T-COL: Generating Counterfactual Explanations for General User Preferences on Variable Machine Learning Systems

MING WANG, Northeastern University, China
DALING WANG*, Northeastern University, China
WENFANG WU, Northeastern University, China and University of Göttingen, Germany
SHI FENG, Northeastern University, China
YIFEI ZHANG, Northeastern University, China

To address the interpretability challenge in machine learning (ML) systems, counterfactual explanations (CEs) have emerged as a promising solution. CEs are unique as they provide workable suggestions to users, instead of explaining why a certain outcome was predicted. The application of CEs encounters two main challenges: general user preferences and variable ML systems. On one hand, user preferences for specific values can vary depending on the task and scenario. On the other hand, the ML systems for verification may change while the CEs are performed. Thus, user preferences tend to be general rather than specific, and CEs need to be adaptable to variable ML models while maintaining robustness even as these models change. Facing these challenges, we propose general user preferences based on insights from psychology and behavioral science, and add the challenge of non-static ML systems as one preference. Moreover, we introduce a novel method, Tree-based Conditions Optional Links (T-COL) for generating CEs adaptable to general user preferences. Moreover, we employ T-COL to enhance the robustness of CEs with specific conditions, making CEs robust even when the ML models are replaced. To assess subjectivity preferences, we define LLM-based autonomous agents to simulate users and align them with real users. Experiments show that T-COL outperforms all baselines in adapting to general user preferences.

JAIR Track: Fairness and Bias in AI

JAIR Associate Editor: Andrea Aler Tubella

JAIR Reference Format:
Ming Wang, Daling Wang, Wenfang Wu, Shi Feng, and Yifei Zhang. 2026. T-COL: Generating Counterfactual Explanations for General User Preferences on Variable Machine Learning Systems. *Journal of Artificial Intelligence Research* 85, Article 5 (January 2026), 24 pages. DOI: 10.1613/jair.1.18166

1 Introduction

Counterfactual explanations (CEs) offer a unique solution to address the interpretability limitations found in widely used and well-performing machine learning (ML) systems. Their implications extend to ML interpretability (Barredo Arrieta et al. 2020; Gunning 2017; Gunning et al. 2019; Molnar 2020) and AI security (Abid et al. 2022; Le et al. 2022; Sokol and Flach 2019), which are effective in increasing user trust in ML systems (Del Ser et al. 2024;

*Corresponding Author.

Authors' Contact Information: Ming Wang, ORCID: 0000-0001-8406-5677, sci.m.wang@gmail.com, Northeastern University, Shenyang, Liaoning, China; Daling Wang, ORCID: 0000-0003-1340-0778, wangdaling@cse.neu.edu.cn, Northeastern University, Shenyang, Liaoning, China; Wenfang Wu, ORCID: 0000-0002-7215-563X, wenfang@stumail.neu.edu.cn, Northeastern University, Shenyang, Liaoning, China and University of Göttingen, Göttingen, Niedersachsen, Germany; Shi Feng, ORCID: 0000-0002-2846-7652, fengshi@cse.neu.edu.cn, Northeastern University, Shenyang, Liaoning, China; Yifei Zhang, ORCID: 0000-0003-0854-2966, zhangyifei@cse.neu.edu.cn, Northeastern University, Shenyang, Liaoning, China.

DOI: 10.1613/jair.1.18166

Metsch et al. 2024). First proposed by Wachter et al. (2018), CEs aim to make minimal changes to the original data points, named as **query sample**, to achieve a desired classification outcome. It can also be considered as a new data sample with a desired category based on the derivation of the query sample. For example, a loan decision ML system might reject a loan request from a user with a profile such as {age: 24, education: bachelor, job: **service**, institution: private}. Conventional methods might state that "the rejection of your loan request is attributed to your service-oriented job", whereas a CE would give a counterfactual like {age: 24, education: bachelor, job: **profession**, institution: private} and can be interpreted as "having a professional job will result in loan approval". This example illustrates that CEs reveal the specific differences in an instance that can lead to a desired outcome. For its importance and uniqueness, many researchers have worked on the CE generation (Artelt and Hammer 2019; Guidotti 2022; Sokol and Flach 2019; Stepin et al. 2021; Verma, Boonsanong, et al. 2022). CEs are also used in various ML-based systems (Albini et al. 2022; Cito et al. 2022; Piccione et al. 2022; Shang et al. 2022; Smith et al. 2022; Wellawatte et al. 2022; Yacoby et al. 2022) and have garnered considerable attention within various ML communities (Dai et al. 2022; Filandrianos et al. 2022; Guidotti 2022; Guidotti, Monreale, et al. 2019; Kuhl et al. 2022; Pawelczyk et al. 2022; Tesic and Hahn 2022; Warren et al. 2022).

However, the further application of CE still faces many challenges. The possibility of CEs is perceived as manipulative from a security view, as discussed in (Slack et al. 2021). The robustness of the CEs is analyzed in (Artelt, Vaquet, et al. 2021b) and (Virgolin and Fracaros 2022). Verma, Dickerson, et al. (2021) summarize and list twelve key challenges for CEs in practical applications and industrial deployments. Inspired by it, we specifically address two of these challenges: challenge 3 of **non-static ML models** and challenge 7 of **capturing personal preferences**. On the one hand, users' preferences need to be captured by CE generation methods. Some researchers (Rasouli and Chieh Yu 2022) have attempted to incorporate user preferences into CEs by adding constraints on feature values. Alternatively, another approach (Cheng et al. 2021) built an interactive interface enabling users to select and set the range of feature values or tendencies for the features. Nevertheless, such specific user preferences for feature values can only reflect the user's tendency for a single task, making it difficult to adapt to complex tasks in real applications. For example, when applying for a loan, a person might tend to report a higher income, whereas when taxed, they might do the opposite. To address this issue, general preferences need to be captured. On the other hand, a common assumption in CE research is that "the validation models for sample classification are fixed and do not change over time", despite the frequent upgrades and changes observed in real-world ML systems. The method in (Hamman et al. 2023) ensures the robustness of CE in variable ML systems by allowing arbitrary changes in the model parameter space while restricting predictive changes for points on the data manifold, and proposes *stability* as a metric to measure this property. PROPLACE obtains an optimal solution that remains valid under all possible parameter variations, even in the presence of uncertainty in model parameters (Jiang et al. 2024). However, these methods only consider changes in model parameters and do not fully account for model replacement. To research these two challenges in a unified way, we consider the robustness of CEs on variable ML systems as a general preference. In other words, it aspires to CEs with the highest possible success probability.

We investigate research in the fields of psychology and behavioral science (Benartzi 2017; Bhatt 2019; Y. Chen et al. 2021), and summarize the behavioral patterns of user preferences. With the help of psychologists, we categorize these patterns into 5 categories of general user preferences. They are abstract tendencies that are independent of specific tasks and reflect psychological traits of users. To capture general user preferences, we propose a novel method called Tree-based Conditions Optional Links (T-COL) to generate CEs that capture general user preferences. For a query sample, T-COL initially selects prototype cases based on user preference to construct a set of trees indicating the combinations of feature values derived from both the query sample and the prototype cases. Each tree contains a portion of the feature values. Then, T-COL selects the optimal combinations of local feature values based on an objective function designed for the user's preferences. The final CE is obtained by concatenating these local feature values and verification. Two components in T-COL are designed to capture

user preferences: the prototype case screening and the local optimization objective. We denote a specific union of these two components as a **condition** of T-COL. We devised a separate condition for each preference, including one oriented to the issue of variable ML systems. Notably, these conditions are flexible and scalable. Simply by designing new conditions, T-COL can be adapted to new preferences.

To evaluate the adaptability of CEs generated by T-COL to general user preferences, we design a large number of Large Language Models (LLMs)-based agents to simulate user experiments with characterization prompt (Wang et al. 2024). By analyzing the behavioral consistency between LLM-based agents and real users in user research, we screened user-simulated agents (US-Agents). Guided by prompt engineering, US-Agents with different profiles are asked to choose the one that better matches their preferences among the CEs generated by different methods. To facilitate US-Agents to make choices, we evaluate the properties of CEs, such as *proximity*, *sparsity*, and *validity*, as reference. Furthermore, we introduce *centrality* and *data fidelity* metrics to accommodate evaluations geared toward general user preferences.

In summary, our work has several primary contributions:

- We propose general user preferences and an instance-based (IB) method, T-COL, with optional components to capture these preferences.
- We leverage user research findings to select and configure US-Agents. These agents then run simulated experiments to assess how well CEs adapt to user preferences.
- Extensive experiments show the strength of the CEs generated by T-COL in adaptation to variable ML systems and general user preferences.

2 Related Work

In this section we introduce the generation methods and evaluation metrics in the research of CE.

2.1 Counterfactual Explanation Generation Approaches

The CE generation approaches can be divided into two main categories: Optimization (OPT) and Heuristic Search Strategy (HSS) (Guidotti 2022). Among these, HSS can be further classified into instance-based (IB) and Decision Tree (DT)-based methods.

OPT approaches generate CEs by perturbing the original data points to cross the decision boundary and be classified as the desired class. There are model-agnostic optimization methods and model-specific computational methods designed for ML models (Artelt and Hammer 2019). Some researchers model the generation of CEs as optimization problems with this idea and set different constraints for the objective (Karimi et al. 2022). For example, Wachter et al. (2018) introduced the concept of CE and constructed an optimization objective based on the distance between the counterfactual and the query sample and the prediction of the counterfactual made by the decision model, which was optimized by Adam (Kingma and Ba 2017). There are also researchers who have formalized most of the properties introduced in previous work on CE as different constraints on the optimization objective and solved for the CE using constrained optimization learning (Maragno et al. 2022). Optimization objectives can be solved using methods such as integer coding (Ustun et al. 2019) and gradient descent (Mothilal et al. 2020). In addition, the solution of the CE problem has been formalized as an optimization problem for a non-monotonic submodular function in (Tsirtsis and Gomez Rodriguez 2020), which can be solved by randomization methods. Guidotti and Ruggieri (2021) proposed an ensemble CE method that combines multiple weak counterfactual explainers to comprehensively consider all desired properties of CEs.

The methods used to solve optimization problems often operate on continuous values, leading to CEs that include unrealistic or understandable but not generally perceived features (Guidotti 2022), such as half a master's degree getting a loan application or a person being 35.8 years old (around 35 years and 10 months, is an understandable feature value but not generally stated as such) having a high income. As a consequence, HSS

methods have been developed to generate CEs based on features selected from the screened sample set or the feature space composed of all data samples. HSS methods involve solving for perturbations around query samples in the data space, where perturbations can be learned in the discrete data space by calculating diversity-enforcing losses (Rodríguez et al. 2021) or finding the sample points most relevant to the target CE from the data space to construct counterfactual pairs, each counterfactual pair consists of a query sample and a selected target sample, and deriving the set of CEs from the counterfactual pairs by an iterative approach (Tran et al. 2021). Silva et al. (2021) generate CEs by sampling an uncertainty model to obtain a range of possible outcome states and selecting counterfactual states using a decision tree-like approach. Keane and Smyth (2020) propose an IB method to construct counterfactual pairs using good cases in the target category and use the different feature values in the counterfactual pairs as the corresponding feature values to compose CEs. Another method is diverse CEs by reusing the k-nearest neighbor case pairs (Smyth and Keane 2022), and Goyal et al. (2019) generated CEs of the bird figures by replacing features in the query samples at the corresponding location with features in the target category samples. In an extension of the IB approach, continuous features are transformed into categorical alternatives in (Warren et al. 2022) to generate more effective CEs from the psychological perspective. For tree-structured classifiers, positive and negative paths are defined based on Boolean tests at internal nodes in Tolomei et al. (2017). By adjusting the feature vector, the classifier ensures that a decision tree predicting a negative outcome satisfies the Boolean conditions along the positive path.

Based on the characteristics of these two types of approaches, we adopt the IB method of HSS to design the CE generation method. Compared to the OPT-based approach, it can better ensure the feasibility and plausibility of CEs.

2.2 Attributes and Evaluation Metrics

How to evaluate the quality or explanatory effect of CEs has also received a great deal of attention from researchers. The properties and evaluation metrics of CEs, as well as the properties of interpreters, have been proposed.

Verma, Boonsanong, et al. (2022) formally and systematically presented the properties of CEs, containing metrics such as validity, actionability, and sparsity, providing metrics for research in related fields and standards for the application of CEs. Several properties on CE are expressed formally in (Maragno et al. 2022) and defined in the form of formulas as constraints in the process of generating CEs. Guidotti (2022) provided a very comprehensive summary of properties, presenting the properties associated with CE in terms of both CE and interpreters, respectively.

In addition, some researchers have also assessed CEs and interpreters from perspectives other than their properties. In order to improve the fairness of CEs, it is proposed in (Artelt, Vaquet, et al. 2021a) that the robustness of CEs must first be improved, and plausible CEs are defined to achieve this. Laugel et al. (2019) also discuss the research work on CE from a robustness perspective. Keane, Kenny, et al. (2021) presented an analysis from a psychological and computational perspective, proposing requirements for distributionally faithful and instance-guided CEs.

Building on these studies on properties and evaluation metrics, Verma, Dickerson, et al. (2021) present 12 challenges for CE of future developments. However, there is currently little research on challenge 3 of **non-static ML systems** and challenge 7 of **capturing personal preferences**. For the two challenges, we propose to generate CEs under the conditions of general user preferences and variable ML models, and corresponding metrics to evaluate them.

3 Problem Description

Existing works (Cheng et al. 2021; Rasouli and Chieh Yu 2022) only take the user's requirements for specific feature values into account, without a general consideration of the user's individual personality and preferences.

However, in practical applications and deployments, users may not be able to express their requirements for specific features with certainty, but rather express their preference for a certain practice or a certain type of CEs. For example, users may be more likely to say "I want an easy way to get approved for a loan" rather than "I want the type of work to be professional or managerial". To achieve this, we define general user preferences.

3.1 General User Preferences

Drawing on research in psychology and behavioral science (Benartzi 2017; Bhatt 2019; Y. Chen et al. 2021; Lee et al. 2024), we propose the following general user preferences:

A **Dedicated Preference**: I prefer to focus on a few things.
B **Minimalist Preference**: I wish it is easier to do.
C **Cautious Preference**: I want the solution with the highest success rate.
D **Admirer Preference**: I hope there are similar successful cases.
E **Collectivist Preference**: I hope the solution aligns with most successful cases.

General user preferences are not constraints on certain feature values and they are related to the user's personality and behavioral habits. As a result, these preferences do not usually change with tasks or scenarios.

3.2 Problem Formulation

To ensure *feasibility*, we follow the IB approach for generating CEs. We selected feature values from the query sample and prototype cases to form new samples that meet the preference requirements, which serve as CEs. Let x denote a query sample, and $\hat{x}$ denote an arbitrary combination of features. Then the CEs denoted as $\tilde{x}$ is the solution to (1).

$$\tilde{x} = \arg\min_{\hat{x}}(|\,\mathbf{M}(\hat{x}) - \tilde{y}| + \mathbf{d}(\hat{x}, x)) \tag{1}$$

where $\tilde{y}$ is the target class, $\mathbf{M}$ denotes the ML model, and $\mathbf{M}(\cdot)$ denotes the classification result of the model on the sample. In addition, $\mathbf{d}$ denotes the distance between the counterfactual and the query sample, which can be calculated using any distance calculation function, such as Euclidean distance, Manhattan distance, Utility Space distance (Främling 2022), etc. The first component represents the absolute value of the error between the classification result of the ML model $\mathbf{M}$ for the current combination of features and the desired category. Reducing this value forces the solved CEs to reach the target category. The second component represents the distance between the current feature combination and the query sample, motivating the CEs generated to be closer to the query sample. The two constraints in (1) are the most essential ones, to which more external constraints can be added depending on the different requirements for the CEs.

Within IB CE explainers, the selection of feature values from different cases, which are selected samples in the target category, provides better assurance and control over the properties of the CEs and has therefore received more attention than the selection of feature values from all samples. In addition to the selection of feature values, the IB approach requires the screening of good cases as Keane and Smyth (2020) mentioned in the target category, to provide their feature values as the candidate set of feature values.

Based on the above analysis, we formalize the generation of CEs into the following three processes:

- **Prototype case screening**: Select good samples from all samples in the target category based on the requirements for CEs as prototype cases, providing the candidate set of feature values to choose from when generating CEs.
- **Feature value selection**: Select a set or combination of sets of feature values from the screened cases and the query sample.

- **Feature value concatenating**: The selected feature values are concatenated in order while ensuring feature consistency, and CEs are obtained after validation.

Of the three processes listed above, **prototype case screening** and **feature value selection** have high correlations with the properties of CEs, which can be used to capture general user preferences with corresponding conditions.

4 Methodology

T-COL captures general user preferences by selecting suitable conditions and generates CEs that match user preferences using processes such as prototype case screening and feature value selection. Additionally, to improve the efficiency of the explainer, we employ a partitioning strategy. T-COL first constructs **local greedy trees** using subsets of the divided local feature values to represent arbitrary local combinations of feature values. The feature values are derived from the query sample and prototype cases selected based on preferences. In addition, locally optimal objectives in conditions of feature values are selected on the **local greedy tree** according to general user preferences. Afterward, the feature value selection paths are obtained from **local greedy trees** and concatenated into a complete feature value selection path for selecting the feature values of CEs.

4.1 General User Preference Analysis

General user preferences are more like the personalities of the users rather than just requirements for a particular feature in a task. However, such general user preferences are abstract and can not simply be represented by feature values. They thus need to be linked to the properties of the CEs so that they can be captured in the generation of CEs. In other words, general user preferences can not be achieved directly by imposing constraints on the generation of CEs. Therefore, we first analyze the connection between general user preferences and the properties of CEs, as shown in (2).

$$mappings = \begin{cases} A : \{sparsity\}, \\ B : \{proximity\}, \\ C : \{actionability,\ coherence,\ validity\}, \\ D : \{data\ manifold\ closeness\}, \\ E : \{diversity,\ data\ manifold\ closeness\} \end{cases} \tag{2}$$

Dedicated preference A requires as few different features as possible between CEs and the query sample so that the user can focus on changing a small number of feature values, requiring a higher degree of *sparsity* in the generated CEs. Minimalist preference B requires CEs to be easier to achieve, which intuitively means that the distance between CEs and the query sample should be as close as possible, placing a higher demand on the *proximity* of CEs. Cautious preference C indicates that the user wants to choose CEs that provide more secure solutions to achieve their needs, which requires a higher level of *validity* of CEs. It is also necessary to ensure that the CEs have a higher degree of feasibility, validity, and consistency. Admirer preference D indicates that the user wants similar success cases, which require CEs to be located in the feature space made up of existing data and to be as close as possible to the existing data, i.e., higher *data manifold closeness*. Collectivist preference E indicates that the user wishes to take the option chosen by the majority, i.e., CEs need to be representative and located in the feature space where the sample points are concentrated. Preference E requires higher *diversity* and *data manifold closeness*.

4.2 Overall Design of T-COL

To capture general user preferences, we introduce T-COL, including the prototype case screening and the feature value selection as optional conditions. In order to improve the efficiency of T-COL, we use a partitioning strategy in the feature value selection process. We first screen prototype cases according to user preferences and construct *local greedy trees* representing the local combination of feature values. Next, the local optimal objective is set according to the user's preference, and the optimal paths of local feature values are selected by using local greedy trees. Finally, all the paths selected by local greedy trees are concatenated in order. The complete path can guide the selection of feature values to compose counterfactuals from the query sample or prototype cases. The overall process is shown in Figure 1.

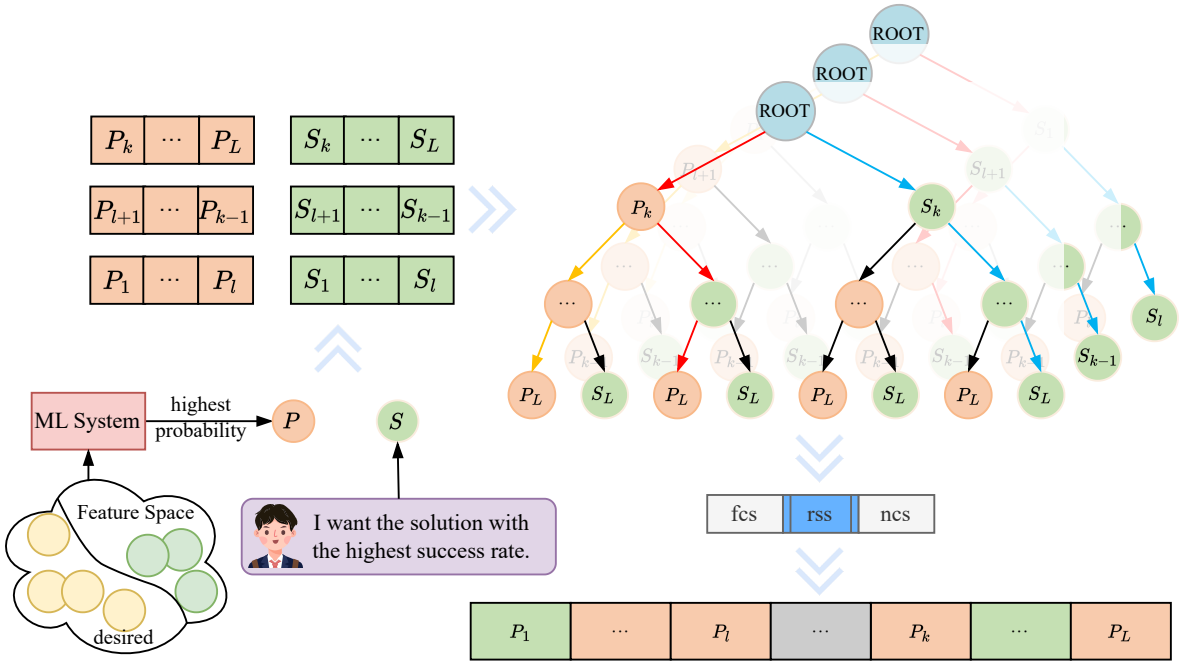

Fig. 1. The whole process of T-COL, with the triangular arrow indicating the flow between the several processes. The user wants a more robust CE, and according to the previous introduction, T-COL first selects the target class sample for which the ML model gives the highest probability as the prototype case. After that, a group of local greedy trees is constructed using the prototype case and the query sample. For each tree, a combination of local feature values is selected using *rss* and finally stitched into a CE.

In Figure 1, the blue dovetail arrows indicate the overall processes of T-COL. First, T-COL selects a prototype case (multiple prototype cases are available when users need diverse CEs) based on user preferences and divides its feature values with those of the query sample. After dividing the two samples, a number of subsets of local feature values are formed, with the orange and green nodes indicating the features of the prototype case and the query sample, respectively. The combination of the local feature values of the two samples is represented by constructing a local greedy tree (see Figure 2). The red path is the optimal selection path from the local greedy tree.

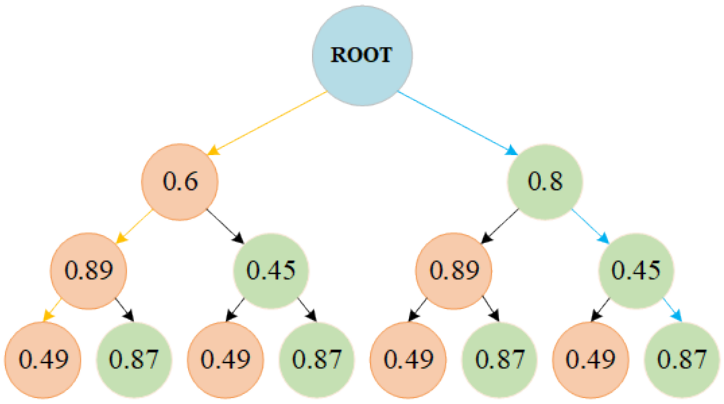

Fig. 2. An example of a local greedy tree. The depth of the tree is equal to the number of elements in the local feature subset, and the nodes at each level store the feature values of the query sample and the prototype case, respectively. By traversing each path of the local greedy tree, several sets of local feature combinations can be obtained, and the optimal local feature combination can be selected according to the preset rules.

If the user prefers more robust CEs, T-COL selects samples of the target category with the highest probability as prototype cases. In addition, 'rss' (a local optimal objective, see Section 4.4) will be used as a rule for feature value selection after the construction of local greedy trees. Ultimately, T-COL selects one or more combinations of feature values that match the user's preferences as CEs, from the set of feature values of the selected prototype cases and the query sample.

4.3 Local Greedy Tree

A local greedy tree is a tree structure used to represent the combination of local feature values of the prototype case P and the query sample S. The nodes of the tree consist of the feature values from a prototype case and the query sample.

For example, when the encoded feature vectors of the first three features of the two samples are $\ddot{x}_l = [0.6, 0.89, 0.49]$ and $x_l = [0.8, 0.45, 0.87]$ respectively, a local greedy tree can be constructed as shown in Figure 2. The orange nodes indicate the encoded feature value of the prototype case P, and the green nodes indicate the encoded feature value of the query sample S. The yellow paths indicate the prototype case's local feature subsets, and the blue paths indicate the local feature subsets of the query sample, while black paths denote other paths indicating arbitrary combinations of feature values.

A local greedy tree is a full binary tree, where the nodes of the tree can be classified into prototype and query nodes depending on their origin. Based on this, a locally greedy tree can be represented as $LGT = Tree(V = (P \cup S), E)$. Then, LGT can be used to select the local optimal feature value combination.

4.4 Conditions for Capturing User Preferences

To construct a local greedy tree and use it to select local feature values, it is necessary to set up conditions, which contain the prototype case screening rules and the local optimal objectives. The prototype case screening and feature value selection processes can be set up with different rules to suit general user preferences. On the basis of this, we further specify the concrete implementation of the two optional components corresponding to general user preferences for T-COL.

For dedicated preference A, CEs with a smaller number of distinct feature values need to be generated. Therefore, when selecting the prototype cases we chose samples of the target category with fewer different feature values from the query sample as the prototype cases. In addition, we define the feature value selection rule "few-counterfactual score (*fcs*)" as shown in (3).

$$fcs = \text{sigmoid}(\cos(\hat{x}, \ddot{x})) \times \sum_{i=1}^{n} \text{equ}(|\hat{x}_i - x_i|) \tag{3}$$

$$\text{equ}(x) = \begin{cases} 1, & x = 0 \\ 0, & x \neq 0 \end{cases} \tag{4}$$

where n denotes the number of features in each sample, $\ddot{x}$ denotes a prototype case, and (4) maps the difference in feature values to the difference in the number of feature values. The first component in (3) indicates the degree of resemblance between the current combination of feature values and the prototype case which means the probability that the current combination of feature values will be classified as the target class by the ML model, and the second part indicates the number of feature values that differ between the current combination of feature values and the query sample. *fcs* combinations with the maximum or greater than a given threshold can be interpreted as alternate counterfactuals. We use a custom encoder that includes a scaler to encode the features, with categorical features being target-encoded (Pargent et al. 2022).

For minimalist user preference B, which except the CEs to be as close as possible to the query samples, we choose the nearest target category sample to the query sample as the prototype case. The "near-counterfactual score (*ncs*)" is designed as the feature value selection rule, calculated as (5).

$$ncs = \frac{\text{sigmoid}(\cos(\hat{x}, \ddot{x}))}{\exp(\text{d}(\hat{x}, x))} \tag{5}$$

The above equation consists of two parts: the upper part represents the degree of similarity between the current combination of feature values and the prototype sample, and the lower part represents the distance between the current combination of feature values and the query sample. In addition, depending on the user's preference for CEs, we scale values using the exponential and sigmoid functions, respectively.

To accommodate cautious preference C on the robustness of CEs on variable ML systems, the sample with the highest probability of being classified into the target category by the given ML model is chosen as the prototype case. In addition, we also follow the previous approach and give evaluation metrics for the combination of feature values in (6).

$$rss = \frac{\exp(\cos(\hat{x}, \ddot{x}))}{\text{sigmoid}(\text{d}(\hat{x}, x))} \tag{6}$$

To allow the generated CEs to be more securely classified as target categories, "relative similarity score (*rss*)" amplifies the importance of the similarity between the combination of feature values and the prototype case in (6).

In order to match the requirements of admirer preference D that requires similar successful cases, the target category sample with the highest similarity to the query sample is selected as the prototype case, and *rss* is used as the metric to evaluate the combination of feature values.

The largest number of samples can be found at the cluster centroid, so the sample near the cluster centroid is the optimal solution with respect to collectivist preference E. Furthermore, *rss* is equally applicable to the evaluation of the combination of feature values under such conditions.

Using *rss* as an example, the local feature value combination scores on each path of the local greedy tree in Figure 2 can be calculated, as shown in Table 1. By calculating the *rss* of the combination of local feature values

Table 1. Local "rss" of each path in LGT

path	similarity	cost	rss
<0, 0, 0>	**1**	0.6148	4.1882
<0, 0, 1>	0.9682	0.4833	4.2572
<0, 1, 0>	0.9466	0.4294	4.2542
<0, 1, 1>	0.8757	0.2	4.3658
<1, 0, 0>	0.9911	0.5814	4.2006
<1, 0, 1>	0.9729	0.44	4.3495
<1, 1, 0>	0.9128	0.38	4.1949
<1, 1, 1>	0.8757	**0**	**4.8013**

on each path of the local greedy tree, it can be concluded that the best CE path on these three feature value subsets is $< 1, 1, 1 >$, which is the local feature value subset of the query sample S.

4.5 Utilizing T-COL to Address the Two Challenges

According to the previous introduction, T-COL is an IB method with two optional components. It determines the selection rules for both prototype cases and local combinations of feature values based on general user preferences.

4.5.1 General user preferences. Capturing user preferences is the main purpose of designing T-COL and is mainly reflected in the two conditional optional components of T-COL, i.e., prototype cases screening and local optimal objective for feature values selection.

The first component controls the selection of prototype cases. In T-COL, the prototype cases are considered ideal cases about user preferences, and CEs are generated by directing the query sample to change toward such cases. The second component controls the selection of local feature value combinations. It can control how the query sample changes to the prototype case by controlling which feature values are selected.

In summary, T-COL generates CEs by controlling the query sample to change a certain degree in a certain direction. In which the direction and degree of change are determined by user preferences. For example, suppose a user wants to focus on a few things, T-COL will select samples with a few feature values different from those of the query sample as the ideal cases. In addition, T-COL selects as few different feature values from the prototype cases as possible during the change.

4.5.2 Variable Machine Learning Systems. Robust CEs on variable ML systems are designed to accommodate practical applications where the systems are frequently updated and changed. Instead of designing a model to deal with this issue, we treat it as a general user preference. As it expresses the user's expectation of CEs with a high success rate, we added Cautious preference C to the general user preferences. Preference C indicates the desire of users for more robust CEs, with the main issues being variable ML systems.

Based on T-COL, we set a condition for preference C. We choose the target category samples with the highest classification confidence on the validation model, i.e., the highest classification probability of the validation model, as the prototype cases. In addition, when selecting a combination of local feature values, T-COL encourages the selection of more feature values from the prototype cases. By changing more in the direction of the highest confidence level, more robust CEs can be generated on variable ML systems.

5 Experiments

To assess the adaptability of CEs to user preferences, we design US-Agents to simulate user experiments. The code, more experimental details, and reappearance methods are available at https://github.com/sci-m-wang/T-COL.

5.1 Datasets

Referring to the work on CE, we chose the five datasets: the Adult Income dataset (Kohavi 1996), the German Credit dataset (Eggermont et al. 2004), the Titanic dataset (Cukierski 2012), the Water Quality dataset (Tharmalingam 2023), and the Phoneme dataset (Mantovani 2015). For more details about the datasets, please refer to Appendix B.

5.2 Metrics

To facilitate US-Agents' understanding of the task and their decision-making, we evaluate the *proximity*, *sparsity*, and *validity* of CEs. Higher values indicate better *validity*, while lower values are preferable for *proximity* and *sparsity*.

To better measure the adaptability of CEs to variable ML systems, we also propose a new property in addition to the above, called *data fidelity*. It is defined as shown in (7).

$$data\ fidelity = \frac{\sum_{i=1}^{m}(w_i \times p_i)}{\sum_{i=1}^{m} w_i} \tag{7}$$

The aim of the proposed *data fidelity* is to evaluate the validity of CEs when ML systems are changed, i.e., the ability of CEs to remain classified as the target class when the ML models change. Therefore, we used the classification results of CEs by third-party models (Arbitrary ML classification models) to assess the *data fidelity* of CEs. In (7), m denotes the number of third-party models and w_i denotes the weight of the third-party models. The more discriminative it is of the original data, i.e. its classification accuracy as evaluated by the $F1-score$, the greater its corresponding weight. Meanwhile, p_i denotes the classification accuracy of the third-party models for CEs expressed as $F1-score$.

The $F1-score$ is commonly used to assess the performance of a classifier, which we use to indicate the degree of endorsement of whether CEs belong to the target class. We show the *data fidelity* of CEs generated by different methods in Table 3.

In addition to assessing general user preferences D and E, we propose the evaluation metric *centrality*, expressed as the distance between the CE and the cluster centroid of the target category samples, in the form of (8),

$$centrality = \frac{1}{n} \times \sum_{i=1}^{n} \frac{\mathrm{d}(\dot{x}_i, \check{x})}{\mathrm{d}(\dot{x}_i, \tilde{x})} \tag{8}$$

where $\check{x}$ is the cluster centroid, $\dot{x}$ denotes the n sample points in the immediate vicinity of the cluster centroid, and $\tilde{x}$ denotes the CE. A higher *centrality* of the CE indicates that the more similar it is to the majority of the target class samples, the better it matches the general user preferences.

5.3 Baselines

Three open-source methods for generating CEs are available in **DiCE** (Mothilal et al. 2020) and can provide CEs of high quality on the available evaluation metrics. As a result, **DiCE** has been used as a baseline for most related studies. We follow the same approach, using **DiCE** as the baseline method. DiCE provides three

methods to generate CEs, “genetic”, “random” and “kd-tree”, which we denote as “DiCE-g”, “DiCE-r” and “DiCE-k”, respectively. In addition, we compare the brute force method (Sokol, Hepburn, et al. 2020) as a baseline method.

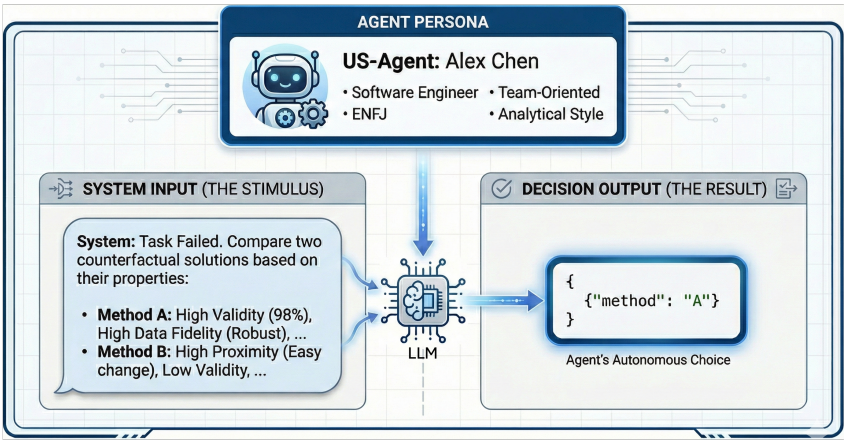

Fig. 3. An example of a simulated user experiment for preference selection. To ensure the fairness of the experiment, T-COL and the baseline methods were replaced with method A or method B. Each US-Agent is asked to choose their preferred method, A or B, based on their preferences and the properties of CEs. To facilitate statistical analysis, each agent's response is restricted to JSON format.

5.4 Experimental Settings

We randomly selected query samples, encoding the categorical features with Target Encoder (Micci-Barreca 2001). In the experiments, we generate five CEs for each query sample. For T-COL, the depth of local greedy trees was set to three.

Table 2. Validation weights of the third-party model

Model	Adult Income	German Credit	Titanic	Water Quality	Phoneme
KNN	0.73	0.66	0.93	0.62	0.84
MLP	0.75	0.67	0.95	0.64	0.82
SVM	0.74	0.69	0.93	0.62	0.8
DT	0.69	0.65	0.94	0.58	0.83
NB	0.74	0.7	0.93	0.52	0.75

A random forest model (Biau and Scornet 2016) was chosen as the validation model for the CE to verify whether the generated CE was the target class. In addition, common ML models like Decision Tree (Safavian and Landgrebe 1991) and Naive Bayesian (Rish 2001) were selected as third-party models to evaluate the *data fidelity* of the CEs. The weights of the third-party models were determined using a ten-fold cross-validation (Refaeilzadeh et al. 2009) approach. They are shown in Table 2.

To evaluate the adaptability of CEs to general user preferences, we conduct simulated user experiments using LLM-based US-Agents. We first set profiles such as MBTI personality, birthday, occupation, etc. Additionally, we assign them examples of statements, symbol usage patterns, etc., referencing (Tsubota and Kano 2024). Then, we

ask these agents with different settings to answer the questionnaire designed for real users. Finally, we compute the distributional consistency of the results of these LLM-based agents' responses to the questionnaire with the results collected in the real user research. For each real user, we screen the agents that have high consistency with them as US-Agents. After consistency screening, we select 23,987 US-Agents with InternLM (Cai et al. 2024), Mistral (M. A. Team 2024), Qwen (Q. Team 2024) and LLaMA (Grattafiori et al. 2024) as backbones. In other words, screened US-Agents made essentially the same choices on the same questionnaire as real users. An example of a simulated user experiment for preference selection is illustrated in Figure 3. An example of a complete US-Agent profile and prompts can be found in the Appendix C.

In the experiments, each agent is required to choose between T-COL and baseline methods according to the assigned preferences and the properties of CEs. T-COL is pitted against each baseline method, and the win rates are counted.

5.5 Adaptability to General User Preferences

To evaluate the adaptability of CEs generated by T-COL and baseline methods to general user preferences, we conduct simulated user experiments using LLM-based US-Agents. Each time a CE is selected, the corresponding generation method is credited with a win. As references, we evaluate the properties of the CEs generated by the T-COL and baseline methods. Since T-COL is an IB method and controls the source of feature values by screening prototype examples, it can effectively ensure the *feasibility*, *consistency*, and *data manifold closeness* of CEs. Thus, we evaluate *centrality*, *data fidelity*, *proximity*, *sparsity*, and *validity*. As described in Section 5.4, we generate CEs oriented to randomly selected query samples using different methods, and then compute their performance on these metrics. The results are shown in Table 3.

Based on these results, we organize large-scale simulated user experiments to randomly match CEs generated by two methods, asking US-Agents to choose the one that better matches the specified user's preference. The results are shown in Table 4.

The results show that the T-COL approach is more resilient to general user preferences. In addition, from the performance on *data fidelity* metric and results of simulated user experiments, it is clear that T-COL can better address the challenges of variable ML systems compared to baseline methods.

5.6 Efficiency Analysis

The efficiency of generating CEs is also very important in applications. During our experiments, we found that the efficiency of DiCE and T-COL in generating CEs differed significantly. Therefore, we analyzed the time complexity of T-COL and DiCE and recorded the actual execution time of the experiment to generate five CEs for each query sample.

5.6.1 Time Complexity Analysis. Define m as the number of CEs generated for each sample, the number of features as n, k as the number of samples generated during evaluation, and T as the number of iterations required to optimize the loss function. To facilitate the analysis of the time complexity of T-COL, we define d as the depth of the local greedy tree.

The main time-consuming steps in DiCE's process of generating CEs are:

- Optimizing the loss function for counterfactual generation.
- Approximating the counterfactual to the decision boundary of the ML model.

The execution time of the first step is related to the number of CEs to be generated and the number of iterations. For each CE to be generated, a target loss value can be obtained based on the loss function. DiCE optimizes this loss value by means of an iterative approach, where each iteration requires the calculation of the distance between the query sample and the generated counterfactual. The time complexity of this part is $O(T * mn)$. In the second step, a large number of samples need to be generated around the query sample at different distances.

Table 3. The contrasting properties of counterfactual explanations. BF denotes the brute force method. **Bolded values** indicate the best-performing results. '-1' indicates that all the CEs generated by this method are invalid.

Datasets	Properties	BF	DiCE-g	DiCE-k	DiCE-r	T-COL-a	T-COL-b	T-COL-c	T-COL-d	T-COL-e
Adult Income	centrality	0.83	0.91	0.93	0.99	1	0.81	0.73	**0.68**	**0.68**
	data fidelity	0.52	**0.74**	0.67	0.52	0.36	0.45	0.62	0.68	0.71
	proximity	1.2	1.17	1.22	1.12	**0.96**	1.09	1.09	1.07	1.07
	sparsity	0.53	0.45	0.42	0.45	**0.33**	0.5	0.5	0.47	0.45
	validity	0.57	0.56	0.56	0.52	**1**	0.91	0.94	0.98	0.96
German Credit	centrality	0.88	0.88	0.96	1.08	1.02	0.89	**0.73**	**0.73**	**0.73**
	data fidelity	0.61	0.77	0.73	0.18	0.6	0.71	0.78	0.78	**0.79**
	proximity	1.3	0.92	0.89	0.85	0.84	0.66	**0.65**	**0.65**	**0.65**
	sparsity	0.63	0.43	0.43	0.5	0.35	0.26	0.25	0.26	**0.23**
	validity	**1**	0.96	**1**	0.36	**1**	0.93	0.97	0.99	0.99
Phoneme	centrality	-1	0.8	0.8	0.96	1.27	**0.75**	**0.75**	**0.75**	**0.75**
	data fidelity	0.12	0.68	0.69	0.53	1	0.72	**0.76**	0.74	0.74
	proximity	-1	0.92	0.92	0.89	1.3	0.54	0.54	**0.5**	**0.5**
	sparsity	-1	1	1	0.88	0.84	**0.48**	**0.48**	**0.48**	**0.48**
	validity	0	**1**	**1**	0.68	**1**	0.96	**1**	**1**	**1**
Titanic	centrality	0.88	0.98	0.87	0.98	0.86	0.65	0.65	0.65	**0.62**
	data fidelity	0.49	0.95	0.98	0.92	**1**	0.99	0.99	**1**	**1**
	proximity	1.23	1.11	1.17	1.05	1.32	**1.03**	**1.03**	**1.03**	**1.03**
	sparsity	0.69	0.49	0.47	0.47	0.49	**0.36**	**0.36**	**0.36**	**0.36**
	validity	0.54	**1**	0.96	0.96	**1**	**1**	**1**	**1**	**1**
Water Quality	centrality	0.88	**0.77**	0.83	0.85	0.99	0.82	0.82	0.82	0.82
	data fidelity	0.18	0.47	0.67	0.75	**0.85**	0.76	0.77	0.81	0.81
	proximity	1.2	0.97	0.89	0.99	1.04	0.57	0.57	0.57	**0.54**
	sparsity	1	1	1	0.82	0.73	0.47	0.47	**0.42**	0.44
	validity	0.5	0.84	**1**	0.68	**1**	0.97	0.97	0.97	0.97

Table 4. Preference selection results from simulated user experiments, with values indicating the winning rate when T-COL confronts the baseline methods.

Preference	A		B		C		D		E	
Baseline	**DiCE**	**BF**	**DiCE**	**BF**	**DiCE**	**BF**	**DiCE**	**BF**	**DiCE**	**BF**
Adult Income	100.00	100.00	100.00	100.00	99.92	100.00	77.65	100.00	100.00	100.00
German Credit	94.08	100.00	100.00	100.00	53.63	99.83	50.21	79.65	64.97	99.75
Phoneme	63.39	100.00	67.97	100.00	77.65	100.00	50.42	100.00	63.22	100.00
Titanic	100.00	100.00	100.00	100.00	99.92	100.00	91.49	99.92	100.00	100.00
Water Quality	81.23	100.00	85.57	100.00	100.00	100.00	50.13	95.83	98.17	99.92

In addition, the distance from the query sample to all counterfactuals needs to be computed. The time complexity of this part is $O(kmn)$. In summary, the time complexity of DiCE is $O((T + k) * mn)$.

The main time-consuming processes of T-COL are the second and third steps, while the first step of feature partitioning takes very little time and can be ignored. For the generation of each counterfactual explanation, the second step requires the computation of the values corresponding to the features on all paths of each local greedy tree, with a time complexity related to the depth of the tree and the number of features in the samples as $O(m * \frac{n}{d} * 2^{(d-1)})$. The third step requires traversing all the generated counterfactual paths and performing feature selection with a time complexity of $O(mn)$. Note that according to the setup of this paper, d is a constant

Table 5. Runtime (seconds) for generating CEs

Datasets	T-COL-a	T-COL-b	T-COL-c	T-COL-d	T-COL-e	DiCE-r	DiCE-g	DiCE-k	BF
Adult Income	0.3	0.29	**0.05**	0.27	0.06	0.61	16.19	1.05	4.45
German Credit	0.53	0.52	**0.1**	0.47	**0.1**	0.85	2.36	19.54	871.45
Titanic	0.30	0.29	**0.05**	0.27	0.06	0.85	2.61	9.36	10.25
Water Quality	0.17	0.17	**0.03**	0.15	**0.03**	3706.03	0.31	1373.27	2021.22
Phoneme	0.13	0.13	**0.02**	0.12	**0.02**	605.43	0.39	600.4	0.22

between three and nine, which gives $1 < \frac{1}{d} * 2^{(d-1)} < 29$, and the time complexity of T-COL can be abbreviated as $O(mn)$.

5.6.2 Runtime Comparison. Based on the previous analysis, the time complexity of T-COL is much smaller than that of DiCE since it does not require a lot of optimization computations. In this section, we list the actual running time of the two methods in our experiments in Table 5, for different datasets.

As shown in Table 5, T-COL generates CEs much more efficiently than DiCE, allowing for real-time response, which is very important in applications. With a local greedy tree depth of 3, T-COL takes less than a second to generate all five CEs for each query sample while "DiCE-r" even takes more than 3,700 seconds to generate five CEs for a sample query in Water Quality.

In our experiments, we found that the generation time difference of T-COL shows a strong correlation with the number of sample features for a fixed local greedy tree depth, which is consistent with the results of our time complexity analysis. In addition, we find that the time required by different optimization methods for DiCE varies significantly in the face of different data types. The "random" method is faster to generate when there are many categorical features, while the "genetic" method is faster to generate when there are many numerical features.

6 Conclusion

In this paper, we define general user preferences and propose T-COL, an instance-based method for generating CEs to capture these preferences. We set different conditions for prototype screening and the local optimal objective of T-COL to generate CEs that can better adapt to different general user preferences. Furthermore, we investigate generating more robust CEs for variable ML systems. To solve this problem, we incorporate a user preference and establish a condition to guide T-COL. Moreover, we evaluate the adaptability of CE generation methods to preferences by conducting simulated user experiments with LLM-based US-Agents that are consistent with the performance of real users. Our experiments on five benchmark datasets demonstrate that T-COL better adapts to different user preferences and effectively handles variable ML systems.

Limitations and Future Work

While T-COL demonstrates effectiveness in generating user-adaptive counterfactuals, our approach has two inherent limitations that warrant further investigation:

Empirical Basis for Function Combinations. The selection of specific functions is primarily driven by our empirical effectiveness in capturing distinct aspects of user preferences, rather than theoretical uniqueness. Although these choices are widely used in the field of explainable AI, alternative functions (e.g., logarithmic scaling for cost sensitivity, hyperbolic tangent for bounded preferences) may prove equally or more suitable in certain domains. This design flexibility—while enabling adaptability—introduces subjectivity in function selection. Future work should establish rigorous criteria for function selection and explore domain-specific formulations.

Discrepancy Between Counterfactual Distance and Practical Actionability. In the research, to simplify the problem, we treat the counterfactual distance as equivalent to the difficulty of achieving the objective, which does not inherently reflect real-world implementation difficulty. This decoupling of geometric distance from true actionability represents a fundamental challenge in counterfactual explanation research. T-COL partially mitigates this through the flexible design of conditions, but deeper integration of domain knowledge is needed to bridge this gap.

In future work, we plan to expand T-COL by adding general user preferences to accommodate complex real-world scenarios. We would like to add further constraints on T-COL and enrich its structure to guarantee CEs' properties better and address emerging challenges. In addition, we will further customize the condition settings under each user preference. Furthermore, we note that the CEs generated by T-COL do not perform consistently with the conditions in terms of correspondence properties. Thus, we will further explore the relationship between general user preferences and the properties of CEs.

Acknowledgments

The work was supported by the National Natural Science Foundation of China (62172086, 62272092).

References

A. Abid, M. Yuksekgonul, and J. Zou. 17–23 Jul 2022. "Meaningfully debugging model mistakes using conceptual counterfactual explanations." In: *Proceedings of the 39th International Conference on Machine Learning* (Proceedings of Machine Learning Research). Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato. Vol. 162. PMLR, (17–23 Jul 2022), 66–88. https://proceedings.mlr.press/v162/abid22a.html.

E. Albini, J. Long, D. Dervovic, et al.. 2022. "Counterfactual shapley additive explanations." In: *2022 ACM Conference on Fairness, Accountability, and Transparency*, 1054–1070.

A. Artelt and B. Hammer. Nov. 2019. *On the computation of counterfactual explanations – A survey.* arXiv:1911.07749 [cs, stat]. (Nov. 2019). doi:10.48550/arXiv.1911.07749.

A. Artelt, V. Vaquet, R. Velioglu, et al.. 2021a. "Evaluating Robustness of Counterfactual Explanations." In: *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, 01–09. doi:10.1109/SSCI50451.2021.9660058.

A. Artelt, V. Vaquet, R. Velioglu, et al.. 2021b. "Evaluating robustness of counterfactual explanations." In: *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 01–09.

A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, et al.. June 2020. "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI." *Information Fusion*, 58, (June 2020), 82–115. doi:10.1016/j.inffus.2019.12.012.

S. Benartzi. 2017. *The smarter screen: Surprising ways to influence and improve online behavior.* Penguin.

S. Bhatt. 2019. *The Attention Deficit.* Springer.

G. Biau and E. Scornet. 2016. "A random forest guided tour." *Test*, 25, 2, 197–227.

Z. Cai, M. Cao, H. Chen, K. Chen, K. Chen, et al.. 2024. *InternLM2 Technical Report.* (2024). arXiv: 2403.17297 [cs.CL].

Y. Chen, P. Cramton, J. A. List, and A. Ockenfels. 2021. "Market design, human behavior, and management." *Management Science*, 67, 9, 5317–5348.

F. Cheng, Y. Ming, and H. Qu. 2021. "DECE: Decision Explorer with Counterfactual Explanations for Machine Learning Models." *IEEE Transactions on Visualization and Computer Graphics*, 27, 2, 1438–1447. doi:10.1109/TVCG.2020.3030342.

J. Cito, I. Dillig, V. Murali, et al.. 2022. "Counterfactual Explanations for Models of Code." In: *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice* (ICSE-SEIP '22). Association for Computing Machinery, Pittsburgh, Pennsylvania, 125–134. ISBN: 9781450392266. doi:10.1145/3510457.3513081.

W. Cukierski. 2012. *Titanic - Machine Learning from Disaster.* (2012). https://kaggle.com/competitions/titanic.

X. Dai, M. T. Keane, L. Shalloo, et al.. 2022. "Counterfactual explanations for prediction and diagnosis in xai." In: *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, 215–226.

J. Del Ser, A. Barredo-Arrieta, N. Díaz-Rodríguez, F. Herrera, A. Saranti, and A. Holzinger. 2024. "On generating trustworthy counterfactual explanations." *Information Sciences*, 655, 119898. doi:https://doi.org/10.1016/j.ins.2023.119898.

D. Dua and C. Graff. 2017. *UCI Machine Learning Repository.* (2017). http://archive.ics.uci.edu/ml.

J. Eggermont, J. N. Kok, and W. A. Kosters. 2004. "Genetic programming for data classification: Partitioning the search space." In: *Proceedings of the 2004 ACM symposium on Applied computing*, 1001–1005.

G. Filandrianos, K. Thomas, E. Dervakos, et al.. 2022. "Conceptual Edits as Counterfactual Explanations." In: *AAAI Spring Symposium: MAKE.*

K. Främling. 2022. "Contextual Importance and Utility: A Theoretical Foundation." In: *AI 2021: Advances in Artificial Intelligence*. Ed. by G. Long, X. Yu, and S. Wang. Springer International Publishing, Cham, 117–128. ISBN: 978-3-030-97546-3.

Y. Goyal, Z. Wu, J. Ernst, et al.. 2019. "Counterfactual visual explanations." In: *International Conference on Machine Learning*. PMLR, 2376–2384.

A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, et al.. 2024. *The Llama 3 Herd of Models*. (2024). https://arxiv.org/abs/2407.21783 arXiv: 2407.21783 [cs.AI].

R. Guidotti. Apr. 2022. *Counterfactual explanations and how to find them: literature review and benchmarking*. en. (Apr. 2022). doi:10.1007/s10618-022-00831-6.

R. Guidotti, A. Monreale, F. Giannotti, et al.. 2019. "Factual and counterfactual explanations for black box decision making." *IEEE Intelligent Systems*, 34, 6, 14–23.

R. Guidotti and S. Ruggieri. 2021. "Ensemble of Counterfactual Explainers." en. In: *Discovery Science*. Ed. by C. Soares and L. Torgo. Springer International Publishing, Cham, 358–368. ISBN: 978-3-030-88942-5. doi:10.1007/978-3-030-88942-5_28.

D. Gunning. 2017. "Explainable artificial intelligence (xai)." *Defense advanced research projects agency (DARPA), nd Web*, 2, 2, 1.

D. Gunning, M. Stefik, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang. 2019. "XAI—Explainable artificial intelligence." *Science robotics*, 4, 37, eaay7120.

F. Hamman, E. Noorani, S. Mishra, D. Magazzeni, and S. Dutta. 23–29 Jul 2023. "Robust Counterfactual Explanations for Neural Networks With Probabilistic Guarantees." In: *Proceedings of the 40th International Conference on Machine Learning* (Proceedings of Machine Learning Research). Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett. Vol. 202. PMLR, (23–29 Jul 2023), 12351–12367. https://proceedings.mlr.press/v202/hamman23a.html.

J. Jiang, J. Lan, F. Leofante, A. Rago, and F. Toni. Nov. 2024. "Provably Robust and Plausible Counterfactual Explanations for Neural Networks via Robust Optimisation." In: *Proceedings of the 15th Asian Conference on Machine Learning* (Proceedings of Machine Learning Research). Ed. by B. Yanıkoğlu and W. Buntine. Vol. 222. PMLR, (Nov. 2024), 582–597. https://proceedings.mlr.press/v222/jiang24a.html.

A.-H. Karimi, G. Barthe, B. Schölkopf, and I. Valera. Dec. 2022. "A Survey of Algorithmic Recourse: Contrastive Explanations and Consequential Recommendations." *ACM Comput. Surv.*, 55, 5, Article 95, (Dec. 2022), 29 pages. doi:10.1145/3527848.

M. T. Keane and B. Smyth. 2020. "Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable ai (xai)." In: *International Conference on Case-Based Reasoning*. Springer, 163–178.

M. T. Keane, E. M. Kenny, E. Delaney, et al.. Feb. 2021. *If Only We Had Better Counterfactual Explanations: Five Key Deficits to Rectify in the Evaluation of Counterfactual XAI Techniques*. arXiv:2103.01035 [cs]. (Feb. 2021). doi:10.48550/arXiv.2103.01035.

D. P. Kingma and J. Ba. Jan. 2017. *Adam: A Method for Stochastic Optimization*. arXiv:1412.6980 [cs]. (Jan. 2017). doi:10.48550/arXiv.1412.6980.

R. Kohavi. Aug. 1996. "Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid." In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (KDD'96). AAAI Press, Portland, Oregon, (Aug. 1996), 202–207. Retrieved Mar. 17, 2024 from.

U. Kuhl, A. Artelt, and B. Hammer. May 2022. *Let's Go to the Alien Zoo: Introducing an Experimental Framework to Study Usability of Counterfactual Explanations for Machine Learning*. arXiv:2205.03398 [cs]. (May 2022). doi:10.48550/arXiv.2205.03398.

T. Laugel, M.-J. Lesot, C. Marsala, et al.. June 2019. *Issues with post-hoc counterfactual explanations: a discussion*. en. (June 2019). doi:10.48550/arXiv.1906.04774.

T. Le, T. Miller, R. Singh, et al.. June 2022. *Improving Model Understanding and Trust with Counterfactual Explanations of Model Confidence*. arXiv:2206.02790 [cs]. (June 2022). doi:10.48550/arXiv.2206.02790.

J. Lee, W. Chu, and C. Baumann. 2024. "The Psychology Behind Design." *Springer Books*.

R. G. Mantovani. 2015. *phoneme*. (2015). https://huggingface.co/datasets/mstz/phoneme.

D. Maragno, T. E. Röber, and I. Birbil. Dec. 2022. *Counterfactual Explanations Using Optimization With Constraint Learning*. arXiv:2209.10997 [cs]. (Dec. 2022). doi:10.48550/arXiv.2209.10997.

J. M. Metsch, A. Saranti, A. Angerschmid, B. Pfeifer, V. Klemt, A. Holzinger, and A.-C. Hauschild. 2024. "CLARUS: An interactive explainable AI platform for manual counterfactuals in graph neural networks." *Journal of Biomedical Informatics*, 150, 104600. doi:https://doi.org/10.1016/j.jbi.2024.104600.

D. Micci-Barreca. 2001. "A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems." *ACM SIGKDD Explorations Newsletter*, 3, 1, 27–32.

C. Molnar. 2020. *Interpretable machine learning*. Lulu. com.

R. K. Mothilal, A. Sharma, and C. Tan. 2020. "Explaining machine learning classifiers through diverse counterfactual explanations." In: *Proceedings of the 2020 conference on fairness, accountability, and transparency*, 607–617.

F. Pargent, F. Pfisterer, J. Thomas, and B. Bischl. Nov. 2022. "Regularized target encoding outperforms traditional methods in supervised machine learning with high cardinality features." *Computational Statistics*, 37, 5, (Nov. 2022), 2671–2692. doi:10.1007/s00180-022-01207-6.

M. Pawelczyk, C. Agarwal, S. Joshi, et al.. 28–30 Mar 2022. "Exploring Counterfactual Explanations Through the Lens of Adversarial Examples: A Theoretical and Empirical Analysis." In: *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics* (Proceedings of Machine Learning Research). Ed. by G. Camps-Valls, F. J. R. Ruiz, and I. Valera. Vol. 151. PMLR, (28–30 Mar 2022), 4574–4594. https://proceedings.mlr.press/v151/pawelczyk22a.html.

A. Piccione, J. Berkery, S. Sabbagh, et al.. 2022. "Predicting resistive wall mode stability in NSTX through balanced random forests and counterfactual explanations." *Nuclear Fusion*, 62, 3, 036002.

P. Rasouli and I. Chieh Yu. 2022. "CARE: Coherent actionable recourse based on sound counterfactual explanations." *International Journal of Data Science and Analytics*, 17, 1–26.

P. Refaeilzadeh, L. Tang, and H. Liu. 2009. "Cross-validation." *Encyclopedia of database systems*, 5, 532–538.

I. Rish. 2001. "An empirical study of the naive Bayes classifier." In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. Vol. 3, 41–46.

P. Rodríguez, M. Caccia, A. Lacoste, et al.. Oct. 2021. "Beyond Trivial Counterfactual Explanations With Diverse Valuable Explanations." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. (Oct. 2021), 1056–1065.

S. R. Safavian and D. Landgrebe. 1991. "A survey of decision tree classifier methodology." *IEEE transactions on systems, man, and cybernetics*, 21, 3, 660–674.

R. Shang, K. K. Feng, and C. Shah. 2022. "Why Am I Not Seeing It? Understanding Users' Needs for Counterfactual Explanations in Everyday Recommendations." In: *2022 ACM Conference on Fairness, Accountability, and Transparency*, 1330–1340.

C. R. Silva, M. Bowling, and L. H. Lelis. 2021. "Teaching people by justifying tree search decisions: An empirical study in curling." *Journal of Artificial Intelligence Research*, 72, 1083–1102.

D. Slack, A. Hilgard, H. Lakkaraju, et al.. 2021. "Counterfactual Explanations Can Be Manipulated." In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 62–75. https://proceedings.neurips.cc/paper/2021/file/009c434cab57de48a31f6b669e7ba266-Paper.pdf.

B. I. Smith, C. Chimedza, and J. H. Bührmann. 2022. "Individualized help for at-risk students using model-agnostic and counterfactual explanations." *Education and Information Technologies*, 27, 2, 1539–1558.

B. Smyth and M. T. Keane. 2022. "A few good counterfactuals: generating interpretable, plausible and diverse counterfactual explanations." In: *International Conference on Case-Based Reasoning*. Springer, 18–32.

K. Sokol and P. Flach. Jan. 2019. "Counterfactual explanations of machine learning predictions: Opportunities and challenges for AI safety." English. In: *Proceedings of the AAAI Workshop on Artificial Intelligence Safety 2019* (CEUR Workshop Proceedings). Vol. 2301. 2019 AAAI Workshop on Artificial Intelligence Safety, SafeAI 2019 ; Conference date: 27-01-2019. CEUR Workshop Proceedings, (Jan. 2019).

K. Sokol, A. Hepburn, R. Poyiadzi, M. Clifford, R. Santos-Rodriguez, and P. Flach. 2020. "FAT Forensics: A Python Toolbox for Implementing and Deploying Fairness, Accountability and Transparency Algorithms in Predictive Systems." *Journal of Open Source Software*, 5, 49, 1904. doi:10.21105/joss.01904.

I. Stepin, J. M. Alonso, A. Catala, et al.. 2021. "A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence." *IEEE Access*, 9, 11974–12001.

M. A. Team. 2024. *Mistral-Nemo-Instruct-2407*. (2024). Retrieved Jan. 8, 2025 from https://huggingface.co/mistralai/Mistral-Nemo-Instruct-2407.

Q. Team. Sept. 2024. *Qwen2.5: A Party of Foundation Models*. (Sept. 2024). https://qwenlm.github.io/blog/qwen2.5/.

M. Tesic and U. Hahn. May 2022. *Can counterfactual explanations of AI systems' predictions skew lay users' causal intuitions about the world? If so, can we correct for that?* arXiv:2205.06241 [cs]. (May 2022). doi:10.48550/arXiv.2205.06241.

L. Tharmalingam. 2023. *Water Quality and Potability*. (2023). https://www.kaggle.com/datasets/uom190346a/water-quality-and-potability.

G. Tolomei, F. Silvestri, A. Haines, and M. Lalmas. 2017. "Interpretable Predictions of Tree-based Ensembles via Actionable Feature Tweaking." In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD '17). Association for Computing Machinery, Halifax, NS, Canada, 465–474. ISBN: 9781450348874. doi:10.1145/3097983.3098039.

K. H. Tran, A. Ghazimatin, and R. Saha Roy. 2021. "Counterfactual Explanations for Neural Recommenders." In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1627–1631.

S. Tsirtsis and M. Gomez Rodriguez. 2020. "Decisions, counterfactual explanations and strategic behavior." *Advances in Neural Information Processing Systems*, 33, 16749–16760.

Y. Tsubota and Y. Kano. Sept. 2024. "Text Generation Indistinguishable from Target Person by Prompting Few Examples Using LLM." In: *Proceedings of the 2nd International AIWolfDial Workshop*. Ed. by Y. Kano. Association for Computational Linguistics, Tokyo, Japan, (Sept. 2024), 13–20. https://aclanthology.org/2024.aiwolfdial-1.2/.

B. Ustun, A. Spangher, and Y. Liu. 2019. "Actionable recourse in linear classification." In: *Proceedings of the conference on fairness, accountability, and transparency*, 10–19.

S. Verma, V. Boonsanong, M. Hoang, et al.. Nov. 2022. *Counterfactual Explanations and Algorithmic Recourses for Machine Learning: A Review*. arXiv:2010.10596 [cs, stat]. (Nov. 2022). doi:10.48550/arXiv.2010.10596.

S. Verma, J. Dickerson, and K. Hines. June 2021. *Counterfactual Explanations for Machine Learning: Challenges Revisited*. arXiv:2106.07756 [cs]. (June 2021). doi:10.48550/arXiv.2106.07756.

M. Virgolin and S. Fracaros. Sept. 2022. *On the Robustness of Sparse Counterfactual Explanations to Adverse Perturbations*. arXiv:2201.09051 [cs]. (Sept. 2022). doi:10.48550/arXiv.2201.09051.

S. Wachter, B. Mittelstadt, and C. Russell. 2018. "Counterfactual explanations without opening the black box: automated decisions and the GDPR." *Harvard Journal of Law and Technology*, 31, 2.

M. Wang et al.. 2024. *Minstrel: Structural Prompt Generation with Multi-Agents Coordination for Non-AI Experts.* (2024). https://arxiv.org/abs/2409.13449 arXiv: 2409.13449 (cs.CL).

G. Warren, M. T. Keane, and R. M. J. Byrne. Apr. 2022. *Features of Explainability: How users understand counterfactual and causal explanations for categorical and continuous features in XAI.* arXiv:2204.10152 [cs]. (Apr. 2022). doi:10.48550/arXiv.2204.10152.

G. P. Wellawatte, A. Seshadri, and A. D. White. 2022. "Model agnostic generation of counterfactual explanations for molecules." *Chemical science*, 13, 13, 3697–3705.

Y. Yacoby, B. Green, C. L. G. Jr, et al.. Oct. 2022. ""If it didn't happen, why would I change my decision?": How Judges Respond to Counterfactual Explanations for the Public Safety Assessment." en. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, 10, (Oct. 2022), 219–230. doi:10.1609/hcomp.v10i1.22001.

A The User Research

In this section, we outline more specific details regarding user research.

A.1 Questionnaire

To verify the validity and completeness of our proposal and to analyze the tendency of different real users towards these preferences, we designed the following questionnaire and organized a user survey.

Problem Solving Preference Research

1. Your Gender:

- ○ Male
- ○ Female

2. Your Age:

- ○ Below 18
- ○ 18 to 25
- ○ 26 to 30
- ○ 31 to 40
- ○ 41 to 50
- ○ Over 50

3. Your Country:

<Select a country>

4. Your Sector:

<Select a sector from the list of "Manufacturing", "Construction", "Education/Training", etc. >

CAUTION: *Choose up to two for all of the following multiple- choice questions, and one is allowed.*

5. If you failed at something, and there was a magic machine that could tell you the solution. Which solution would you favour?

- □ Wish to work on one aspect to make it happen, don't change it in many ways.
- □ Wish to make it as easy as possible, can be multi-faceted, but preferably change only a little in each area.
- □ A solution with the highest possible success rate.
- □ Solutions that have success stories.
- □ What most people do.
- □ Others. <Additional content required>

6. Suppose you are a drug researcher and you have the following reference scenarios after a failed development.

- □ Only a few structural differences from the programme you designed.
- □ Less differences between structures compared to yours.
- □ Structure with the highest success rate.
- □ Structure with success stories.
- □ Structure adopted by most people.
- □ Others. <Additional content required>

7. If an application for a loan is rejected, there are several successful cases available to you. Which one would you choose?

- □ Different from your situation in only a few ways.
- □ Very much like your situation.
- □ Option most likely to be successful.
- □ Option that has been applied successfully before.
- □ Most people's situation.

☐ Others. <Additional content required>

8. Suppose you are a shopper and you can't always pick the items that your customers like, and there are a few exemplary shoppers to learn from, who would you learn from?

☐ Different from you in a few ways.
☐ Very much like you.
☐ A sales champion.
☐ The shopper who successfully sells the product.
☐ The average, popular shopper.
☐ Others. <Additional content required>

9. Suppose you were a teacher with a class of underachieving students and there were other teachers you could ask for advice, who would you ask?

☐ Different from you in a few ways.
☐ Very much like you.
☐ A top model teacher.
☐ Teachers with better teaching results.
☐ Ordinary teachers.
☐ Others. <Additional content required>

10. Suppose you are a farmer, and the locust plague always fails in prevention. There are some programs, which one would you learn?

☐ The farm is different from you in a few ways
☐ The farm situation is very much like yours
☐ Farms of famous farmers
☐ Farms that have effectively prevented locust plagues
☐ Most farms
☐ Others. <Additional content required>

The first four questions in the questionnaire dealt with the user's personal information, such as age, gender, nationality, and the industry in which he or she is engaged. The latter questions correlate well with the general user preferences we predefined. In addition to a generic question, we set up scenarios in five domains - medicine, finance, sales, education, and agriculture - with options associated with general user preferences.

A.2 User Profile

Considering user privacy and ethical issues, we do not show specific user samples directly. As an alternative, we show the proportions of different user characteristics.

Table 6. Gender distribution in user research.

Gender	Proportion (%)
Male	80.6
Female	19.4

A.2.1 Gender. The gender distribution in the real user research is shown in Table 6. The number of women in the user study was relatively small due to the demographic composition of the audience groups promoted. However, there is no order of magnitude significant deviation between the minority and the majority, and it still expresses to some extent the tendency of the users towards these preferences. In future work, we will also consider further expanding the groups included in the user research and balancing the gender distribution.

A.2.2 Age. As shown in Table 7, the user research we organized covered the entire age group. Among them, the research subjects are mainly concentrated between the ages of 18 and 40.

A.2.3 Region. In the user research, most users are from Asia, such as China, Singapore, Pakistan, etc. In addition, there are also some users from the United States, France, Germany, etc. The specific distribution is shown in Figure 4.

Table 7. Age distribution in user research.

Age	Below 18	18~25	26~30	31~40	41~50	Over 50
Proportion (%)	1.49	37.31	32.84	19.40	4.48	4.48

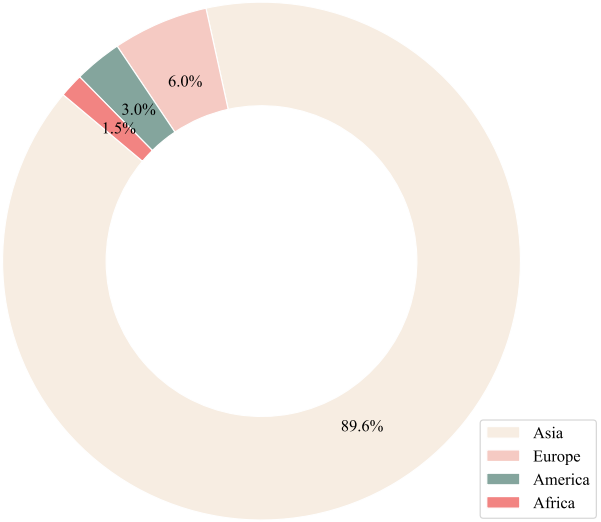

Fig. 4. The regional distribution of user research. The regions are grouped by continent.

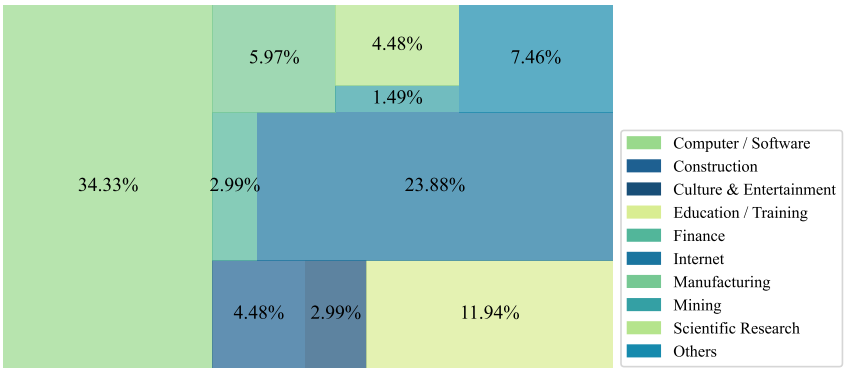

Fig. 5. The industry distribution in user research. Others include industries with few participating users in the research, such as *Healthcare or Social Security*, *Logistics and Transport*, *Professional Services* (e.g., legal or consultancy services), *Wholesale and Retail*.

A.2.4 Sector. As shown in Figure 5, the user research covered practitioners from a wide range of industries, with the computer and Internet industries dominating.

Above all, the composition of the respondents for the user research we organized is very diverse, and it can be thought that the results of the research are universal.

A.3 User Preference Research

In the questionnaire, we collected information such as age, gender, industry, and nationality to ensure diverse and comprehensive samples. The content of the questionnaire and specific user distribution are shown in Appendix A. The distribution of the proportion of users choosing different preferences is shown in Figure 6.

Fig. 6. Heatmap for general user preference research. The more users that select a certain preference, the darker the color of the corresponding area. The values in these areas indicate the percentage of users who chose that preference for the corresponding scenario.

In addition to the general scenario, we also set up scenarios for different fields. The specific questionnaire is available at Appendix A.1. It is worth noting that most users are favoring the cautious preference *C* and the admirer preference *D*. Preference *C* corresponds to the solution with the highest success possibility. This user preference is consistent with our goal of expecting more robust CEs on variable ML systems. Preference *D* to some extent inspired our approach of generating CEs based on prototype samples.

B Datasets

The introduction and processing details of the datasets are as follows:

- *Adult Income*. The dataset contains information on population, education, etc., based on the 1994 Census database and is available from the UCI Machine Learning Repository (Dua and Graff 2017). In this paper, a pre-processed version from (Mothilal et al. 2020) was chosen to filter eight of the features. The task of the classification model is to classify whether the individual income of each sample exceeds 50,000 dollars.
- *German Credit*. The information in this dataset is obtained from banks in relation to personal loans, such as the number of credit cards currently held by a particular bank, the duration of current employment, and other information on a total of 20 characteristics. We use the version obtained directly from the UCI

database without processing. The task of the classification model is to determine whether a user is a credit risk or not by determining their credit type based on their attributes.

- *Titanic*. This dataset is derived from the Titanic incident, in which the gender, age, ticket number, class of ticket, and other attributes of some of the passengers were collected. The dataset contains information about 891 passengers in total, and we removed the instances containing null values. The task of the classification model is to determine whether a passenger will eventually survive.
- *Water Quality*. This dataset contains water quality measurements and assessments related to potability. Each sample contains nine attributes of one water sample, such as pH, hardness, etc. The dataset contains data for a total of 3276 water instances, and we removed some of the instances that contained null values. The task of the classification model is to discriminate whether a water sample is potable or not.
- *Phoneme*. The dataset collects five different attributes from 1809 isolated syllables to characterize each vowel. The dataset contains a total of 5404 instances. The task of the classification model is to distinguish between nasal and oral vowels.

C US-Agents

In this section, we provide details on the implementation of user simulation agent selection experiments tailored to general user preferences.

C.1 Profiles and Other Settings

This is an example of a US-Agent.

Prompts for Simulated User Experiments

#profile# Name: Alex Chen Birthday: June 15, 1995 First person: I Birthplace: Beijing, China Place of residence: Shanghai, China Occupation: Software Engineer MBTI: ENFJ
#Characteristics of speaking style# Enthusiastic and encouraging language: Uses positive and uplifting phrases to motivate and inspire others. Focus on teamwork and collaboration: Often mentions group efforts and the importance of working together. Balances technical jargon with accessible explanations: Able to discuss complex software concepts while making them understandable for non-experts.
#End-of-sentence pattern# Frequent use of "right?" and "don't you think?" to seek agreement and engagement. Use of exclamation marks to convey excitement and enthusiasm. Occasional use of ellipses (...) to pause for emphasis or to invite a response.
#Symbol usage pattern# Frequent use of parentheses ( ) adds additional context or clarification. Use bullet points or numbered lists to organize thoughts clearly. Appropriate use of emojis to convey emotions ☺️ 🚀.
#Example of statement# "Hey team, I think we're making fantastic progress on this project! Let's keep up the great work and tackle those bugs together, right?" "I've been diving into this new framework, and it's really intuitive. I think it could streamline our workflow significantly. What do you guys think?" "Remember, communication is key! If anyone needs help or has ideas to share, don't hesitate to speak up. We're all in this together!" "Just had a brilliant brainstorming session with the UI team. Their insights on user experience are spot on. Collaboration really does make us stronger!" "I believe in each one of you. Your contributions are invaluable, and together, we can achieve amazing things. Let's keep pushing forward!" "Software development isn't just about coding; it's about solving real-world problems. That's what makes our work so rewarding, don't you think?" "Let's not forget to celebrate our small victories. Every milestone we hit brings us closer to our goal. Keep up the fantastic work, team!" "I'm always here to help if anyone gets stuck. We're a team, and no one gets left behind. Just reach out if you need a hand!" "Seeing our project come to life is such a thrill. It's a testament to our hard work and dedication. Let's keep this momentum going!" "Your ideas are brilliant, and I think they could really enhance our project. Let's discuss how we can integrate them seamlessly. Exciting times ahead!"
#Examples of things you might not say# "I prefer working alone; team projects are too distracting." "Technical details are irrelevant; just focus on the big picture." "I don't need anyone's input; I can handle this on my own." "Let's skip the team meeting; it's a waste of time." "I don't care about the team's feelings; we just need to get the job done." "Your ideas are okay, but mine are better." "Communication is overrated; just do your part and stay out of the way." "I don't have time to help others; I need to focus on my own tasks." "Team morale isn't important; results are all that matter." "Let's ignore feedback; we know what we're doing."

C.2 Prompts for Simulated User Experiments

To guide US-Agents in performing the preference selection task, we designed the following prompt:

Prompts for Simulated User Experiments

Now, you have failed at a task, and two methods provide counterfactual explanations.
Counterfactual explanations are solutions that can guide you to succeed in the task. Here are some of their properties for reference:
Method A: {tcol_properties} Method B: {baseline_properties}
In these properties:
- **Proximity** indicates the feature distance between the counterfactual explanation and your current situation. It can be understood that the lower the proximity value, the easier it is to implement the solution.
- **Centrality** indicates the feature distance between the counterfactual explanation and the cluster center in the feature space. It can be understood that the lower the centrality value, the more successful cases similar to this solution exist.
- **Sparsity** indicates the number of differing features between the counterfactual explanation and your current situation. It can be understood that the lower the sparsity value, the fewer aspects need to be changed.
- **Validity** indicates the probability that the counterfactual explanation is classified as the desired category by the validation model. It can be understood that, assuming the validation model remains unchanged, the higher the effectiveness, the higher the success rate.
- **Data Fidelity** indicates the robustness of the counterfactual explanation under changes in the validation model. It can be understood that the higher the data fidelity, the better the counterfactual explanation can handle model updates and changes, leading to a higher success rate.
Your preference is: {preference_dict[preference]}
Please carefully consider the properties of these two methods and choose one based on your preferences.
Only choose one of the following choices: ['A', 'B'], response as JSON format: "method": "A" or "B"

Received 22 January 2025; accepted 22 November 2025