlier factor (Kanamori et al., 2020).

To express an action $a \in \mathcal{A}(X)$, we introduce binary variables $\pi_{d,i} \in \{0,1\}$ for $d \in [D]$ and $i \in [I_d]$, which indicate that the action $a_{d,i} \in A_d$ is selected ($\pi_{d,i} = 1$) or not ($\pi_{d,i} = 0$). The variables $\pi_{d,i}$ must satisfy $\sum_{i=1}^{I_d} \pi_{d,i} = 1$ for $d \in [D]$. Using $\pi_{d,i}$, each element a_d of an action $a = (a_1, \dots, a_D) \in \mathcal{A}(X)$ can be expressed as

$$a_d = \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}.$$

To formulate the objective function $g_\gamma(a \mid X)$ for a given set of instances $X = \{x^{(1)}, \dots, x^{(N)}\}$, we need to express the cost $c(a \mid x)$ and loss $l(f(x+a), +1)$ for $x \in X$ as linear combinations and constraints of decision variables. For $n \in [N]$, we introduce binary variables $\zeta_n \in \{0,1\}$ such that $\zeta_n = l(f(x^{(n)} + a), +1)$. Then, we can express $g_\gamma(a \mid X)$ as follows:

$$g_\gamma(a \mid X) = \sum_{n=1}^{N} \sum_{d=1}^{D} \sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} + \gamma \cdot \sum_{n=1}^{N} \zeta_n,$$

where $c_{d,i}^{(n)} \geq 0$ is the constant such that $c_{d,i}^{(n)} = c_d(a_d \mid x_d^{(n)})$, which can be computed when $x^{(n)}$ and $\mathcal{A}(X)$ are given. We need to express $\zeta_n = l(f(x^{(n)} + a), +1)$ by linear constraints of decision variables for each type of classifiers f. In the following, we present an MILO formulation of Problem 1 for each classifier.

B.2 Linear Classifier

Let f be a linear classifier $f(x) = \operatorname{sgn}\left(w^\top x + b\right)$, where $w \in \mathbb{R}^D$ is a coefficient vector and $b \in \mathbb{R}$ is an intercept. Then, Problem 1 with the linear classifier f can be formulated as the following MILO problem (Ustun et al.,

[2] https://www.ibm.com/analytics/cplex-optimizer

2019):

$$\begin{aligned}
\text{minimize} \quad & \sum_{n=1}^{N}\sum_{d=1}^{D}\sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} + \gamma \cdot \sum_{n=1}^{N} \zeta_n \\
\text{subject to} \quad & M_n \cdot \zeta_n \leq \sum_{d=1}^{D} w_d \cdot \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i} + F_n, && \forall n \in [N], \\
& \sum_{i=1}^{I_d} \pi_{d,i} = 1, && \forall d \in [D], \\
& \pi_{d,i} \in \{0,1\}, && \forall d \in [D], \forall i \in [I_d], \\
& \zeta_n \in \{0,1\}, && \forall n \in [N],
\end{aligned} \tag{5}$$

where $F_n = w^\top x^{(n)} + b$ and $M_n = \min_{a \in \mathcal{A}(X)} w^\top a + F_n$ and are constants. These values can be computed when f, $x^{(n)}$, and $\mathcal{A}(X)$ are given.

B.3 Tree Ensembles

Let f be a tree ensemble $f(x) = \operatorname{sgn}\left(\sum_{j=1}^{J} w_j \cdot f_j(x)\right)$, where $f_j \colon \mathcal{X} \to \mathbb{R}$ is a decision tree, $w_j \in \mathbb{R}$ is a weight value of the j-th decision tree f_j, and $J \in \mathbb{N}$ is the total number of decision trees. Each decision tree f_j can be expressed as $f_j(x) = \sum_{l=1}^{L_j} \hat{y}_{j,l} \cdot \mathbb{I}\left[x \in r_{j,l}\right]$, where $L_j \in \mathbb{N}$ is the total number of leaves in f_j, and $\hat{y}_{j,l} \in \mathbb{R}$ and $r_{j,l} \subseteq \mathcal{X}$ are the predictive label and the region corresponding to a leaf $l \in [L_j]$, respectively. Then, Problem 1 with the tree ensemble f can be formulated as the following MILO problem (Cui et al., 2015; Kanamori et al., 2020):

$$\begin{aligned}
\text{minimize} \quad & \sum_{n=1}^{N}\sum_{d=1}^{D}\sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} + \gamma \cdot \sum_{n=1}^{N} \zeta_n \\
\text{subject to} \quad & M_n \cdot \zeta_n \leq \sum_{j=1}^{J} w_j \cdot \sum_{l=1}^{L_j} \hat{y}_{j,l} \cdot \phi_{j,l}^{(n)}, && \forall n \in [N], \\
& \sum_{i=1}^{I_d} \pi_{d,i} = 1, && \forall d \in [D], \\
& \sum_{i=1}^{I_d} \phi_{j,l}^{(n)} = 1, && \forall j \in [J], \forall n \in [N], \\
& D \cdot \phi_{j,l}^{(n)} \leq \sum_{d=1}^{D} \sum_{i \in I_{j,l}^{(d)}(x^{(n)})} \pi_{d,i} = 1, && \forall j \in [J], \forall l \in [L_j], \forall n \in [N], \\
& \pi_{d,i} \in \{0,1\}, && \forall d \in [D], \forall i \in [I_d], \\
& \zeta_n \in \{0,1\}, && \forall n \in [N], \\
& \phi_{j,l}^{(n)} \in \{0,1\}, && \forall j \in [J], \forall j \in [L_j], \forall n \in [N],
\end{aligned} \tag{6}$$

where $I_{j,l}^{(d)}(x) = \{i \in [I_d] \mid x_d + a_{d,i} \in r_{j,l}^{(d)}\}$ and $r_{j,l}^{(d)}$ is the subspace with respect to the feature d such that $r_{j,l} = r_{j,l}^{(1)} \times \cdots \times r_{j,l}^{(D)}$. M_n is a constant such that $M_n \leq \min_{a \in \mathcal{A}(X)} \sum_{j=1}^{J} w_j \cdot f_j(x + a)$. From the properties of decision trees, we can set $M_n = \sum_{j=1}^{J} \min_{l \in [L_j]} w_j \cdot \hat{y}_{j,l}$ for any $n \in [N]$, which can be computed when f is given. For example, if f is a random forest, we can set $M_n = -1$ since $w_j = \frac{1}{J}$ and $\hat{y}_{j,l} \in \mathcal{Y}$ for any $j \in [J]$.

B.4 Deep ReLU Networks

For simplicity, we focus on a two-layer ReLU network $f(x) = \operatorname{sgn}\left(\sum_{j=1}^{J} w_j \cdot \max\{0, u_j^\top x + b_j\}\right)$, where $(u_j, b_j) \in \mathbb{R}^{D+1}$ is a pair of a coefficient vector and an intercept of the j-th neuron, $w_j \in \mathbb{R}$ is a weight value of the j-th neuron, and $J \in \mathbb{N}$ is the total number of neurons in the middle layer. Then, Problem 1 with the two-layer ReLU network f can be formulated as the following MILO problem (Serra et al., 2018; Kanamori et al., 2021):

$$\begin{array}{lll} \text{minimize} & \sum_{n=1}^{N} \sum_{d=1}^{D} \sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} + \gamma \cdot \sum_{n=1}^{N} \zeta_n & \\ \text{subject to} & M_n \cdot \zeta_n \leq \sum_{j=1}^{J} w_j \cdot \xi_j^{(n)}, & \forall n \in [N], \\ & \sum_{i=1}^{I_d} \pi_{d,i} = 1, & \forall d \in [D], \\ & \xi_j^{(n)} \leq H_{j,n} \cdot \nu_j^{(n)}, & \forall j \in [J], \forall n \in [N], \\ & \bar{\xi}_j^{(n)} \leq -\bar{H}_{j,n} \cdot (1 - \nu_j^{(n)}), & \forall j \in [J], \forall n \in [N], \\ & \xi_j^{(n)} - \bar{\xi}_j^{(n)} = \sum_{d=1}^{D} u_{j,d} \cdot \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i} + F_{j,n}, & \forall j \in [J], \forall n \in [N], \\ & \pi_{d,i} \in \{0,1\}, & \forall d \in [D], \forall i \in [I_d], \\ & \zeta_n \in \{0,1\}, & \forall n \in [N], \\ & \xi_j^{(n)}, \bar{\xi}_j^{(n)} \geq 0, & \forall j \in [J], \forall n \in [N], \\ & \nu_j^{(n)} \in \{0,1\}, & \forall j \in [J], \forall n \in [N], \end{array} \tag{7}$$

where M_n, $H_{j,n}$, $\bar{H}_{j,n}$, and $F_{j,n}$ are constants such that $M_n \leq \min_{a \in \mathcal{A}(X)} \sum_{j=1}^{J} w_j \cdot \max\{0, f_j(x^{(n)} + a)\}$, $H_{j,n} \geq \max_{a \in \mathcal{A}(X)} f_j(x^{(n)} + a)$, $\hat{H}_{j,n} \leq \min_{a \in \mathcal{A}(X)} f_j(x^{(n)} + a)$, and $F_{j,n} = f_j(x^{(n)})$, where $f_j(x) = u_j^\top x + b_j$. These values can be computed when f, $x^{(n)}$, and $\mathcal{A}(X)$ are given. Note that our formulation can be extended to general multilayer ReLU networks (Serra et al., 2018).

B.5 Group-wise AR-LIME

While we can obtain the optimal solutions for Problem 1 by solving the problems in Eqs. (5), (6), and (7), the total numbers of variables and constraints are respectively $\mathcal{O}(N)$ times larger than that of the individualized CE. In fact, we observed that formulations (6) and (7) were often computationally infeasible in our preliminary experiments on standard datasets even with $N = 10$.

For computational efficiency, we propose an approximation method based on *AR-LIME*, which has been introduced by Rawal and Lakkaraju (2020) in their experiments as a comparison baseline methods. For a given single instance $x \in \mathcal{X}$, AR-LIME approximates a classifier f by a linear model $\hat{f}(x) = w^\top x + b$ in the neighborhood of x like LIME (Ribeiro et al., 2016), and then extracts an optimal action from $\hat{f}$ instead of f. We extend AR-LIME to our setting of group-wise CE by approximating a linear model $\hat{f}_n$ for each of the given instances $x^{(n)} \in X$ and solving eq. (5) for $\hat{f}_1, \dots \hat{f}_N$ instead of f. It can be formulated as the following MILO problem:

$$\begin{array}{lll} \text{minimize} & \sum_{n=1}^{N} \sum_{d=1}^{D} \sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} + \gamma \cdot \sum_{n=1}^{N} \zeta_n & \\ \text{subject to} & M_n \cdot \zeta_n \leq \sum_{d=1}^{D} w_d^{(n)} \cdot \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i} + F_n, & \forall n \in [N], \\ & \sum_{i=1}^{I_d} \pi_{d,i} = 1, & \forall d \in [D], \\ & \pi_{d,i} \in \{0,1\}, & \forall d \in [D], \forall i \in [I_d], \\ & \zeta_n \in \{0,1\}, & \forall n \in [N], \end{array} \tag{8}$$

where $w^{(n)} \in \mathbb{R}^D$ is a parameter of a linear model $\hat{f}_n$ approximated for $x^{(n)}$, and $M_n = \min_{a \in \mathcal{A}(X)} \hat{f}_n(x + a)$ and $F_n = \hat{f}_n(x^{(n)})$ are constants. In our experiments, we used formulation (8) in line 13 of Algorithm 1.

C IMPLEMENTATION DETAILS OF BASELINE METHODS

C.1 Clustering

As a baseline, we implemented a clustering-based method, which has been introduced by Gao et al. (2021) in their experiments as a comparison baseline method. It consists of the following two steps:

1. For a given set of training instances $X \subset \mathcal{X}$, we divide X into 8 distinct subsets by K-means clustering with the default parameters of scikit-learn[3].
2. For each subset $X' \subset X$, we assign an optimal action $a^* \in \mathcal{A}(X)$ by solving an MILO problem in appendix B.

For computational efficiency, we use formulation (8) described in Appendix B.5.

C.2 AReS (Rawal and Lakkaraju, 2020)

Actionable Recourse Summary (AReS) (Rawal and Lakkaraju, 2020) is an interpretable summary of actions expressed by a rule set. It assists users to globally understand the recourse provided by underlying models with emphasises on specific subgroups of interest. We can train an AReS from a given set of instances through submodular optimization. Unlike our CET, an AReS assigns actions to input instances as a form of conditions rather than perturbations vectors. To make a uniform comparison, we slightly adjusted their submodular optimization framework so as to handle standard cost functions of CE and to provide perturbation vectors.

Let $\mathcal{Q}$ be a set of conditions, which can be constructed by a frequent itemset mining algorithm (e.g., FP-growth). A *recourse rule* $q = (q_{\mathrm{pre}}, q_{\mathrm{post}}) \in \mathcal{Q} \times \mathcal{Q}$ is a pair of pre- and post-conditions $q_{\mathrm{pre}}, q_{\mathrm{post}} \colon \mathcal{X} \to \{0, 1\}$. For a set of recourse rules $Q \subset \mathcal{Q} \times \mathcal{Q}$, an AReS $h_Q \colon \mathcal{X} \to \mathcal{A}$ is defined as

$$h_Q(x) = a_{q^*}(x) := \operatorname*{arg\,min}_{a \in \mathcal{A}(x)} \; c(a \mid x) \quad \text{subject to} \;\; q^*_{\mathrm{post}}(x + a) = 1,$$

where $q^* \in Q$ such that $q^*_{\mathrm{pre}}(x) = 1$. That is, for an input instance $x \in \mathcal{X}$, an AReS assigns a perturbation vector $a \in \mathcal{A}(x)$ such that the perturbed instance $x + a$ satisfies the post-condition q^*_{post}, i.e., $q^*_{\mathrm{post}}(x + a) = 1$, where $q^* = (q^*_{\mathrm{pre}}, q^*_{\mathrm{post}})$ is the corresponding recourse rule such that $q^*_{\mathrm{pre}}(x) = 1$. If there are multiple corresponding recourse rules, we choose one with the highest empirical probability $\frac{|X_q|}{|X|}$ among them, where $X \subset \mathcal{X}$ is a given set of training instances and $X_q = \{x' \in X \mid q_{\mathrm{pre}}(x') = 1\}$. Note that there may be no corresponding recourse rule in Q for an input instance $x \in \mathcal{X}$. In our experiments, we also computed a default recourse rule $(q^\circ_{\mathrm{pre}}, q^\circ_{\mathrm{post}})$ such that $q^\circ_{\mathrm{pre}}(x) = 1$ for any $x \in \mathcal{X}$ and q°_{post} is optimized for the instances $\{x \in X \mid \forall q \in Q \colon q_{\mathrm{pre}}(x) = 0\}$.

To learn an AReS h_Q from a given set of training instances X, Rawal and Lakkaraju (2020) proposed several objective functions that measure the quality of an AReS h_Q. We used three of their objective functions: recourse accuracy, coverage, and cost, which are defined as follows:

$$\begin{aligned}
o_{\mathrm{acc}}(Q \mid X) &= |X| \cdot B - \sum_{q \in Q} |\{x \in X \mid q_{\mathrm{pre}}(x) = 1 \land f(x + h_Q(x)) \neq +1\}|, \\
o_{\mathrm{cov}}(Q \mid X) &= \sum_{x \in X} \mathbb{I}\left[\exists q \in Q \colon q_{\mathrm{pre}}(x) = 1\right], \\
o_{\mathrm{cost}}(Q \mid X) &= \sum_{q \in Q} \frac{1}{|X_q|} \sum_{x \in X_q} c(a_q(x) \mid x),
\end{aligned}$$

where $B \in \mathbb{N}$ is a given maximum number of recourse rules. While the recourse accuracy $o_{\mathrm{acc}}(Q \mid X)$ corresponds to our loss $l(f(x+h(x)), +1)$, the cost $o_{\mathrm{cost}}(Q \mid X)$ corresponds to our cost $c(h(x) \mid x)$. Finally, a task of learning an AReS h_Q is formulated as the following non-monotone submodular optimization problem with a cardinality constraint:

$$\begin{aligned}
\underset{Q \subset \mathcal{Q} \times \mathcal{Q}}{\text{maximize}} \quad & \lambda_{\mathrm{acc}} \cdot o_{\mathrm{acc}}(Q \mid X) + \lambda_{\mathrm{cov}} \cdot o_{\mathrm{cov}}(Q \mid X) - \lambda_{\mathrm{cost}} \cdot o_{\mathrm{cost}}(Q \mid X) \\
\text{subject to} \quad & |Q| \leq B,
\end{aligned}$$

[3] `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html`

Table 3: Average values of the total number of actions (**#Actions**), ratio of test instances that had been assigned no action (**Uncover**), and ratio of test instances that had been assigned multiple actions (**Conflict**).

Classifier	**Dataset**	**#Actions**		**Uncover**		**Conflict**	
		AReS	**CET**	**AReS**	**CET**	**AReS**	**CET**
LightGBM	Attrition	8.0	7.1	13.32%	0.0%	30.70%	0.0%
	German	8.0	6.0	51.97%	0.0%	21.33%	0.0%
TabNet	Attrition	8.0	6.7	13.28%	0.0%	23.49%	0.0%
	German	8.0	5.7	4.89%	0.0%	33.10%	0.0%
Average		8.0	**6.38**	20.87%	**0.0**%	27.15%	**0.0**%

Table 4: Average computational time on the Attrition and German datasets.

Classifier	**Dataset**	**Clustering**	**AReS**	**CET**
LightGBM	Attrition	24.023 ± 3.77	1268.14 ± 107.41	8021.79 ± 2600.78
	German	19.865 ± 1.32	18767.0 ± 538.51	5782.37 ± 901.46
TabNet	Attrition	2036.75 ± 595.11	4118.18 ± 375.96	27009.4 ± 5891.93
	German	4258.59 ± 1012.93	36242.8 ± 2844.69	9969.62 ± 2463.63
Average		1584.81	15099.0	12695.8

where λ_{acc}, λ_{cov}, and λ_{cost} are given trade-off parameters. In our experiments, we solve the above optimization problem by a standard greedy algorithm. We set $B = 8$, and chose each trade-off parameter from $\{0.01, 0.1, 1.0, 10.0, 100.0\}$ by the grid search so as to improve our invalidity score $i_\gamma(h_Q(x) \mid x)$ on hold-out datasets.

D ADDITIONAL EXPERIMENTAL RESULTS

D.1 Performance Comparison

We present additional experimental results of performance comparison (Section 5.2) in Tables 3 and 4. Table 3 presents average values of the total number of actions (**#Actions**), ratio of test instances that had been assigned no action (**Uncover**), and ratio of test instances that had been assigned multiple actions (**Conflict**) for **AReS** and **CET**. Note that #Actions of **AReS** was always 8.0 because the maximum number of recourse rules, i.e., actions, was set to $B = 8$, and that Uncover and Conflict of **CET** were 0.0% for all the cases due to the properties of decision trees. The results of computational time for each method are shown in Table 4.

D.2 Analysis of Trade-off Between Effectiveness and Interpretability

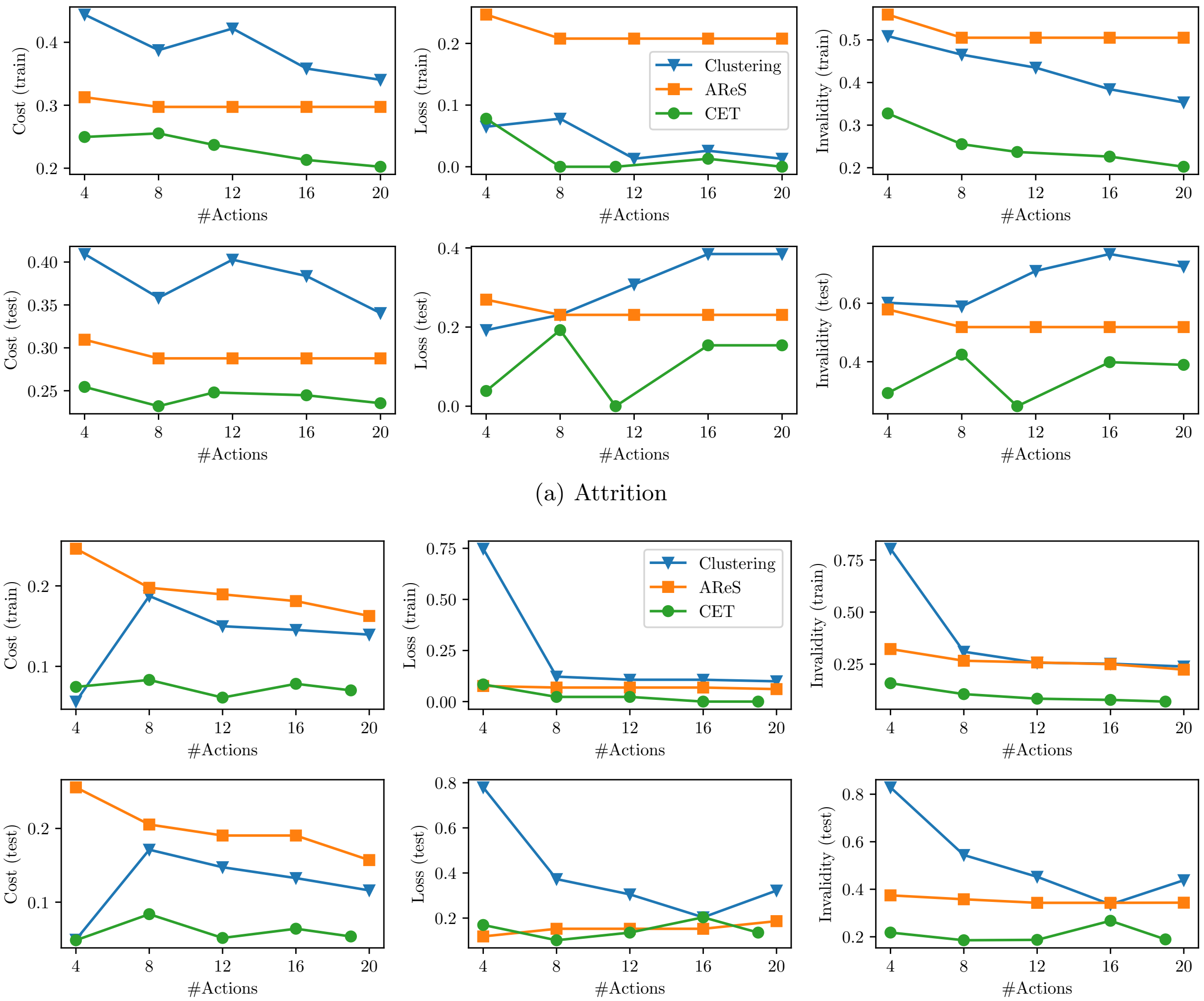
Figure 5 presents the complete results of the trade-off analyses between the effectiveness of assigned actions and interpretability of each method in Section 5.2. We measured the average values of the cost, loss, and invalidity on both train and test datasets.

D.3 Sensitivity Analysis of Problem 1

The results of sensitivity analyses with respect to the trade-off parameter γ in Problem 1 are shown in Figure 6. We randomly sampled 10 instances from test instances, and solving Problem 1 for them by varying the value of γ. We repeated this procedure 100 times, and report the average values of the cost and loss.

D.4 Convergence Analysis of Algorithm 1

The results of convergence analyses of Algorithm 1 are shown in Figure 7. We measured the objective values $o_{\gamma,\lambda}$ of t-th solution $h^{(t)}$ and best solution h^* during running Algorithm 1.



(a) Attrition

(b) German

Figure 5: Analyses of trade-off between effectiveness and interpretability.

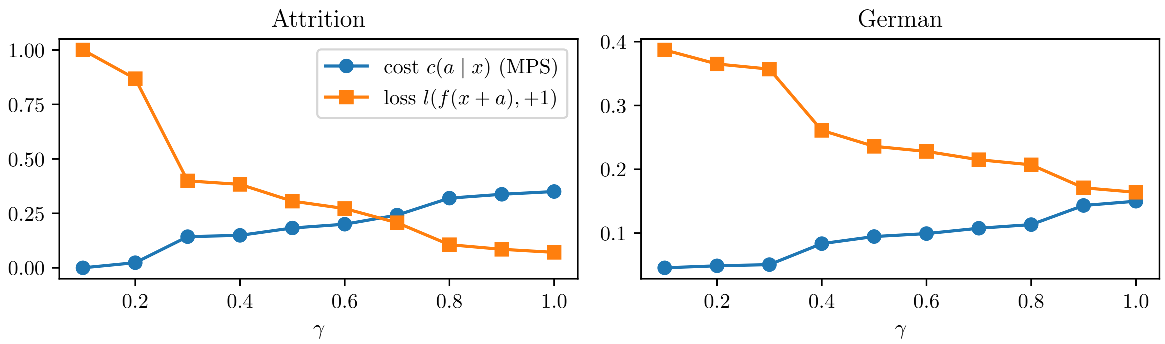
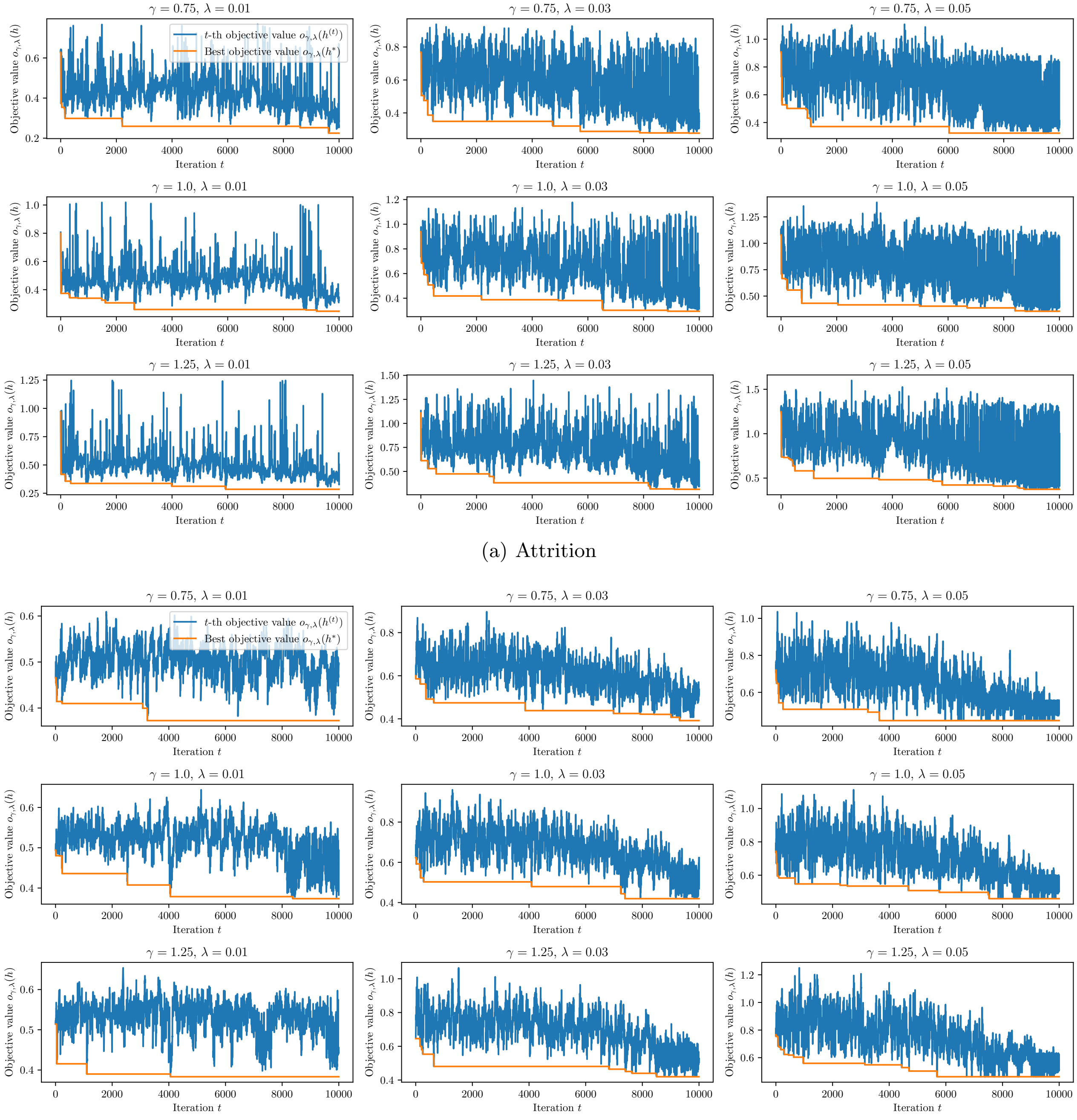


Figure 6: Sensitivity analyses of the trade-off parameter γ.



(a) Attrition

(b) German

Figure 7: Convergence analyses of Algorithm 1.

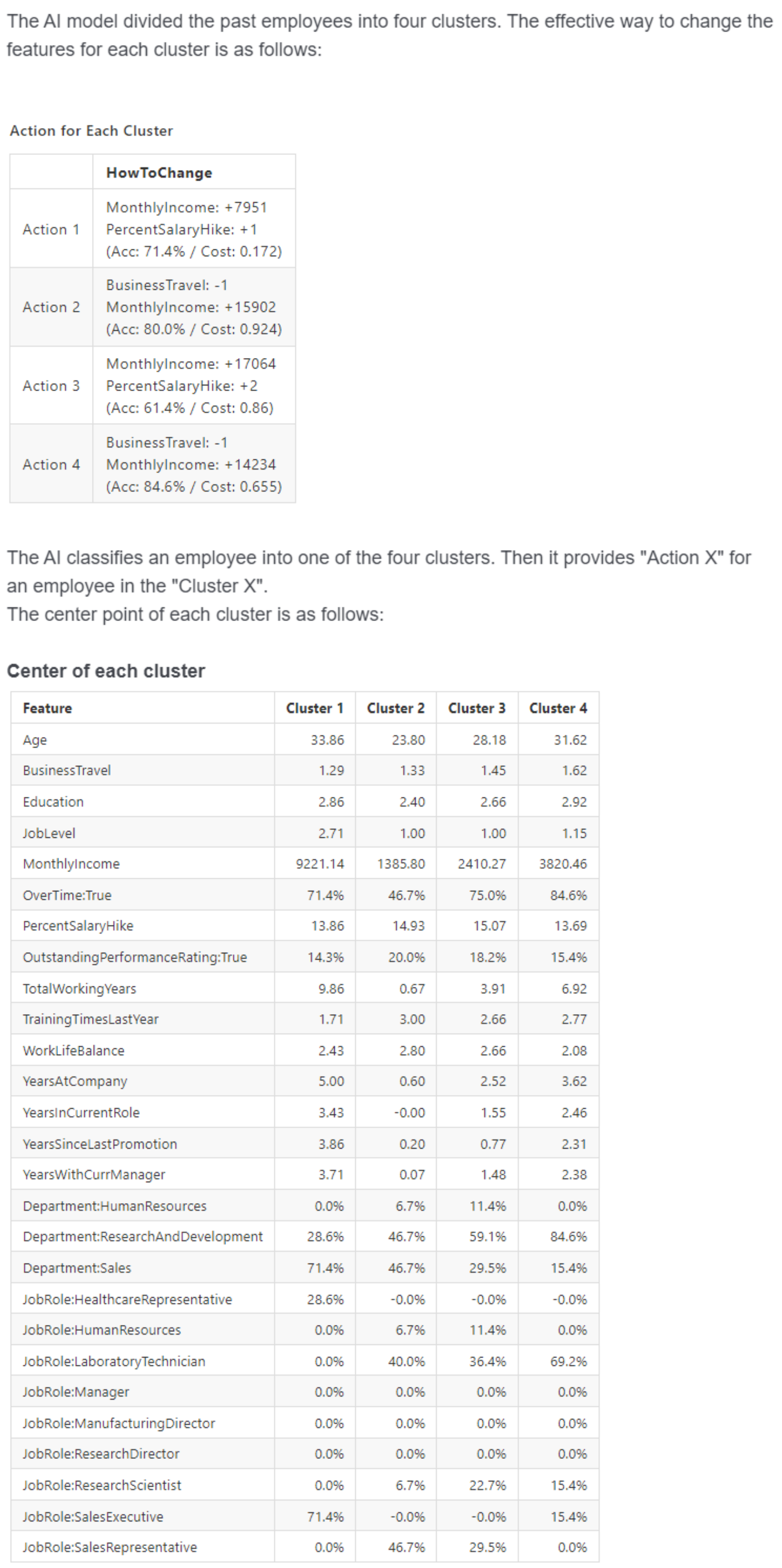
Table 5: IQR of the time to answer in the user study.

Method	Q_1	median	Q_3
Clustering	901.25	1094.5	1491
AReS	660.75	730.	1049.75
CET	426.	524.	997.25

E DETAILS OF USER STUDY

An example of each method and questions of our user study (Section 5.3) is shown in Figures 8 to 11. The participants see the explanation of the assigned method; the action and center of each cluster for **Clustering** (Figure 8), the rule set for **AReS** (Figure 9), and the decision tree for **CET** (Figure 10). In addition, they answered the questions displayed like in Figure 11. Each method and questions were displayed on a single page. Therefore, the participants were able to answer the questions with referring to the method.

We have attached as supplementary material a spreadsheet that summarizes the results of all participants' answers. Note that we excluded the participants whose answer time exceeded one hour or who did not answer all the questions when we calculated the average time. We show IQR of the answer time for each method in Table 5.



The AI model divided the past employees into four clusters. The effective way to change the features for each cluster is as follows:

Action for Each Cluster

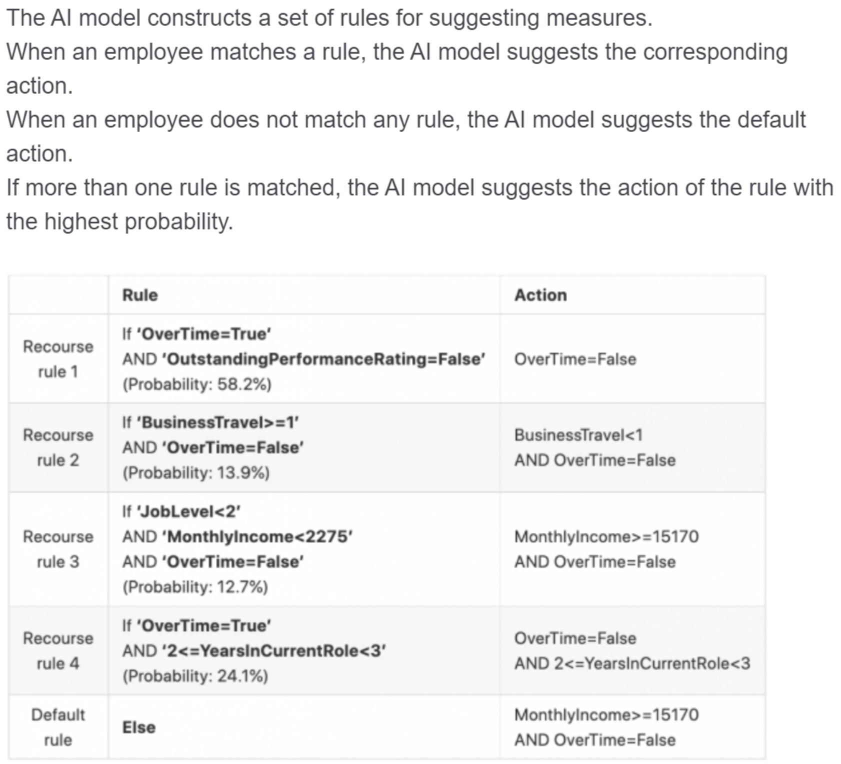
	HowToChange
Action 1	MonthlyIncome: +7951 PercentSalaryHike: +1 (Acc: 71.4% / Cost: 0.172)
Action 2	BusinessTravel: -1 MonthlyIncome: +15902 (Acc: 80.0% / Cost: 0.924)
Action 3	MonthlyIncome: +17064 PercentSalaryHike: +2 (Acc: 61.4% / Cost: 0.86)
Action 4	BusinessTravel: -1 MonthlyIncome: +14234 (Acc: 84.6% / Cost: 0.655)

The AI classifies an employee into one of the four clusters. Then it provides "Action X" for an employee in the "Cluster X".
The center point of each cluster is as follows:

Center of each cluster

Feature	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Age	33.86	23.80	28.18	31.62
BusinessTravel	1.29	1.33	1.45	1.62
Education	2.86	2.40	2.66	2.92
JobLevel	2.71	1.00	1.00	1.15
MonthlyIncome	9221.14	1385.80	2410.27	3820.46
OverTime:True	71.4%	46.7%	75.0%	84.6%
PercentSalaryHike	13.86	14.93	15.07	13.69
OutstandingPerformanceRating:True	14.3%	20.0%	18.2%	15.4%
TotalWorkingYears	9.86	0.67	3.91	6.92
TrainingTimesLastYear	1.71	3.00	2.66	2.77
WorkLifeBalance	2.43	2.80	2.66	2.08
YearsAtCompany	5.00	0.60	2.52	3.62
YearsInCurrentRole	3.43	-0.00	1.55	2.46
YearsSinceLastPromotion	3.86	0.20	0.77	2.31
YearsWithCurrManager	3.71	0.07	1.48	2.38
Department:HumanResources	0.0%	6.7%	11.4%	0.0%
Department:ResearchAndDevelopment	28.6%	46.7%	59.1%	84.6%
Department:Sales	71.4%	46.7%	29.5%	15.4%
JobRole:HealthcareRepresentative	28.6%	-0.0%	-0.0%	-0.0%
JobRole:HumanResources	0.0%	6.7%	11.4%	0.0%
JobRole:LaboratoryTechnician	0.0%	40.0%	36.4%	69.2%
JobRole:Manager	0.0%	0.0%	0.0%	0.0%
JobRole:ManufacturingDirector	0.0%	0.0%	0.0%	0.0%
JobRole:ResearchDirector	0.0%	0.0%	0.0%	0.0%
JobRole:ResearchScientist	0.0%	6.7%	22.7%	15.4%
JobRole:SalesExecutive	71.4%	-0.0%	-0.0%	15.4%
JobRole:SalesRepresentative	0.0%	46.7%	29.5%	0.0%

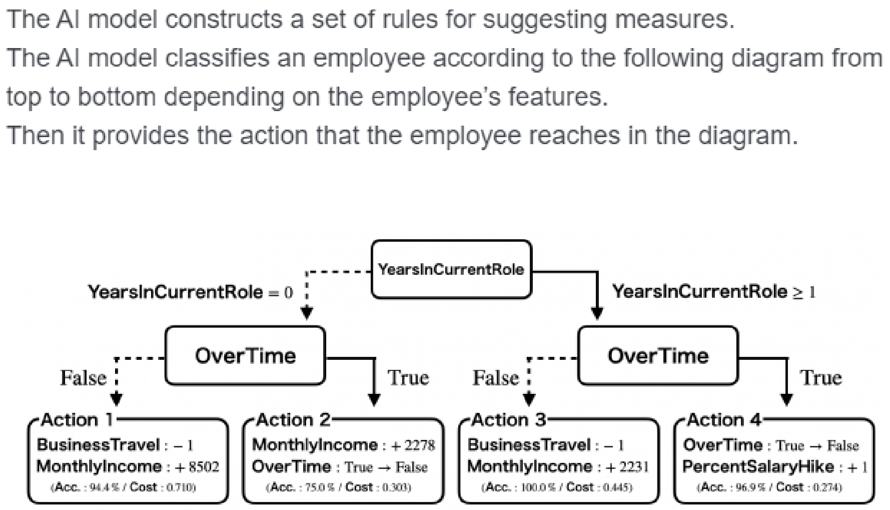
Figure 8: Example of **Clustering** presented in the user study.



The AI model constructs a set of rules for suggesting measures.
When an employee matches a rule, the AI model suggests the corresponding action.
When an employee does not match any rule, the AI model suggests the default action.
If more than one rule is matched, the AI model suggests the action of the rule with the highest probability.

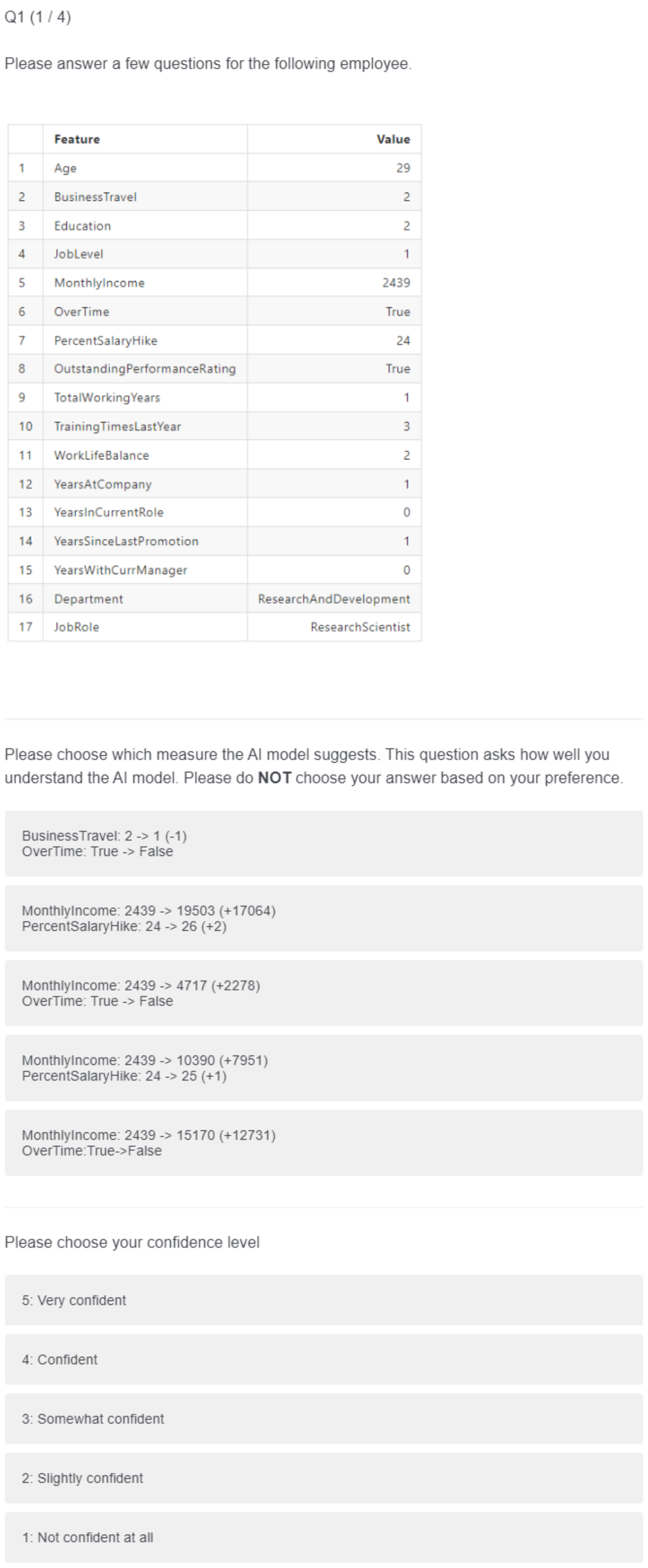
	Rule	Action
Recourse rule 1	If **'OverTime=True'** AND **'OutstandingPerformanceRating=False'** (Probability: 58.2%)	OverTime=False
Recourse rule 2	If **'BusinessTravel>=1'** AND **'OverTime=False'** (Probability: 13.9%)	BusinessTravel<1 AND OverTime=False
Recourse rule 3	If **'JobLevel<2'** AND **'MonthlyIncome<2275'** AND **'OverTime=False'** (Probability: 12.7%)	MonthlyIncome>=15170 AND OverTime=False
Recourse rule 4	If **'OverTime=True'** AND **'2<=YearsInCurrentRole<3'** (Probability: 24.1%)	OverTime=False AND 2<=YearsInCurrentRole<3
Default rule	**Else**	MonthlyIncome>=15170 AND OverTime=False

Figure 9: Example of **AReS** presented in the user study.



The AI model constructs a set of rules for suggesting measures.
The AI model classifies an employee according to the following diagram from top to bottom depending on the employee's features.
Then it provides the action that the employee reaches in the diagram.

Figure 10: Example of **CET** presented in the user study.



Q1 (1 / 4)

Please answer a few questions for the following employee.

	Feature	Value
1	Age	29
2	BusinessTravel	2
3	Education	2
4	JobLevel	1
5	MonthlyIncome	2439
6	OverTime	True
7	PercentSalaryHike	24
8	OutstandingPerformanceRating	True
9	TotalWorkingYears	1
10	TrainingTimesLastYear	3
11	WorkLifeBalance	2
12	YearsAtCompany	1
13	YearsInCurrentRole	0
14	YearsSinceLastPromotion	1
15	YearsWithCurrManager	0
16	Department	ResearchAndDevelopment
17	JobRole	ResearchScientist

Please choose which measure the AI model suggests. This question asks how well you understand the AI model. Please do **NOT** choose your answer based on your preference.

BusinessTravel: 2 -> 1 (-1)
OverTime: True -> False

MonthlyIncome: 2439 -> 19503 (+17064)
PercentSalaryHike: 24 -> 26 (+2)

MonthlyIncome: 2439 -> 4717 (+2278)
OverTime: True -> False

MonthlyIncome: 2439 -> 10390 (+7951)
PercentSalaryHike: 24 -> 25 (+1)

MonthlyIncome: 2439 -> 15170 (+12731)
OverTime:True->False

Please choose your confidence level

5: Very confident

4: Confident

3: Somewhat confident

2: Slightly confident

1: Not confident at all

Figure 11: Example of our survey questions.