Studies on Explainable Machine Learning Based on Integer Linear Optimization

（整数計画法に基づく説明可能な機械学習）

Kentaro Kanamori

Graduate School of Information Science and Technology

Hokkaido University

January 2022

Abstract

This thesis studies new approaches to realizing the explainability of machine learning. For the last decade, due to the remarkable progress of machine learning technologies, complex machine learning models, such as deep neural networks, have been ubiquitous in society. They have performed well in various prediction tasks and assisted human users with their critical decision-making tasks in the real world, including medical diagnoses, loan approvals, and judicial decisions. In such critical decision-making tasks, decisions based on the predictions of machine learning models might have a significant impact on human users. Consequently, there have been urgent demands to develop methods for improving not only their accuracy but also their trustworthiness in recent years. An important key to improving the trustworthiness of machine learning models is to realize explainability, which means the ability to provide explanations of the predictions made by the models with human users in understandable terms. Therefore, studies on explainable machine learning have attracted increasing attention.

Existing approaches of explainable machine learning can be divided into two groups: post-hoc methods for extracting local explanations of the individual predictions from complex models, and methods for learning interpretable models that can provide explanations of their predictions by themselves. Among these existing methods, in this thesis, we focus on Counterfactual Explanation (CE) and decision trees, which are

popular methods for explainable machine learning. While CE is a local explanation method that provides perturbation vectors as "actions" for obtaining desired prediction results from models, a decision tree is an interpretable model that consists of "if-then-else" prediction rules expressed as a binary tree. Since both CE and decision trees are expected to help humans obtain better insights from the models and understand the given predictions with confidence, they have been actively studied in recent years.

However, concerns about the practicality of the existing methods for explainability have been raised from various perspectives. For example, although perturbations given by CE are interpreted by human users as actions for obtaining their desired decisions, existing CE methods often fail to provide plausible actions in the sense that the users can actually execute them. Another example is the stability of interpretation of decision trees during their deployment; that is, explanations given by decision trees are often unnecessarily significantly changed when they are retrained under additional constraints, such as fairness. To address these practicality issues, it is essential to explicitly clarify desiderata that explanations should satisfy by exploring causes of the issues and to design mathematical methods for satisfying them. Therefore, (i) identifying concrete desiderata of explanations from their practicality issues, (ii) formulating the tasks of realizing explainability as optimization problems by mathematically designing adequate evaluation criteria for the desiderata, and (iii) developing optimization methods for the formulated problems are important challenges on explainable machine learning.

The aim of this study is to develop new practical frameworks of explainable machine learning. For that purpose, (i) we first identify concrete desiderata of CE and decision trees by exploring their practicality issues. Then, (ii) we introduce new optimization problems for satisfying the desiderata by adequately designing objective functions and constraints. Finally, (iii) we propose flexible and efficient optimization methods for

them based on mixed-integer linear optimization (MILO), which is a sophisticated mathematical modeling framework for solving complex optimization tasks. In this thesis, we make three technical contributions corresponding to CE and decision trees.

In Chapter 4, we propose a new framework of CE, named Distribution-Aware Counterfactual Explanation (DACE), that provides realistic actions by taking the characteristics corresponding to the underlying data distribution into account. To suggest realistic actions that human users can execute, it is desirable to evaluate the reality of actions based on the characteristics of each prediction task, such as feature-correlations and outlier risks over the underlying data distribution. For that purpose, we introduce a new cost function as an objective function of CE that evaluates actions based on the characteristics of the underlying data distribution. Our cost function consists of the Mahalanobis distance and local outlier factor, which are a popular metric taking feature-correlations into account and a prominent score for outlier detection, respectively. Then, we propose an MILO formulation for extracting an optimal action that minimizes our cost function. For computational efficiency, we also propose a surrogate objective function that approximates our cost function, and show that it can dramatically reduce the number of variables and constraints of our MILO formulation.

In Chapter 5, we propose another new framework of CE, named Ordered Counterfactual Explanation (OrdCE), that provides not only an action but also an order of changing its features. While existing CE frameworks provide users with perturbations as actions, in practice, the required efforts of the actions depend on the order of changing the features because there are often asymmetric feature interactions, such as causal effects. Therefore, it is desirable to show an appropriate order of changing the features in addition to a perturbation. To suggest an appropriate order, we introduce a new optimization problem of CE by designing an ordering cost function that evaluates

a pair of a perturbation and its order, called an "ordered action." Our ordering cost function adequately measures the required effort of ordered actions based on given prior knowledge of the feature interactions, such as structural causal models, and determines a reasonable order of changing the features. Then, we propose an MILO formulation for extracting an optimal ordered action that minimizes our cost function.

In Chapter 6, we propose a new learning problem of decision trees, named Fairness-Aware Decision tree Editing (FADE), for stabilizing their prediction and interpretation under fairness constraints. Fairness of machine learning models has become an important perspective in critical decision-making. If decision-makers detect unfairness in their models during deployment, they must retrain their models under fairness constraints. However, the differences in prediction and interpretation between the initial and retrained models may cause serious reliability issues. Therefore, it is desirable to modify only those parts of their prediction and interpretation that are essentially necessary to satisfy the constraints. For that purpose, we formulate a new learning task for editing a given unfair decision tree into a fair one without significantly changing its prediction and interpretation. We introduce two dissimilarity measures between decision trees based on the prediction discrepancy and edit distance as objective functions. Then, we propose an MILO formulation for obtaining an optimal decision tree that minimizes the dissimilarity with a given decision tree under the fairness constraints.

In summary, we (i) elucidate three desiderata of CE and decision trees by exploring their practicality issues, (ii) introduce new optimization problems by mathematically modeling the desiderata as objective functions and constraints, and (iii) propose efficient and flexible optimization methods for them based on MILO. Through this study, we contribute to a first step for developing new practical frameworks of explainable machine learning that are applicable to critical decision-making tasks in the real world.

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor, Prof. Hiroki Arimura, who supervised me from the beginning of my research activity. He gave me a lot of insightful advice not only about my research but also about how to behave as a researcher. I feel really fortunate to have had his guidance. I would also like to thank Prof. Masaharu Yoshioka, Prof. Takashi Horiyama, and Prof. Atsuyoshi Nakamura for their comments on this thesis as referees. If I did not receive their useful advice about my research, I probably would not have been able to finish writing this thesis.

I would like to thank Satoshi Hara, Masakazu Ishihata, Takuya Takagi, Ken Kobayashi, Yuichi Ike, and Kento Uemura, for their helpful discussions and insightful comments through the joint work with them. They always helped me when I had trouble with my research and writing papers. I learned a lot of things not only about how to write papers but also about the fun of research from their kind support. I would also like to express my appreciation to the members of Artificial Intelligence Laboratories of Fujitsu Limited, especially Kotaro Ohori and Hirokazu Anai, for their support and encouragement during my internship. Our papers and this thesis have improved very much with their kind support.

I am deeply grateful to the past and present members of our laboratory. In particular, I would like to thank Ms. Yu Manabe, Ms. Yukie Watanabe, and Ms. Sachiko

Soma, who are secretaries at our laboratory, for helping me with my paperwork and business trip arrangements. My colleagues at our laboratory always encouraged me and supported my research activity. Kazuhiro Kurita and Isamu Furuya kindly advised me on my research activities. If they had not taught me the fun of research, I would not have gone on to a doctoral course. Ye Wang, Yuta Matsuda, and Kota Mata often took their time to discuss the recent research topics on machine learning with me. Naoya Toriyabe, who has been assigned to our laboratory in the same year as me, always helped me with my laboratory life until he graduated. Thanks to them, my laboratory life has been very exciting and meaningful.

I have received the financial support of Research Fellowship for Young Scientists (DC1) by Japan Society for the Promotion of Science (JSPS). All the work in this thesis was supported in part by JSPS KAKENHI Grant-in-Aid for JSPS Research Fellow 20J20654. I would like to deeply appreciate their financial support.

Finally, I would like to thank my friends and my family for their kind support. I would be grateful to my significant other, Haruka, for her support and encouragement for more than five years. Last but not least, I would like to express my special thanks to my parents for giving me the chance to study computer science and supporting my student life at Hokkaido University.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

For the last decade, due to the remarkable progress of machine learning technologies, complex machine learning models, such as *deep neural networks* and *tree ensembles*, have been ubiquitous in society. They have achieved high prediction performances in various decision-making tasks in the real world, including medical diagnoses [94, 358], financial decisions [97, 255], and judicial decisions [24, 369]. As a consequence, there has been an increasing trend to leverage machine learning models for such critical decision-making tasks, which is called *algorithmic decision-making* [119].

In the process of algorithmic decision-making for the above critical tasks, the predictions made by models might have a significant impact on human users who are subjects of the decisions. Consequently, problems concerning the *trustworthiness* [286,338] of machine learning models have been increasingly attracting attention. The term "trustworthiness" includes various perspectives, such as *fairness* [29, 264], *accountability* [184, 297], and *transparency* [208, 357]. In fact, decision-makers often recog-

nize that using complex models for their tasks is risky in terms of their trustworthiness [60, 161, 356]. Furthermore, there are several moves to legally regulate the use of complex machine learning models for critical decision-making tasks (e.g., "General Data Protection Regulation (GDPR)" by European Union [119]). Therefore, there are urgent demands for methods that improve not only prediction accuracy but also trustworthiness in the sense of fairness, accountability, and transparency.

To improve the trustworthiness of machine learning models, an important key challenge is to realize *explainability*, which means the ability to provide explanations of the predictions made by models with human users in understandable terms [1, 121]. Such explanations help human users (1) discover novel knowledge or overlooked bugs contained in the models [1, 183], and (2) accept the given decisions based on the predictions with confidence [85, 121]. With these advantages, they are expected to be useful for ensuring fairness, accountability, and transparency in the development and deployment of machine learning models. Consequently, studies on *explainable machine learning* have become an emerging topic in recent years, and a number of methods for realizing explainability have been proposed[1].

Existing methods to realize explainability roughly can be categorized into two major approaches [286]. One major approach is the post-hoc method that extracts *local explanations* of the individual predictions from complex models. To help human users understand why a certain prediction result is given by a model, local explanation methods provide some understandable factors that are most relevant to the prediction, such as important features [223,280] or influential training examples [189]. Among these methods, *Counterfactual Explanation (CE)* [335, 344] is a popular local explanation method that provides a perturbation vector of features for altering the prediction result

[1] A brief overview of existing work on explainable machine learning will be presented in Chapter 3.

into the desired result. Because human users can interpret the perturbation, i.e., how to change features, as an "action" for altering their decision result, they can understand not only why certain predictions are given but also how they should act to obtain their desired prediction results from the model. Therefore, CE can provide human users with better insights on their decision-making tasks [233, 237].

Another major approach to explainability is learning globally *interpretable models* instead of complex models. Interpretable models, such as sparse linear models [325] and rule models [49, 211, 283], are designed to be easy for humans to understand how the models make predictions. Therefore, they can provide explanations for their predictions by themselves, unlike complex models with a large number of parameters [121, 356]. For example, a *decision tree* [49, 275] is a traditional rule model that consists of a set of interpretable "if-then-else" rules for prediction and expresses them as a form of a binary tree structure. Due to their interpretability, decision trees have attracted increasing attention in recent years, and there have been several studies not only on learning accurate decision trees that are comparable to complex models [38, 149], but also on learning under some constraints other than accuracy, such as fairness [8, 167].

While both approaches to realizing explainability have been massively studied, concerns about their practicality have been pointed out for the last few years from various perspectives [2, 12, 18, 113, 286]. One major issue is the *plausibility* of local explanations. For example, although perturbations given by CE are interpreted by human users as actions for obtaining their desired decisions, existing CE methods often fail to provide plausible actions in the sense that the users can actually execute them [202, 339]. Another issue is the *stability* of interpretable models; that is, both the predictions and explanations given by interpretable models are often unnecessarily significantly changed during the deployment of them [48, 122]. These issues suggest the risk of

providing impractical explanations that prevent human users from obtaining better insights from models and accepting the predictions with confidence [42, 161]. To apply the existing methods for explainability to the process of algorithmic decision-making, addressing the issues with their plausibility and stability is essential [286].

A natural way to address these practicality issues is to define criteria that quantitatively evaluate explanations by taking the issues into account and to formulate tasks of realizing explainability as optimization problems with the criteria. Namely, it is essential to explicitly clarify "*desiderata*" that explanations should satisfy by exploring causes of the issues and to design mathematical methods for satisfying the identified desiderata [121, 258]. For the past two decades, researchers have been discussing several desiderata of explanations from various perspectives, including not only machine learning [99, 261] but also philosophy [237] and social science [233]. However, explicitly clarifying all the desiderata and mathematically expressing them are generally difficult because they are often elusive and depend on each situation [85, 217, 356]. Therefore, (i) identifying concrete desiderata of explanations from their practicality issues, (ii) formulating optimization tasks of realizing explainability by mathematically designing adequate evaluation criteria for the desiderata, and (iii) developing optimization methods for the tasks are important challenges on explainable machine learning.

1.2 Research Goals

The aim of this study is to develop new practical frameworks of explainable machine learning that are applicable to algorithmic decision-making. For that purpose, (i) we identify concrete desiderata of explanations by exploring the issues with the practicality of existing methods for explainability. Then, (ii) we introduce new optimization prob-

lems for satisfying the identified desiderata by adequately designing objective functions and constraints. Furthermore, (iii) we propose efficient optimization methods for them based on *mixed-integer linear optimization (MILO)*, which is one of the most sophisticated mathematical modeling frameworks [70, 359]. Due to its flexibility, MILO has been widely recognized as a reasonable solution for the optimization tasks regarding explainable machine learning in recent years [38, 331]. By incorporating several modeling techniques, we develop efficient optimization methods based on MILO.

1.3 Contributions

In this thesis, we make three technical contributions corresponding to explainable machine learning. We focus on CE and decision trees, which are popular methods for realizing explainability, and study methods for improving their practicality. Each technical contribution of this thesis can be summarized as follows:

- In Chapter 4, we propose a new framework of CE, named *Distribution-Aware Counterfactual Explanation (DACE)*, that provides realistic actions by taking the characteristics corresponding to the underlying data distribution into account. Existing CE methods often fail to provide realistic actions that human users can directly refer to and execute. This is mainly because they do not sufficiently take the characteristics of each prediction task, such as feature-correlations and outlier risks over the underlying data distribution [30, 202, 259]. Therefore, a desideratum for providing realistic actions is to evaluate the reality of actions based on the characteristics of the underlying data distribution. To satisfy the desideratum, we introduce a new cost function as an objective function of CE that evaluates actions based on the Mahalanobis distance [227] and local outlier

factor [52], which are a popular metric taking feature-correlations into account and a prominent score for outlier detection, respectively. Then, we propose an MILO formulation for extracting an optimal action that minimizes our cost function. For computational efficiency, we also propose a surrogate objective function that approximates our cost function, and show that it can dramatically reduce the number of variables and constraints of our MILO formulation[2].

- In Chapter 5, we propose another new framework of CE, named *Ordered Counterfactual Explanation (OrdCE)*, that provides not only an action but also an order of changing its features. Existing CE frameworks provide users with perturbation vectors as actions for obtaining their desired prediction results. In practice, however, the required effort to execute the action depends on the order of changing the features because there are often asymmetric interactions among features, such as causal effects [177, 178, 270]. Therefore, it is desirable to show an appropriate order of changing the features in addition to a perturbation vector as an action. To suggest an appropriate order, we introduce a new optimization problem of CE by designing an ordering cost function that evaluates a pair of an action and its executing order, which we refer to as an "ordered action," based on the feature interactions. Given prior knowledge of the feature interactions, such as structural causal models estimated from the dataset [262, 303], our cost function adequately measures the required effort of ordered actions and determines a reasonable order of changing the features. Then, we propose an MILO formulation for extracting an optimal ordered action that minimizes our cost function[3].

- In Chapter 6, we propose a new learning problem of decision trees, named

[2]This result has been published in [171, 172]

[3]This result has been published in [174]

Fairness-Aware Decision tree Editing (FADE), for stabilizing their prediction and interpretation under fairness constraints. As with explainability, fairness of machine learning models has been recognized as an important perspective in critical decision-making [127, 264]. If decision-makers detect unfairness in their models during deployment, they must retrain their models under fairness constraints. However, simply retraining a model from scratch tends to result in unnecessarily significant differences in both the prediction and interpretation between the initial and retrained models [48, 122]. Such differences can cause serious reliability issues, such as reverse discrimination [229, 332] and malicious rationalization of unfairness [12, 13]. Therefore, when the deployed models are retrained, modifying only those parts of their prediction and interpretation that are essentially necessary to satisfy fairness constraints is desirable. For that purpose, we formulate a new learning task for editing a given unfair decision tree into a fair decision tree without significantly changing its prediction and interpretation. We introduce two dissimilarity measures between decision trees based on the prediction discrepancy [229] and edit distance [322, 372] as objective functions. Then, we propose an MILO formulation for obtaining an optimal decision tree that minimizes the dissimilarity with a given decision tree under the fairness constraints[4].

In summary, we (i) elucidate concrete desiderata of CE and decision trees by exploring their practicality issues, (ii) introduce new optimization problems for satisfying the identified desiderata by adequately designing objective functions and constraints, and (iii) propose efficient and flexible optimization methods for them based on MILO. Through this study, we contribute to the first step of developing new practical frameworks of explainable machine learning for algorithmic decision-making in the real world.

[4]This result has been published in [169]

1.4 Organization

The remainder of this thesis is organized as follows. In Chapter 2, we provide basic definitions and notations to be used in this thesis. We also provide a brief overview of recent studies on explainable machine learning in Chapter 3. In Chapter 4, we present DACE, which is a new framework of CE for providing realistic actions by taking the feature-correlations and outlier risks into account. In Chapter 5, we present OrdCE, which is another new framework of CE for providing not only actions but also an appropriate execution order of its features. In Chapter 6, we present FADE, which is a new learning framework for stabilizing the prediction and interpretation of decision trees under fairness constraints. Finally, we conclude this thesis, and then, discuss the future direction in Chapter 7.

Chapter 2

Preliminaries

In this chapter, we provide basic notations and definitions to be used in the following chapters. Almost all the notations and definitions in this thesis follow standard textbooks in machine learning [43, 139, 238, 239, 301].

2.1 Basic Notations

For a positive integer $n \in \mathbb{N}$, we write $[n] := \{1, \ldots, n\}$. 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. For a real number $x \in \mathbb{R}$, the sign function $\mathrm{sgn} : \mathbb{R} \to \{-1, +1\}$ is defined as $\mathrm{sgn}(x) = 1$ if $x \geq 0$, and $\mathrm{sgn}(x) = -1$ if $x < 0$. For a D-dimensional vector $x = (x_1, \ldots, x_D) \in \mathbb{R}^D$ and real number $p \geq 1$, the ℓ_p-norm $\|x\|_p$ of x is defined as $\|x\|_p := \left(\sum_{d=1}^{D} |x_d|^p\right)^{\frac{1}{p}}$. For $p = 0$, we define $\|x\|_0 := \sum_{d=1}^{D} \mathbb{I}[x_d \neq 0]$, which is call the ℓ_0-norm of x.

2.2 Supervised Learning

In this thesis, we focus on machine learning models for *supervised learning problems*. Given an input domain $\mathcal{X}$, an output domain $\mathcal{Y}$, and a set of functions $\mathcal{F} \subseteq \{f \mid f\colon \mathcal{X} \to \mathcal{Y}\}$, the aim of supervised learning is to find a function $f \in \mathcal{F}$ that minimizes the *expected prediction risk* $R\colon \mathcal{F} \to \mathbb{R}_{\geq 0}$ defined as follows:

$$R(f) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[l(x, y; f)\right],$$

where $\mathcal{D}$ is a joint distribution over the input and output domains $\mathcal{X} \times \mathcal{Y}$, and $l\colon \mathcal{X} \times \mathcal{Y} \times \mathcal{F} \to \mathbb{R}_{\geq 0}$ is a *loss function* that measures the difference between the prediction $f(x)$ and the true output y. For example, $l(x, y; f) = (y - f(x))^2$ is the *squared loss* for *regression problems* ($\mathcal{Y} = \mathbb{R}$), and $l(x, y; f) = \mathbb{I}\left[y \neq f(x)\right]$ is the 0–1 *loss* for *classification problems* (e.g., $\mathcal{Y} = \{-1, +1\}$). We call a function $f\colon \mathcal{X} \to \mathcal{Y}$ a *machine learning model* (or *model*, for short), and a set $\mathcal{F}$ a *model class*[1]. For a classification (resp. regression) problem, we also call a model f a *classifier* (resp. *regressor*).

Unfortunately, it is impossible to directly compute the expected prediction risk $R(f)$ in practice because we cannot observe the true distribution $\mathcal{D}$. Instead, we assume a *sample* $S = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$ with $N \in \mathbb{N}$ *labeled examples* $(x, y) \in \mathcal{X} \times \mathcal{Y}$ that are independently and identically distributed to $\mathcal{D}$. We call $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ an *input instance* and *output label*, respectively. Given a sample S, the task of supervised learning is defined as the following optimization problem:

$$\hat{f} \quad = \quad \operatorname*{arg\,min}_{f\in\mathcal{F}} \quad \hat{R}(f \mid S) + \Omega(f). \tag{2.1}$$

The first term $\hat{R}$ of the objective function of the problem (2.1) is the *empirical risk*

[1]In the literature of computational learning theory, a model f and a model class $\mathcal{F}$ is called a *hypothesis* and a *hypothesis set*, respectively [238]

defined as

$$\hat{R}(f \mid S) := \frac{1}{N} \sum_{(x,y) \in S} l(x, y; f). \tag{2.2}$$

The second term $\Omega \colon \mathcal{F} \to \mathbb{R}_{\geq 0}$ is a *regularization* term that evaluates how complex a model f is and avoids overfitting of f on S. We call the problem (2.1) the *regularized empirical risk minimization* [238].

Hereinafter, we assume an input domain $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_D \subseteq \mathbb{R}^D$ for a positive integer $D \in \mathbb{N}$, and an output domain $\mathcal{Y} \subseteq \mathbb{R}$. We denote an input instance $x \in \mathcal{X}$ by $x = (x_1, \ldots, x_D) \in \mathcal{X}$, and call each dimension d a *feature*.

2.3 Model Class

In this thesis, we mainly focus on a model class $\mathcal{F}$ based on the *linear basis expansion model* [139]. A linear basis expansion model $f \colon \mathcal{X} \to \mathcal{Y}$ is expressed as the following additive form:

$$f(x) = \sum_{t=1}^{T} w_t \cdot f_t(x) - b, \tag{2.3}$$

where $f_1, \ldots, f_T \colon \mathcal{X} \to \mathbb{R}$ are the *base learners*[2], $T \in \mathbb{N}$ is the total number of base learners, $w_t \in \mathbb{R}$ is a weight value of t-th base learner f_t for $t \in [T]$, and $b \in \mathbb{R}$ is an intercept. While the linear basis expansion models can be directly applied to regression problems ($\mathcal{Y} = \mathbb{R}$), they can be also applied to classification problems (e.g., $\mathcal{Y} = \{-1, +1\}$) by adequately transforming their outputs, which will be shown in Section 2.4. In the following, we show that several well-known machine learning models can be expressed as the linear basis expansion models.

[2]Generally, f_t is also called a *basis function* [139].

2.3.1 Linear Models

Linear models (LMs), such as the logistic regression and linear support vector machines, are one of the most popular models [139]. An LM makes a prediction depending on the inter product $\langle w, x \rangle$, i.e.,

$$f(x) = \sum_{d=1}^{D} w_d \cdot x_d - b, \tag{2.4}$$

where $w = (w_1, \ldots, w_D) \in \mathbb{R}^D$ is a coefficient vector and $b \in \mathbb{R}$ is an intercept. We note that an LM is a special case of the linear basis expansion models such that $T = D$ and $f_d(x) = x_d$ for $d \in [D]$.

Fortunately, learning problems of LMs $(w, b) \in \mathbb{R}^{D+1}$ can be often solved efficiently. For a regression problem $\mathcal{Y} = \mathbb{R}$, for example, let the loss function l and regularization term Ω be the squared loss and squared ℓ_2-norm of w, respectively. Then, an optimal solution (w^*, b^*) can be computed analytically, which is known as the *Ridge regression* [139]. Another well-known example is the linear support vector machines (SVMs) for a binary classification problem $\mathcal{Y} = \{-1, +1\}$, whose loss function is the *hinge loss* defined as $l(x, y; f) = \max\{0, 1 - y \cdot f(x)\}$. Because the learning problem of SVMs is a convex quadratic optimization problem, it can be efficiently solved by several algorithms, such as sequential minimal optimization (SMO) [266]. Generally, if l and Ω is differentiable and convex with respect to (w, b), linear models can be efficiently learned by gradient-based optimization methods, such as gradient descent, coordinate descent, and stochastic gradient descent [118].

2.3.2 Decision Trees

A *decision tree* is a model consisting of a set of exclusive if-then-else rules expressed in the form of a binary tree [49, 275]. It makes a prediction according to the predictive

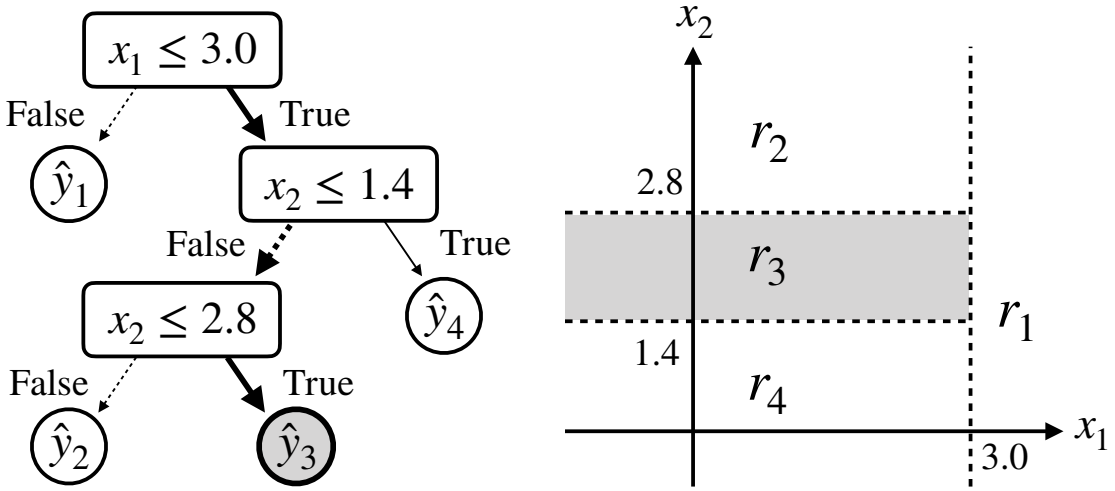


Figure 2.1: An example of a decision tree and its partition on the input space $\mathcal{X} = \mathbb{R}^2$.

label $\hat{y} \in \mathcal{Y}$ of the leaf that an input instance $x \in \mathcal{X}$ reaches. The corresponding leaf is determined by traversing the tree from the root depending on whether the statement $x_d \leq b$ is true or not, where $(d, b) \in [D] \times \mathbb{R}$ is a pair of a branching feature and a branching threshold of each internal node.

A decision tree with L leaves represents a partition $r_1, \ldots, r_L$ of the input domain $\mathcal{X}$, i.e., $\bigcup_{l=1}^{L} r_l = \mathcal{X}$ and $r_l \cap r_k = \emptyset$ for $l, k \in [L]$ [139]. Then, a decision tree $f \colon \mathcal{X} \to \mathcal{Y}$ can be expressed as

$$f(x) = \sum_{l=1}^{L} \hat{y}_l \cdot \mathbb{I}\left[x \in r_l\right], \tag{2.5}$$

where $L \in \mathbb{N}$ is the total number of leaves in the decision tree f, and $\hat{y}_l \in \mathcal{Y}$ and $r_l \subseteq \mathcal{X}$ is a predictive label and region corresponding to the leaf $l \in [L]$, respectively. We note that a decision tree is a special case of the linear basis expansion models such that $T = L$, $w_l = \hat{y}_l$, and $f_l(x) = \mathbb{I}\left[x \in r_l\right]$ for $l \in [L]$. Figure 2.1 presents an example of a decision tree and its partition on the input space $\mathcal{X}$.

Algorithm 1 Top-down greedy algorithm for learning decision trees [49, 275].

Input: A training sample $S \subset \mathcal{X} \times \mathcal{Y}$, and an impurity criterion i (e.g., Gini index for classification or variance for regression).

Output: A decision tree $f \colon \mathcal{X} \to \mathcal{Y}$

function REC(S)
  if a given stopping criterion met **then**
    initialize a leaf node n;
    if classification **then**
      optimize a predictive label $\hat{y}^* \leftarrow \arg\max_{y' \in \mathcal{Y}} \sum_{(x,y) \in S} \mathbb{I}[y = y']$;
    else if regression **then**
      optimize a predictive label $\hat{y}^* \leftarrow \frac{1}{|S|} \sum_{(x,y) \in S} y$;
    end if
    return a leaf node n with $\hat{y}^*$;
  else
    initialize an internal node n;
    optimize a branching rule (d^*, b^*) by the following optimization problem:

$$(d^*, b^*) \leftarrow \operatorname*{arg\,max}_{(d,b) \in [D] \times \mathbb{R}} \left(i(S) - \left(\frac{|S_0|}{|S|} \cdot i(S_0(d,b)) + \frac{|S_1|}{|S|} \cdot i(S_1(d,b)) \right) \right),$$

    where $S_0(d,b) = \{(x,y) \in S \mid x_d > b\}$ and $S_1(d,b) = \{(x,y) \in S \mid x_d \leq b\}$;
    $n.left \leftarrow$ REC($S_0(d^*, b^*)$); $n.right \leftarrow$ REC($S_1(d^*, b^*)$);
    return the internal node n with (d^*, b^*);
  end if
end function
initialize a root node r;
$r \leftarrow$ REC(S);
return a decision tree f with the root node r;

Unfortunately, because decision trees are not differentiable due to their discrete nature, gradient-based optimization methods cannot be directly applied, unlike linear models. Moreover, a problem for finding a minimum-sized decision tree that minimizes the empirical risk is an NP-hard problem [151,368]. Therefore, top-down greedy learning algorithms, such as the *Classification And Regression Trees (CART)* [49] and *C4.5* [275], have been widely used. Algorithm 1 presents a general framework of a top-down greedy algorithm for learning decision trees. Given a training sample S, Algorithm 1 learns a branching rule $x_d \leq b$ for each internal node by recursively partitioning S depending on a given impurity criterion i, such as the Gini index $i(S) = 1 - \sum_{y' \in \mathcal{Y}} p_{y'}^2$ for classification or variance $i(S) = \frac{1}{|S|} \sum_{(x,y) \in S} (\bar{y} - y)^2$ for regression, where $p_{y'} = \frac{1}{|S|} \sum_{(x,y) \in S} \mathbb{I}[y = y']$ and $\bar{y} = \frac{1}{|S|} \sum_{(x,y) \in S} y$, respectively [49, 139]. If a given stopping condition, such as minimum sample size or maximum tree depth, is satisfied, then the algorithm generates a leaf with a predictive label $\hat{y}$ optimized for a subset of the sample S that reaches the leaf.

In contrast, recently, several methods for exactly minimizing the empirical risk and size of decision trees based on modern optimization techniques, such as branch-and-bound algorithms and mixed-integer linear optimization, have been proposed [38, 149]. The recent work on learning *optimal decision trees* will be described in Chapter 3.

2.3.3 Tree Ensembles

Tree ensembles (TEs), such as the random forest [47] and gradient boosting decision tree [68, 101, 181, 272], are known for their high prediction performances in machine learning competitions. Each base learner f_t is a decision tree, and thus a TE f makes a prediction by combining the prediction results of T decision trees. Since a decision

tree can be expressed as (2.5), a TE f can be expressed as follows:

$$f(x) = \sum_{t=1}^{T} w_t \cdot \sum_{l=1}^{L_t} \hat{y}_{t,l} \cdot \mathbb{I}\left[x \in r_{t,l}\right], \tag{2.6}$$

where $L_t \in \mathbb{N}$ is the total number of leaves in t-th decision tree f_t, and $\hat{y}_{t,l} \in \mathcal{Y}$ and $r_{t,l} \subseteq \mathcal{X}$ is a predictive label and region corresponding to the leaf $l \in [L_t]$ of f_t, respectively.

A key to improving the performance of ensemble models is to average many models that have high variance but low bias [46]. Because decision trees are known to have high variance and low bias if they are sufficiently deep, they are ideal candidates for ensemble models [139]. One of the most popular TEs is the *random forest* [47] and its variants, such as the *extremely randomized trees (ExtraTrees)* [112]. The random forest is a substantial modification of *bootstrap aggregation (bagging)* [46]. The learning algorithm of random forests builds a set of diverse decision trees by subsampling not only examples but also features. Algorithm 2 presents a generic algorithm for learning random forests. In each iteration t of the algorithm, a decision tree f_t is trained with a bootstrap sample $S_t \subseteq S$ and a randomly selected features $P \subseteq [D]$. Each of the decision trees is trained independently, and the random forest f makes predictions by averaging them, i.e., $w_t = \frac{1}{T}$ for any $t \in [T]$.

Another popular TE is the *gradient boosting decision tree* [101]. Unlike random forests where each decision tree is independently trained and equally averaged, the gradient boosting decision tree sequentially optimizes a decision tree f_t and its weight w_t in each step t so as to compensate the weaknesses of decision trees $f_1, \dots f_{t-1}$ build in previous steps. Algorithm 2 presents a generic algorithm for learning gradient boosting decision trees. Let $f^{(t)} = \sum_{t'=1}^{t} w_{t'} \cdot f_{t'}(x)$ be a model trained before t-th iteration. In each iteration t of the algorithm, a decision tree f_t is trained so as to fit

Algorithm 2 Algorithm for learning random forests [47].

Input: A training sample $S \subset \mathcal{X} \times \mathcal{Y}$, a total number of decision trees $T \in \mathbb{N}$, and subsample sizes for examples $m \in [N]$ and features $p \in [D]$.

Output: A random forest $f \colon \mathcal{X} \to \mathcal{Y}$.

1: **for** $t = 1, \ldots, T$ **do**
2: draw a bootstrap sample S_t with m examples from S;
3: randomly select p features $P \subseteq [D]$ from $[D]$;
4: train t-th decision tree f_t with the sample S_t and features P (e.g., Algorithm 1);
5: **end for**
6: **return** a random forest f with decision trees $f_1, \ldots, f_T$;

Algorithm 3 Algorithm for learning gradient boosting decision trees [101].

Input: A training sample $S \subset \mathcal{X} \times \mathcal{Y}$, and a total number of decision trees $T \in \mathbb{N}$.

Output: A gradient boosting decision tree $f \colon \mathcal{X} \to \mathcal{Y}$.

1: **for** $t = 1, \ldots, T$ **do**
2: compute gradients $g(x, y) = -\frac{\partial l(x, y; f^{(t-1)})}{\partial f^{(t-1)}(x)}$ for $(x, y) \in S$;
3: construct a new sample $S_t \leftarrow \{(x, g(x, y)) \mid (x, y) \in S\}$ with the gradients;
4: train t-th decision tree f_t with the sample S_t (e.g., Algorithm 1);
5: compute t-th weight $w_t \leftarrow \arg\min_{w \in \mathbb{R}} \hat{R}(f_w^{(t)} \mid S)$, where $f_w^{(t)}(x) = f^{(t-1)}(x) + w \cdot f_t(x)$;
6: **end for**
7: **return** a gradient boosting decision tree f with decision trees $f_1, \ldots, f_T$ and their weights $w_1, \ldots, w_T$;

not the labels y but the gradients $\frac{\partial l(x,y;f^{(t-1)})}{\partial f^{(t-1)}(x)}$ of the loss of the previous model $f^{(t-1)}$, which is called generalized residuals [139]. Algorithms for learning gradient boosting decision trees are implemented as several libraries (e.g., XGBoost [68], LightGBM [181], and CatBoost [272]). These libraries are renowned for their fastness and prediction performances in practical machine learning competitions such as Kaggle[3].

2.3.4 Multilayer Perceptrons

Multilayer perceptrons (MLPs) have become increasingly common over the past decade due to the remarkable progress of *deep neural networks (DNNs)* [118]. For simplicity, we consider two-layer ReLU networks, i.e., MLPs with a single hidden layer and the *rectified linear unit (ReLU)* function $g(x) = \max(0, x)$ as an activation function[4]. With MLPs, each base learner f_t is an output of a neuron in its hidden layers with the ReLU function g, i.e., $f_t(x) = g(w^{(t)} \cdot x + b^{(t)})$, where $w^{(t)} \in \mathbb{R}^D$ and $b^{(t)} \in \mathbb{R}$ are weight values and an intercept with respect to the t-th neuron in the hidden layer, respectively. Then, an MLP f can be expressed as

$$f(x) = \sum_{t=1}^{T} w_t \cdot g(w^{(t)} \cdot x + b^{(t)}). \tag{2.7}$$

Generally, because learning problems of MLPs are non-convex optimization problems, efficiently finding optimal solutions is computationally challenging. However, unlike decision trees and tree ensembles, the learning objective of MLPs is differentiable, and thus efficient gradient-based optimization methods can be used. Let $f_\theta \colon \mathcal{X} \to \mathcal{Y}$ be an MLP with a parameter $\theta \in \Theta$ including w_t and $w^{(t)}$ for $t \in [T]$, where Θ is a parameter space. Gradient-based methods iteratively update the parameter θ depending

[3] `https://www.kaggle.com/`

[4] Our proposed methods presented in Chapters 4 and 5 can be extended to general multilayer ReLU networks.

Algorithm 4 Stochastic gradient descent algorithm for learning MLPs [118].

Input: A training sample $S \subset \mathcal{X} \times \mathcal{Y}$, a set of learning rate $\{\varepsilon_1, \varepsilon_2, \ldots\}$, a size of minibatch $m \in [N]$, and an initial parameter $\theta \in \Theta$.

Output: An MLP $f \colon \mathcal{X} \to \mathcal{Y}$.

initialize a parameter $\theta^* \leftarrow \theta$;
$k \leftarrow 1$;
while a given stopping criterion not met **do**
    sample a minibatch $S_k \subseteq S$ with m examples from S;
    compute gradient estimates $\hat{g}_k \leftarrow \nabla_{\theta^*} \hat{R}(f_{\theta^*} \mid S_k)$;
    update the parameter $\theta^* \leftarrow \theta - \varepsilon_k \cdot \hat{g}_k$;
    $k \leftarrow k + 1$;
end while
return an MLP f_{θ^*} with the parameter θ^*;

on the gradients of the empirical risk $\hat{R}$ with respect to θ. One of the popular learning algorithms is the *stochastic gradient descent* [118] and its variants. Algorithm 4 presents a generic algorithm of the stochastic gradient descent. In each iteration k of Algorithm 4, the model parameter θ^* is updated depending on a given learning rate ε_k and estimated gradients $\hat{g}_k$ by a minibatch $S_k \subseteq S$ with m examples sampled from a given training sample S. Although a set of learning rate $\{\varepsilon_1, \varepsilon_2, \ldots\}$ is a hyperparameter that we must determine in advance, it is known to significantly affect the performance of learned models. To set learning rates adequately, several strategies for adaptively determining a learning rate ε_k of each iteration k during training have been proposed, such as the RMSProp [118] or Adam [188].

2.4 Binary Classification

In the contributions of this thesis, i.e., in Chapters 4 to 6, we consider *binary classification problems* as prediction tasks. Unless otherwise stated, we assume $\mathcal{Y} = \{-1, +1\}$. To adapt to the output domain of a binary classification problem $\mathcal{Y} = \{-1, +1\}$, we modify a linear basis expansion model $f \colon \mathcal{X} \to \mathcal{Y}$ so that it makes a prediction through the sign function, i.e.,

$$f(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} w_t \cdot f_t(x) - b\right). \tag{2.8}$$

We call f an *additive classifier (AC)*.

2.5 Mixed-Integer Linear Optimization

In this thesis, we formulate new optimization problems for realizing the explainability of machine learning. As optimization methods for the formulated problems, we focus

on *mixed-integer linear optimization (MILO)*. MILO is one of the most sophisticated mathematical modeling frameworks for solving complex optimization tasks [70, 359]. Generally, a key of MILO-based approaches is to express a given optimization task as a (binary) *mixed-integer linear optimization problem (MILO problem)* defined as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{I} c_i \cdot x_i + \sum_{j=1}^{J} h_j \cdot y_j \\ \text{subject to} \quad & \sum_{i=1}^{I} a_{p,i} \cdot x_i + \sum_{j=1}^{J} g_{p,j} \cdot y_j \leq b_p, \quad \forall p \in \{1, \ldots, P\}, \\ & x_i \in \{0, 1\}, \qquad\qquad \forall i \in \{1, \ldots, I\}, \\ & y_j \in \mathbb{R}, \qquad\qquad \forall j \in \{1, \ldots, J\}, \end{aligned} \tag{2.9}$$

where $x_i \in \{0, 1\}$ and $y_j \in \mathbb{R}$ are variables to be optimized, $c_i, h_j, a_{p,i}, g_{p,j}, b_p \in \mathbb{R}$ are constants computed in advance, $I, J, P \in \mathbb{N}$ are the total number of binary variables, real-valued variables, and constraints, respectively. While the MILO problem (2.9) is NP-hard, several algorithmic techniques for efficiently solving the MILO problem (2.9) have been massively studied and developed until today [70, 359]. Hence, once we formulate a given optimization task as a form of the MILO problem (2.9), we have a chance to efficiently obtain an optimal solution for the original optimization task by solving the formulated MILO problem (2.9).

One key advantage of the MILO-based approaches is their modeling flexibility [70, 359]. Using the integer variables $x_i \in \{0, 1\}$, we can exactly express discrete structures (e.g., the tree structure of decision trees, constraints on categorical features, and non-differentiable objective functions) and handle them as constraints and objective functions. In addition, once the optimization task is formulated as an MILO problem, it can be efficiently solved by off-the-shelf MILO solvers, such as CPLEX [153] and Gurobi [125]. Over the past 30 years, remarkable progress of computer hardware and algorithmic techniques has allowed modern MILO solvers to handle large optimization

problems including real-world applications [38, 331]. Taking advantage of their flexibility and efficiency, MILO-based approaches can efficiently deal with a wide range of objective functions and constraints without implementing designated algorithms, which is a desirable property for explainable machine learning [59, 76, 257, 288, 333, 335, 341].

On the other hand, the scalability of the MILO solvers mainly depends on the size of MILO problems, i.e., the total numbers of variables $I+J$ and constraints P. To alleviate the issues with scalability, several modeling techniques for efficient MILO formulation have been massively studied [359]. By incorporating these modeling techniques, we attempt to efficiently formulate the optimization tasks for realizing the explainability of machine learning models.

Chapter 3

Overview of Explainable Machine Learning

In this chapter, we present a brief overview of studies on explainable machine learning. We provide a survey on major existing methods for realizing explainability, and also discuss their issues that have been raised from the perspectives of their practicality. Finally, we clarify how the contributions of this thesis relate to the previous studies.

3.1 Background

Due to the remarkable progress of machine learning techniques, complex machine learning models, such as deep neural networks (DNNs) [118] and tree ensembles [47,68,181], have performed well in various prediction tasks for the last decade. As a consequence, there has been an increasing trend to leverage machine learning models for decision-making tasks in the real world, which is called *algorithmic decision-making* [119]. Actually, they have already achieved high prediction performances in various real decision-

making tasks, including:

- *Medicine*: assessing the risk of disease (e.g., diabetes [254], sleep apnea [336], and cardiovascular [358]), health care [34, 94], and medical imaging [93, 370].

- *Finance*: assessing the risk of default in loan approvals or credit scoring [97, 255, 291], fraud detection [141], and portfolio management [142].

- *Criminal justice*: assessing the risk of recidivism or flight of offenders [24, 51, 327], and setting bail amounts [369].

In the above decision-making tasks, decisions are often *critical* for humans; that is, decisions based on the predictions of the models might have a significant impact on human users [69]. Consequently, problems concerning the *trustworthiness* of machine learning models have been increasingly attracting attention [286, 338]. To emphasize the need for the trustworthiness of machine learning models, several regulations, such as the "General Data Protection Regulation (GDPR)" by European Union [119], "Ethically Aligned Design" by IEEE [324], and "Ethics guidelines for trustworthy AI" by European Commission [143], have been released in recent years. For example, in the "Social Principles of Human-Centric AI" released by the Cabinet Secretariat of Japan in March 2019 [54], they mentioned *fairness* [29, 127, 264], *accountability* [86, 121, 184, 297], *transparency* [85, 208, 357], and *robustness* [15, 321] as perspectives that should be considered in the development and deployment of machine learning models. Under these regulations, decision-makers must follow them to use machine learning models for their decision-making tasks [119]. Therefore, there have been urgent demands to develop methods for improving the trustworthiness of machine learning models in the sense of fairness, accountability, transparency, and robustness [286, 338].

In particular, *explainability* has been recognized as a key to improving the trustworthiness of machine learning models in the research community, and studies on *explainable machine learning* have been attracting increasing attention [1,121]. While the term "explainable machine learning" is relatively new, issues with explainability have been traditionally studied in various fields of artificial intelligence [337], such as expert systems [240] and data mining [261], since the mid-1970s. In the light of the technical advances in machine learning models and their applications to the critical decision-making tasks, research papers on explainable machine learning have been exponentially increased for the last five years [1], and various methods for realizing explainability, which we will refer to as "*explanation methods*," have been proposed until today. Furthermore, to provide a better understanding of the models and their predictions, desiderata that explanations should satisfy have been actively discussed from various perspectives, including not only machine learning [30,85,99,217,356], but also social science [208,233], cognitive science [20, 237], law [119, 297], privacy [14, 234, 304], human-computer interaction [21, 157, 179, 196, 346], and real deployment [40, 42, 271, 329, 339]. Fortunately, there have been several exhaustive survey papers [1, 31, 61, 87, 115, 121, 215, 287] and excellent textbooks [43, 239] on explainable machine learning.

To improve the trustworthiness of machine learning models, explanation methods are expected to play important roles to (1) discover novel knowledge or overlooked bugs contained in machine learning models [1, 183], and (2) assist human users to accept the predictions given by the models with confidence [121]. More specifically, they are expected to be useful both for identifying the causes of issues with fairness or robustness and for ensuring accountability and transparency in the development and deployment of machine learning models. In contrast to such expectations, several limitations regarding the practicality of existing methods have been pointed out for

the last few years from various perspectives, such as *plausibility* [128, 131, 202, 251], *faithfulness* [2,158], and *stability* [18,48,122]. These studies have revealed not only the risk of providing misleading explanations that prevent human users from understanding the models accurately [83, 113, 286], but also the risk of malicious manipulation of explanations, e.g., for rationalizing the discrimination of models [12, 71, 105, 129]. Therefore, existing explanation methods are insufficient to fulfill their expected roles, and there are still difficulties to apply them in real decision-making tasks [286]. In fact, although several papers conducted user experiments to investigate whether the existing explanation methods actually help humans, there are still a few papers that have fully concluded their effectiveness [17, 137, 161, 163, 269]. In real deployments of machine learning models, Bhatt et al. have reported that the existing explanation methods are only used by model developers for debugging their models, and are not used to explain the predictions to the human users who are subjects of the prediction [40, 42, 179].

This chapter provides a brief overview of studies on explainable machine learning by elaborately selecting some important papers published recently. We review two major approaches of explanation methods: (1) *post-hoc local explanation* and (2) *learning globally interpretable models*, respectively. Tables 3.1 and 3.2 summarize existing representative methods for each approach. Then, we also present some practicality issues with them that have been raised for the last few years. Finally, we conclude this chapter by clarifying the relation of the contributions in this thesis to the previous studies.

3.2 Post-hoc Local Explanation

A recent major approach for realizing explainability is the post-hoc extraction of local explanations from learned complex models. Post-hoc approaches for extracting knowl-

Table 3.1: Existing representative methods for post-hoc local explanation.

Name	**Explanation Form**	**Explanation Type**	**Procedure**
LIME [280]	Feature-based	Local surrogate	Approximating complex models locally by linear models
SHAP [223]	Feature-based	Feature attribution	Scoring each feature based on the Shapley value
Grad-CAM [298]	Feature-based	Saliency map	Scoring each feature by averaging its weighted gradients
Influence [189]	Example-based	Influential example	Scoring each example based on the influence function
Fisher Kernel [182]	Example-based	Similar example	Evaluating the relevance of examples by the Fisher kernel
CE [344]	Counterfactual	Perturbation	Optimizing perturbations for desired prediction results
AR [335]	Counterfactual	Perturbation	Optimizing perturbations under actionability-constraints

Table 3.2: Existing representative methods for learning globally interpretable models.

Name	**Model Class**	**Learning Problem**	**Optimization**
Lasso [325]	Linear model	Minimizing the squared loss and ℓ_1-norm	e.g., coordinate descent
SLIM [333]	Linear model	Minimizing the 0–1 loss and ℓ_0-norm under integer-constraints	MILO
CART [49]	Decision tree	Recursive partitioning based on the Gini index	Top-down greedy algorithm
OCT [38]	Decision tree	Minimizing the 0–1 loss and total number of nodes	MILO
OSDT [149]	Decision tree	Minimizing the 0–1 loss and total number of leaves	Branch-and-bound algorithm
CORELS [23]	Rule list	Minimizing the 0–1 loss and total length of rules	Branch-and-bound algorithm
IDS [198]	Rule set	Maximizing accuracy under cardinality-constraints on rules	Submodular optimization

edge or explanation have been widely studied, such as rule extraction from complex models [50, 74, 75, 130, 231, 244], knowledge distillation of complex models [103, 106, 145, 373], scoring the relevance of features by permutation importance [16, 47, 206], feature selection with statistical testing [116, 117, 126, 293], and representative example selection [44, 114, 185]. These approaches are designed for extracting explanations from an entire model $f\colon \mathcal{X} \to \mathcal{Y}$ or training sample $S \subset \mathcal{X} \times \mathcal{Y}$. While these methods attempt reveal "global" behaviour of f, recent post-hoc methods attempt to extract "local" explanations of the prediction result $f(x^\circ)$ for a given specific input instance $x^\circ \in \mathcal{X}$.

A main advantage of the post-hoc local explanation methods is that they can provide explanations of prediction results $f(x^\circ)$ while maintaining the high prediction performances of the complex models f [1]. Therefore, various methods for post-hoc local explanation have been actively studied in recent years. Key challenges of post-hoc local explanation methods are (i) what form of explanation to provide, and (ii) what criterion to use for evaluating the explanation. Existing post-hoc local explanation methods can be divided into three groups depending on their forms of explanation [239]: (1) feature-based, (2) example-based, and (3) counterfactual explanation.

3.2.1 Feature-based Explanation

Feature-based explanation methods, also known as *instance-wise feature selection* methods, provide a score of each feature $d \in [D]$ for how much the feature d contributes to the prediction result $f(x^\circ)$ of a complex model f for a given specific instance x°. By quantitatively evaluating the contribution of each feature to prediction results, we can identify on which features the model f outputs the specific prediction result $f(x^\circ)$ for a given individual instance x°. It helps to verify the validity of the predictions by f (e.g., whether its prediction is based on sensitive features such as gender or race [4, 264]).

One of the major feature-based explanation methods is the *local surrogate* method that learns an interpretable model for locally approximating a complex model f near a given instance x°. Because interpretable models can explain how they make predictions using each feature by themselves, we can understand the contribution of each feature to the original prediction result $f(x^\circ)$ through them. The most popular framework is *Locally Interpretable Model-agnostic Explanation (LIME)* [280] and its variants [124, 164, 197, 268, 281, 289, 313, 375]. As its explanation, LIME provides a surrogate interpretable model h that locally approximates the behavior of a complex model f near a given instance x°. Given an input instance $x^\circ \in \mathcal{X}$ and complex model $f\colon \mathcal{X} \to \mathcal{Y}$, the framework of LIME can be roughly formulated as the following optimization problem:

$$h^* = \operatorname*{arg\,min}_{h \in \mathcal{H}} \hat{R}_{x^\circ}(h \mid f) + \Omega(h), \tag{3.1}$$

where $\mathcal{H}$ is a class of surrogate models $h\colon \mathcal{X} \to \mathcal{Y}$, and $\Omega\colon \mathcal{H} \to \mathbb{R}_{\geq 0}$ is a regularization term that evaluates the complexity of h. The first term $\hat{R}_{x^\circ}\colon \mathcal{H} \to \mathbb{R}_{\geq 0}$ is the weighted empirical risk defined as:

$$\hat{R}_{x^\circ}(h \mid f) := \frac{1}{|N(x^\circ)|} \sum_{x \in N(x^\circ)} \Delta_{x^\circ}(x) \cdot l(x, f(x); h), \tag{3.2}$$

where $N(x^\circ) \subset \mathcal{X}$ is a set of neighbor instances of x°, and $\Delta_{x^\circ}(x) = \Delta(x^\circ, x)$ is the distance between x° and x based on some metric $\Delta\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ on $\mathcal{X}$. In recent literature, $\hat{R}_{x^\circ}$ is known as the *infidelity* score for evaluating the quality of surrogate models, i.e., local explanations [158, 203, 361]. A typical choice of $\mathcal{H}$ and Ω are linear models $h(x) = w^\top x + b$ and the ℓ_1-norm $\|w\|_1$ of w, respectively. We can interpret each value w_d of an obtained coefficient vector $w = (w_1, \dots w_D)$ as a contribution score of the feature $d \in [D]$ to the prediction result $f(x^\circ)$. Figure 3.1 presents an example of

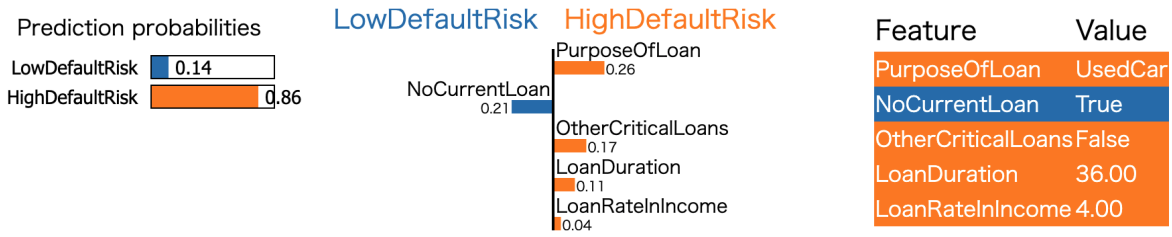


Figure 3.1: An example of an explanation provided by LIME [280].

an explanation extracted by LIME from a LightGBM classifier [181] on the German dataset [88]. Note that a class of local surrogate models can be extended to other interpretable models, e.g., a set of predictive rules with respect to features, which is called *Anchors* [281].

Another major approach is the *feature attribution* method that evaluates the contribution of each feature by some appropriate criterion. One of the most sophisticated frameworks is *SHapley Additive exPlanations (SHAP)* [223] and its variants [67, 72, 192, 222, 350]. In the framework of SHAP, a contribution score of each feature is evaluated based on the Shapley values in cooperative game theory [302]. Lundberg and Lee have theoretically shown that the scores based on the Shapley values satisfy some desirable properties in interpreting the prediction $f(x)$, e.g., local accuracy, missingness, and consistency [223]. Moreover, as with LIME, SHAP is a model-agnostic method; that is, the scores of SHAP can be calculated for any machine learning model f.

For differentiable models such as DNNs, the *saliency maps* [308] are also popular feature attribution methods to visualize the features that are focused by a complex model f. These methods evaluate the contribution of each feature d based on its input gradient $\frac{\partial f(x)}{\partial x_d}$ of $f(x^\circ)$ with respect to x°_d. The saliency maps are popular methods mainly for image classification tasks because they can visualize a set of important features, i.e., pixels, by highlighting them in the input image x°, which is easy for humans to under-

stand. To remove noises in gradients and sharpen the highlights, several methods have been proposed in recent years, such as GuidedBackprop [317], SmoothGrad [314], IntegratedGrad [320], Epsilon-LRP [27], Grad-CAM [298], and DeepLIFT [305]. Recently, a few recent studies have attempted to theoretically analyze the relationships between the saliency maps and other model-agnostic methods, such as LIME and SHAP [6,223].

3.2.2 Example-based Explanation

In contrast to the feature-based explanation methods that evaluate the contribution of each feature $d \in [D]$ to a specific prediction result $f(x^\circ)$, *example-based explanation* methods, also known as *instance-based* or *similarity-based explanation* methods, evaluate that of each example in the training sample S for f. Roughly speaking, example-based methods provide one or more training examples $(x, y) \in S$ similar to x° as its explanation of the prediction result $f(x^\circ)$. This type of explanation is known to be preferable to humans because it is analogous to the way humans make decisions by referring to their prior experiences [128, 163].

To evaluate the relevance of training examples $z = (x, y) \in S$ with respect to a given instance x° and its prediction $f(x^\circ)$ by f, several criteria, such as the *influence function* [32,123,189], *Fisher kernel* [182], and inner product between input gradients [362], have been proposed. The main idea of these criteria is to evaluate the influence of a training example z on the prediction result $f(x^\circ)$ by approximating the difference between the prediction results for x° by the model trained with z and that of without z. Taking advantage of this property, example-based explanation methods are applied to data cleansing tasks for identifying a subset of the training examples S that increases the loss of the trained model [135, 273]. As a variant of example-based explanation methods, a concept-based explanation method that provides abstract concepts, such

as stripe or dot patterns, rather than a specific example has been proposed [186].

3.2.3 Counterfactual Explanation

Both feature-based and example-based explanations provide humans with reasons why a machine learning model f output the prediction result $f(x^\circ)$ for a given specific instance x°. *Counterfactual Explanation (CE)* [344], also known as *Actionable Recourse (AR)* [335], is a post-hoc local explanation method that provides humans with not only why the prediction result $f(x^\circ)$ was obtained, but also how the prediction result $f(x^\circ)$ could be changed into their desired prediction result, such as "low risk of diabetes" or "approval of the loan." From the perspectives of social science, such an explanation of the "what-if" form, i.e., *counterfactual* [210], is known to be motivated by how humans ask questions; that is, humans tend to ask "why event P happened instead of some event Q," rather than "why event P happened" [216, 233, 237, 356]. Therefore, CE has attracted increasing attention as a new framework that can provide more constructive and preferable explanations for human users [176, 318, 340].

Generally, the task of CE can be formulated as the following optimization problem:

$$a^* := \underset{a \in \mathcal{A}}{\arg\min} \quad C(a \mid x^\circ) \quad \text{subject to} \quad f(x^\circ + a) = y^*, \tag{3.3}$$

where $\mathcal{A} \subset \mathbb{R}^D$ is a set of feasible perturbations for x°, $C\colon \mathcal{A} \to \mathbb{R}_{\geq 0}$ is a cost function that evaluates the required effort of a perturbation $a \in \mathcal{A}$ for x°, and $y^* \in \mathcal{Y}$ is a desired prediction result. Since a perturbation a^* alters the prediction result by a model f into the desired one y^*, humans x° can interpret a^* as an "*action*" for obtaining their desired prediction result from the model f. A few examples of extracted actions by CE are shown in Table 4.1 of Chapter 4.

CE can be seen as having aspects of both feature-based and example-based expla-

nations in the sense that it provides a feature vector a^* and a counter-example $x^\circ + a^*$ as its explanation. On the other hand, unlike existing feature-based and example-based explanation methods, CE aims to provide "actionable" explanations in the sense that humans can directly refer to its explanation and execute it as an action [233]. For that purpose, it is necessary that the obtained perturbation vector a^* can be executed by humans, unlike *adversarial examples* [15, 321]. Therefore, the set of feasible actions $\mathcal{A}$ must be constructed by adequately taking the properties of each feature $d \in [D]$ into account [30, 339]. For example, Ustun et al. have introduced two types of features that should be constrained in the feasible actions: (i) immutable features (e.g., age or marital status) that should not be considered actionable, and (ii) conditionally immutable features (e.g., education level) that should be actionable in only one direction, i.e., only increasing or decreasing [335].

Key challenges of CE are (1) how to adequately design a cost function C and (2) how to efficiently solve the optimization problem (3.3). To adequately evaluate the required effort of actions for humans, several useful cost functions, such as weighted ℓ_1-norm of *median absolute deviation* [292, 344], *total-log percentile shift* [335], and ℓ_2-norm based on *structural causal models* [226], have been proposed. Optimization frameworks of existing CE methods can be roughly categorized as gradient-based [178, 241, 242, 330, 344], autoencoder [81,226,259], SAT [175], heuristic search [200,270,277,328], or mixed-integer linear optimization (MILO) [76,257,292,335]. A few survey papers present more detailed recent advances on CE [176, 318, 340].

3.2.4 Advanced Methods for Local Explanation

Amortized Explanation. Almost all the existing methods for post-hoc local explanation formulate their task of extracting local explanations as some optimization

problem, such as the problem (3.1) or (3.3). However, given multiple instances, we need to solve the optimization problem for each instance, which often requires a large computational cost. To resolve this issue, frameworks for the *amortized explanation* have been proposed [66, 77, 162, 187, 296, 363]. For a complex machine learning model f, the amortized explanation methods learn an "explainer" function $e\colon \mathcal{X} \to \mathcal{E}$, which maps an input instance $x^\circ \in \mathcal{X}$ into an explanation $e(x^\circ) \in \mathcal{E}$ for the prediction result $f(x^\circ)$ (e.g., D-dimensional vector $e(x^\circ) \in \mathbb{R}^D$ whose d-th element represents the contribution of the feature d to $f(x^\circ)$), from a training sample based on some criteria, such as mutual information [66]. Once an explainer e is trained, we can quickly obtain explanations for unseen instances without solving optimization problems.

Global Summary of Local Explanations. Explanations extracted from a complex model f are often expected to help humans understand the behavior of f. However, existing post-hoc local explanation methods, such as LIME [280], are designed for a given specific instance x°, and it is unclear whether the obtained explanation can be applied to other instances. Thus, it is difficult for humans to estimate the prediction results of f for unseen instances by the obtained explanations. To resolve this difficulty, frameworks for globally summarizing local explanations have been proposed [41, 108, 173, 212, 263, 278, 279]. These methods can be seen as a framework that constructs an explainer function $e\colon \mathcal{X} \to \mathcal{E}$ expressed as an interpretable form, such as hierarchical clustering [278] or rule models [108, 173, 279]. Because these interpretable forms provide human users with information on the regions where a specific explanation can be applied, they help humans understand the behavior of f on the entire input domain $\mathcal{X}$. For example, Kanamori et al. have proposed *Counterfactual Explanation Tree (CET)*, which is a framework of CE that summarizes actions over the entire input

space $\mathcal{X}$ by a decision tree [173]. Taking advantage of decision trees, CET can assign a unique action to any input instance over $\mathcal{X}$ in an interpretable manner.

3.2.5 Plausibility Issues with Local Explanation

As shown above, various post-hoc methods for extracting local explanations have been proposed in recent years. This has led to growing demands for methods to compare the explanations provided by several different methods [161]. To compare different methods, several evaluation criteria [128, 201, 202] and sanity checking methods [2, 3, 147] that validate whether the existing methods can provide *plausible* [131, 158, 251] explanations for human users have been proposed.

Unfortunately, these comparison methods have revealed the issue with *faithfulness* [22, 158]; that is, existing methods often provide implausible local explanations that do not actually reflect how models make predictions. For example, Adebayo et al. have proposed the *model parameter randomization test* that compares explanations extracted from a given complex model f with explanations extracted from a model whose parameters are randomly perturbed from f [2]. They have observed that explanations extracted by several existing methods, such as the GuidedBackprop [317] and GuidedGradCAM [298], were invariant before and after perturbing, which indicates that the extracted explanations did not reflect the models f.

Another issue with existing post-hoc local explanation methods is *vulnerability*. To obtain valid insights on how a complex model makes predictions, we expect that a local explanation extracted for an instance x° is robust to perturbations of the instance x°. In other words, it is desirable to provide the same explanation for instances in the neighborhood of x°. Based on this desideratum, Alvarez-Melis and Jaakkola have proposed a criterion for evaluating the robustness of post-hoc local explanation meth-

ods based on the Lipchitz condition [18]. Given an instance $x^\circ \in \mathcal{X}$ and a post-hoc local explanation method $e\colon \mathcal{X} \to \mathcal{E}$, where $\mathcal{E}$ be a set of possible explanations, the vulnerability of e on x° is defined as

$$v_{x^\circ}(e) := \max_{x \in B_\varepsilon(x^\circ)} \frac{\Delta_{\mathrm{E}}(e(x^\circ), e(x))}{\Delta(x^\circ, x)}, \tag{3.4}$$

where $B_\varepsilon(x^\circ) = \{x \in \mathcal{X} \mid \Delta(x^\circ, x) \leq \varepsilon\}$ is a set of perturbed instances with radius $\varepsilon > 0$, $\Delta\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $\Delta_{\mathrm{E}}\colon \mathcal{E} \times \mathcal{E} \to \mathbb{R}_{\geq 0}$ are dissimilarity measures for instances and explanations, respectively. In their experiments, they have observed that some of the existing feature-based explanation methods, such as LIME [280] and SHAP [223], tend to be vulnerable to small perturbations. Similar results have also been observed for example-based explanation methods [33]. Furthermore, by taking advantage of the vulnerability, several empirical methods for adversarially manipulating explanations have been also proposed [82, 113, 311, 312], like *adversarial examples* that manipulate predictions [15, 321].

To conclude, these results have revealed that the existing post-hoc local explanation methods (1) often fail to provide plausible explanations that faithfully reflect how complex models make predictions, and (2) might be adversarially manipulated due to their vulnerability. These facts imply that there are reliability risks in the sense that explanations not only become meaningless for human users, but also lead them to mislead [286]. To elucidate these issues, recently, there have been several studies on theoretical analyses of the existing methods from various perspectives, such as gradients of the models [83, 148, 247, 310, 361], causalities between features [160, 193], the sensitivity of hyperparameters [110, 111], convergence [109, 228], out-of-distribution [138], or multiplicity [319]. However, concrete solutions for addressing these issues with post-hoc local explanation methods still remain important challenges [286].

3.3 Learning Globally Interpretable Models

Another line of work on explainable machine learning is learning globally interpretable models instead of complex models. Interpretable models, such as sparse linear models [325] and decision trees [49,275], have simple mathematical structures that humans easily understand how these models make predictions. Therefore, interpretable models can provide humans with explanations for their predictions by themselves, unlike complex models including DNNs and tree ensembles [121, 287, 356].

It has been well-known that there is a trade-off between the accuracy and interpretability of machine learning models [48]. Thus, a key challenge of learning interpretable models is to adjust the trade-off. For a given training sample $S \subset \mathcal{X} \times \mathcal{Y}$, almost all the existing methods attempt to adjust it by formulating the task of learning interpretable models as the following optimization problem [286]:

$$h^* = \arg\min_{h \in \mathcal{H}} \hat{R}(h \mid S) + \lambda \cdot \Omega(h), \tag{3.5}$$

where $\mathcal{H}$ is a class of interpretable models $h \colon \mathcal{X} \to \mathcal{Y}$, $R \colon \mathcal{H} \to \mathbb{R}_{\geq 0}$ is the empirical risk of h on S, $\Omega \colon \mathcal{H} \to \mathbb{R}_{\geq 0}$ is a regularizer that evaluates the complexity of h, and $\lambda > 0$ is a trade-off parameter. While Ω evaluates the interpretability of h for humans, its definition often varies depending on the model class $\mathcal{H}$ or application domains [85,217]. Hence, how to define Ω is still under discussion from the perspectives of not only theoretical analyses but also user experiments [107, 150, 194, 265].

Because $\mathcal{H}$ and Ω often contain discrete nature, the problem (3.5) tends to be a combinatorial optimization problem, and generally, finding optimal solutions for (3.5) is often computationally challenging [151, 286, 331, 368]. Due to its non-differentiable properties, gradient-based optimization algorithms, such as stochastic gradient descent, cannot be directly applied to the problem, unlike learning DNNs [118]. Thus, another

key challenge is to design an efficient optimization algorithm for the problem (3.5). Therefore, several efficient algorithms based on modern mathematical optimization techniques, such as branch-and-bound algorithms [10, 23, 149, 214], Bayesian optimization [347, 353, 360], and mixed-integer linear optimization (MILO) [8, 38, 333, 334, 341], have been massively studied in recent years.

3.3.1 Sparse Linear Models

One of the most popular interpretable models is a *sparse linear model*. As shown in Section 2.3.1, linear models $h_{(w,b)} \colon \mathcal{X} \to \mathcal{Y}$ makes a prediction depending on the value $\langle w, x \rangle - b$, where $\langle w, x \rangle$ is the inner product between an input instance $x \in \mathcal{X}$ and a coefficient vector $w \in \mathbb{R}^D$, and $b \in \mathbb{R}$ is an intercept. The term "sparse" means that the coefficient w has a few non-zero elements and that $w_d = 0$ holds for many $d \in [D]$. The sparsity of a model is known to be a reasonable measure of interpretability from the perspective of cognitive psychology [73, 232, 286]. From the definition of linear models, features d with $w_d = 0$ do not contribute to the prediction result, and thus we can identify a set of important features as $\mathrm{supp}(w) := \{d \in [D] \mid w_d \neq 0\}$.

A traditional sophisticated framework for learning sparse linear models is the *Least Absolute Shrinkage and Selection Operator (Lasso)* [325] and its variants, such as the ℓ_1-regularized logistic regression [207], Elastic Net [377], Fused Lasso [326], and Group Lasso [154, 307, 364]. For learning space linear models $(w, b) \in \mathbb{R}^{D+1}$, Lasso solves the following optimization problem:

$$(w^*, b^*) = \operatorname*{arg\,min}_{(w,b) \in \mathbb{R}^{D+1}} \hat{R}(h_{(w,b)} \mid S) + \lambda \cdot \|w\|_1. \tag{3.6}$$

Note that the regularizer Ω is set to the ℓ_1-norm of w, i.e., $\|w\|_1 = \sum_{d=1}^{D} |w_d|$. It is known to induce sparsity on the resultant coefficient w^* [139, 325]. Fortunately,

while the ℓ_1-norm is not differentiable at the origin, it has a subgradient. Thus, the problem (3.6) can be efficiently solved by several optimization algorithms, such as the coordinate descent [100] and Least-Angle Regression (LARS) [91], when the loss function l of the empirical risk $\hat{R}$ is convex (e.g., the squared loss for a regression problem, or the logistic loss for a classification problem).

Recently, a more interpretable variant of sparse linear models, named *Supersparse Linear Integer Model (SLIM)*, has been proposed [316,331,333,334,371]. Motivated by traditional scoring systems [331], a SLIM is designed as a sparse linear model whose coefficients w_d and intercept b are constrained to be a small integer value, i.e., $(w, b) \in \mathbb{Z}^{D+1}$. The task of learning a SLIM is formulated as follows:

$$(w^*, b^*) = \underset{(w,b)\in\mathcal{I}}{\arg\min}\, \hat{R}(h_{(w,b)} \mid S) + \lambda \cdot \|w\|_0, \tag{3.7}$$

where $\mathcal{I} = \mathcal{I}_1 \times \cdots \times \mathcal{I}_{D+1} \subset \mathbb{Z}^{D+1}$ is a set of small integer coefficients, e.g., $\mathcal{I}_d = \{-10, \ldots, 10\}$ for $d \in [D+1]$. Note that the regularizer Ω is set to the ℓ_0-norm of w, i.e., $\|w\|_0 = \sum_{d=1}^{D} \mathbb{I}\,[w_d \neq 0]$, which encourages the resultant coefficient w^* to be sparse more directly than the ℓ_1-norm of Lasso. Compared to ordinary sparse linear models, the SLIM can be easily understood and validated by humans because its prediction results can be calculated through simple arithmetic without a computer [316, 331]. However, due to the discrete nature of the model class $\mathcal{I}$ and the non-differentiable property of the ℓ_0-norm $\|w\|_0$, exactly solving the problem (3.7) is computationally challenging. To exactly solve this problem, Ustun and Rudin have proposed an efficient algorithm based on MILO [333, 334]. Table 3.3 presents an example of a SLIM trained on the German dataset [88].

Predict **"HighDefaultRisk"** if **SCORE** > -1.		
1.	LoanDuration ≤ 36	-2 points
2.	NoCurrentLoan = True	$+2$ points
3.	PurposeOfLoan = Repairs	$+2$ points
4.	PurposeOfLoan = UsedCar	-2 points
	Add points from 1–4	**SCORE**

Table 3.3: An example of a SLIM [333] trained on the German dataset [88].

3.3.2 Rule Models

Another popular interpretable model is a *rule model*. One of the standard rule models is a decision tree [49, 275]. As shown in Section 2.3.2, decision trees consist of a set of exclusive if-then-else rules expressed in the form of a binary tree. Given an input instance $x \in \mathcal{X}$, a decision tree makes a prediction according to the leaf that the instance x reaches. The corresponding leaf is determined by traversing the tree from the root depending on a branching rule, e.g., the statement $x_d \leq b$ with a feature $d \in [D]$ and threshold $b \in \mathbb{R}$. Such rule-based expression makes it easier for humans to understand how a model h makes predictions [121].

A problem for finding a minimum-sized decision tree $h\colon \mathcal{X} \to \mathcal{Y}$ minimizing the empirical risk $\hat{R}(h \mid S)$ on a given training sample $S \subset \mathcal{X} \times \mathcal{Y}$ is known as NP-hard [151,368]. Hence, top-down greedy learning algorithms, such as the *Classification And Regression Trees (CART)* [49] and *C4.5* [275], had been popular for a long time. Although there have been several non-greedy algorithms based on continuous relaxation techniques, such as the gradient descent [250, 367] and probabilistic modeling [53, 96, 104,252,349], these methods have the disadvantage that they reduce the interpretability

of decision trees by continuous relaxation of branching rules.

Recently, in contrast to them, exact methods for learning optimal decision trees have been increasingly attracting attention. Almost all of these methods formulate the task for learning decision trees as the following optimization problem:

$$h^* = \underset{h \in \mathcal{H}_{\mathrm{DT}}}{\arg\min}\, \hat{R}(h \mid S) + \lambda \cdot \Omega_{\mathrm{DT}}(h), \tag{3.8}$$

where $\mathcal{H}_{\mathrm{DT}}$ is a set of decision trees $h\colon \mathcal{X} \to \mathcal{Y}$, and Ω_{DT} is a regularization term for evaluating the complexity of decision trees, e.g, the total number of nodes [38] or leaves [149]. As mentioned above, the problem (3.8) is a combinatorial optimization problem, and finding its optimal solution is challenging. To solve this computationally difficult optimization problem, several efficient approaches based on MILO [8, 38, 341, 343, 376], branch-and-bound algorithms [10, 149, 214, 230], SAT [26, 245, 295], and other mathematical optimization techniques [9, 80] have been proposed. Major frameworks for learning optimal decision trees are *Optimal Classification Trees (OCT)* [38] and *Optimal Sparse Decision Trees (OSDT)* [149], which are based on MILO and branch-and-bound algorithms, respectively. These studies reported that optimal decision trees obtained by solving the problem (3.8) achieved high prediction accuracy comparable to state-of-the-art complex models, such as the random forests [47], in several benchmark tabular datasets. For more details of recent advances in learning optimal decision trees, there have been a few survey papers on them [59, 156]. Figure 3.2 presents a decision tree trained by *DL8.5* algorithm [10], which is based on branch-and-bound techniques, on the German dataset [88].

In addition to decision trees, rule models include rule lists [23, 65, 283, 288, 347, 360], rule sets [78, 155, 198, 211, 353], rule ensembles [35, 36, 102, 235], and generalized additive rule models [60, 62, 219, 249, 355]. Although problems for learning these models also tend to be combinatorial optimization problems that require a large computational

cost in general, several efficient learning methods based on mathematical optimization techniques, such as the Bayesian optimization [347, 353, 360] and submodular optimization [198], have been proposed. While Table 3.4 presents a rule list trained by *Certifiable Optimal RulE ListS (CORELS)* algorithm [23], which is based on branch-and-bound techniques, Table 3.5 present a rule set trained by *Interpretable Decision Set (IDS)* algorithm [198], which is based on submodular optimization.

It is also noteworthy that decision trees are also attracting attention from the perspectives of their high potential representability. Taking the advantage of their interpretability and representability, decision trees have been applied not only for imitating the entire behavior of complex models, such as DNNs [205] and tree ensembles [342], but also for other non-standard prediction problems motivated by actual decision-making tasks (see, e.g., [39, 92, 199, 306]). A novel framework for partially replacing the predictions of complex models with that of rule models, such as rule sets [352] and rule lists [256], has been also proposed. Moreover, the actual interpretability of the rule models has been investigated by user studies [107, 150, 194, 265].

3.3.3 Interpretable Neural Networks

Generally, interpretable models are often inferior to complex models, such as DNNs, for prediction performance. Because DNNs have high representability for modeling complex relationships between the input domain $\mathcal{X}$ and output domain $\mathcal{Y}$, they have the potential to achieve high prediction performances in several prediction tasks. However, it is difficult for humans to understand how DNNs make predictions due to their complex structure [121, 356]. To enhance the interpretability of DNNs while maintaining their prediction performances, two approaches for learning interpretable DNNs have been proposed in recent years.

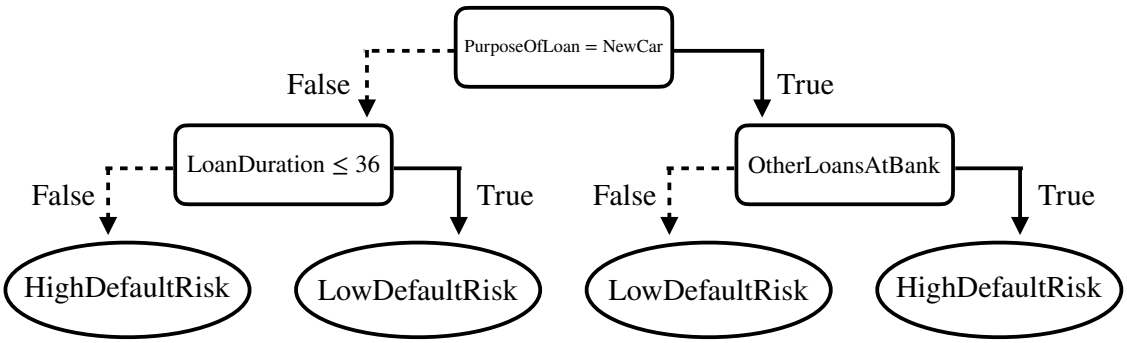


Figure 3.2: An example of a decision tree [10] trained on the German dataset [88].

	Rule	Prediction
if	NoCurrentLoan = True **and** PurposeOfLoan = NewCar	**"HighDefaultRisk"**
else if	PurposeOfLoan = UsedCar **and** YearsAtCurrentHome ≤ 3	**"LowDefaultRisk"**
else if	OwnsHouse = False **and** LoanDuration ≤ 24	**"HighDefaultRisk"**
else		**"LowDefaultRisk"**

Table 3.4: An example of a rule list [23] trained on the German dataset [88].

	Rule	Prediction
if	Age ≤ 29 **and** PurposeOfLoan = NewCar	**"HighDefaultRisk"**
if	OwnsHouse = False **and** LoanDuration > 24	**"HighDefaultRisk"**
if	Age ≤ 29 **and** LoanDuration > 30	**"HighDefaultRisk"**
else		**"LowDefaultRisk"**

Table 3.5: An example of a rule set [198] trained on the German dataset [88].

Interpretable Network Design. Several architectures of neural networks that are designed to be interpretable for humans have been proposed [5, 19, 25, 63, 64, 221, 253, 274, 345, 351, 354]. For example, *Two-layer Additive Risk Model* [64], which is a winner of the "FICO Explainable Machine Learning Challenge" competition [97], is a two-layer neural network that consists of multiple linear models[1]. It makes predictions by combining the output of each linear model transformed by a non-linear function. Each linear model uses only a distinct subset of features partitioned depending on the meanings of the features, and thus humans can understand how a specific set of features contributes to the final prediction results of the model. Another example is *TabNet* [25], which is a DNN designed for tabular datasets. TabNet can provide feature importance scores in both global and local levels like tree ensembles [180, 213, 220].

Interpretability Regularizer. To extract explanations from DNNs, post-hoc local explanation methods, such as LIME [280] or saliency maps [308] described in Section 3.2, are often used. However, explanations extracted by these methods sometimes conflict with domain knowledge or become unclear due to noise. To learn DNNs with accurate and clear explanations, a framework for regularizing explanations of DNNs has been proposed [95, 195, 204, 267, 282, 284, 285]. For any class of models $\mathcal{F}$, their framework can be formulated as follows:

$$f^* = \arg\min_{f \in \mathcal{F}} \hat{R}(f \mid S) + \lambda \cdot \Omega_{\mathrm{IR}}(f \mid S), \tag{3.9}$$

where Ω_{IR} is an *interpretability regularizer* [267], which regularizes the quality of local explanations extracted by post-hoc methods from a model f. The interpretability regularizer Ω_{IR} can be defined as various criteria, such as the infidelity $\hat{R}_x(h \mid f)$ of local surrogate models h [204, 267], absolute values of the input gradients of the saliency

[1] The demonstration of their model is available at `http://dukedatasciencefico.cs.duke.edu/`.

maps [95,284], or differences between domain knowledge and input gradients [282,285]. For example, the interpretability regularizer based on LIME is defined as follows [267]:

$$\Omega_{\mathrm{IR}}(f \mid S) := \frac{1}{N} \sum_{(x,y) \in S} \min_{h \in \mathcal{H}} \hat{R}_x(h \mid f). \tag{3.10}$$

It evaluates the average of the minimum infidelity of LIME over a given training sample S. By minimizing $\Omega_{\mathrm{IR}}(f \mid S)$, explanation extracted by LIME from a learned model f^* is expected to have low infidelity, i.e., to be high-quality local explanations.

3.3.4 Rashomon Effects in Interpretable Models

It has been known that there often exist multiple distinct models that are approximately equally accurate for the same prediction task. This phenomenon, named "*Rashomon effect*" by Breiman [48], has attracted much attention in recent years [13, 71, 84, 98, 129, 229, 286, 287, 299, 315]. While the Rashomon effect is related to some areas of traditional learning theory, such as flat minima [89, 144, 146] and algorithmic stability [45,218], it is a new practical perspective of machine learning models and their learning algorithms [286, 287].

The term "Rashomon" comes from the 1950 film directed by Akira Kurosawa, in which four characters describe the same crime in four different ways [79]. For a practical prediction task, similarly, there often exist multiple machine learning models that are approximately equally accurate but provide different explanations of their predictions. In fact, sparse linear models including Lasso have been known to often fail in recovering important features due to correlations between features [133, 323]. Because decision trees have high variance, a small change in a training sample often significantly changes the decision tree learned from the sample [47, 139]. These facts imply that finding a single optimal model that minimizes the empirical risk is often

insufficient for obtaining reliable insights on a prediction task [133].

The Rashomon effect can be regarded as either a positive or negative phenomenon. On one hand, human users have a chance to obtain multifaceted insights on their prediction task from a set of almost equally accurate models [286, 287]. For a model class $\mathcal{H}$, sample S, and offset value $\varepsilon \geq 0$, the *Rashomon set* $\hat{\mathcal{R}}_\varepsilon(\mathcal{H} \mid S)$ is defined as

$$\hat{\mathcal{R}}_\varepsilon(\mathcal{H} \mid S) := \{h \in \mathcal{H} \mid \hat{R}(h \mid S) \leq \hat{R}(h^* \mid S) + \varepsilon\}, \tag{3.11}$$

where $h^* \in \mathcal{H}$ is an optimal model that minimizes the empirical risk $\hat{R}$, i.e., $h^* = \arg\min_{h \in \mathcal{H}} \hat{R}(h \mid S)$. Although the models in $\hat{\mathcal{R}}_\varepsilon$ achieve similar accuracy, they often differ markedly in their predictions for individual inputs and thus may have different properties [129, 286, 287]. Consequently, characterizing the Rashomon set $\hat{\mathcal{R}}_\varepsilon$ can provide new insights on the reliability of a certain model class $\mathcal{H}$ on a specific prediction problem [286, 287]. Several criteria for characterizing the Rashomon set have been proposed from various perspectives, e.g., interpretability [84, 98, 315], the ratio of its volume to the model space [299], predictive multiplicity [229], and fairness [13, 71]. Unfortunately, exactly computing the Rashomon set $\hat{\mathcal{R}}_\varepsilon$ still remains challenging [286, 287]. However, an algorithmic framework of enumerating models with distinct interpretations in non-increasing order of their learning objective has been proposed for various models, such as the Losso [133, 134], decision trees [290], rule sets [132], and SVM [170]. Human users are expected to be able to choose their preferable model from the Rashomon set in the sense of consistency of its interpretation with their prior knowledge or some properties, e.g., sparsity, fairness, or robustness [286, 287].

On the other hand, the Rashomon effect can be also considered as the issue with the *stability* of interpretable models [35, 36, 90, 122, 323]. For example, Guidotti and Ruggieri have empirically analyzed the stability of the interpretations of interpretable models, and observed that there have been several accurate decision trees with various

interpretations for a single prediction task [122]. Similar concerns have been also discussed in the context of feature selection [140, 165, 248]. Furthermore, recently, some studies have revealed the reliability issues with the multiplicity of explanations caused by the Rashomon effect [12, 13, 129]. Aïvodji et al. have proposed a framework, named *fairwashing*, for making an unfair complex model seems fair by enumerating interpretable surrogate models that locally approximate the behavior of the complex model and satisfy fairness constraints [12]. In their experiments, they have succeeded in learning rule lists that approximate given complex models well and satisfy fairness constraints, although the complex models were unfair. These results indicate the risk that undesirable behavior of machine learning models is rationalized by explanations provided by interpretable models intentionally chosen from the Rashomon set [13, 71, 129]. Therefore, to apply interpretable models to actual decision-making tasks, taking the Rashomon effect in them into account and addressing their stability issue have been recognized as important challenges in recent years [286, 287].

3.4 Conclusion

In this chapter, we presented a brief review of recent techniques for explainable machine learning. We described two major approaches to explainability, i.e., post-hoc local explanation methods and learning globally interpretable models. Existing explanations methods reviewed in this chapter are summarized in Table 3.6. As their practicality issues, which have been revealed for the last few years, we also provided recent discussions on the plausibility issues with post-hoc local explanation methods and the Rashomon effect in interpretable models, respectively.

As shown in this chapter, while several methods for realizing explainability have

Table 3.6: Summary of existing methods for explainable machine learning.

Approach	Method / Model	Representative Reference
Local explanation	Local surrogate	[197, 280, 281, 289, 313, 375]
	Feature attribution	[67, 72, 192, 222, 223, 350, 351]
	Saliency map	[27, 298, 305, 308, 314, 317, 320]
	Influential example	[32, 123, 182, 186, 189, 362]
	Counterfactual explanation	[175, 242, 257, 259, 328, 335, 344]
	Amortized explanation	[66, 77, 162, 187, 296, 363]
	Summary of explanation	[41, 108, 173, 278, 279]
Interpretable model	Sparse linear model	[154, 207, 307, 325, 326, 364, 377]
	Sparse linear integer model	[316, 331, 333, 334, 371]
	Decision tree	[8, 10, 38, 49, 149, 214, 341]
	Rule list / Rule set	[23, 288, 347, 360] / [78, 198, 353]
	Rule ensemble	[35, 36, 60, 102, 235, 355]
	Interpretable NN	[19, 25, 63, 64, 253, 274, 345]
	Interpretability regularizer	[95, 204, 267, 282, 284, 285]

been developed, several issues with existing explanation methods that might prevent them from being applied to real decision-making tasks have been raised. Nevertheless, there have been certainly a few studies reporting that the existing explanation methods actually help decision-makers in their tasks [60, 196, 209, 224]. Therefore, identifying the issues with the practicality of the existing explanation methods and developing methods for addressing these issues are critical research directions going forward.

Motivated by the discussions in this chapter, the remainder of this thesis aims (i) to identify the issues and desiderata with the existing explanation methods, (ii) to

introduce optimization problems for satisfying the desiderata by addressing the issues, and (iii) to propose solution methods for them based on MILO. The relations of the contributions in this thesis to the previous studies are summarized as follows:

- In Chapters 4 and 5, we focus on *Counterfactual Explanation (CE)* [344] described in Section 3.2.3. CE is one of the post-hoc local explanation methods that show both why undesirable predictions are given and how users should act to obtain their desired prediction results [233, 237]. Since CE provides a perturbation vector for altering the prediction result of a machine learning model into the desired result, human users can interpret it as an "action" for obtaining their desired prediction result. However, existing CE methods often fail to provide realistic actions in the sense that the users can actually execute them [339]. To adequately evaluate the reality of actions, several criteria have been proposed from various perspectives. For example, while Laugel et al. have introduced the notion of *connectedness* to determine whether the obtained action is plausible [201, 202], Karimi et al. have proposed the *intervention* framework for taking the causal effects between features into account [177]. This line of these studies is related to the plausibility issues with other post-hoc local explanation methods described in Section 3.2.5. However, these previous studies have introduced only new mathematical models of CE, and they have not proposed solution methods for them. In Chapters 4 and 5, we propose new frameworks of CE for providing realistic actions. In contrast to previous studies, we propose not only new mathematical models of CE but also solution methods for them based on MILO.

- In Chapter 6, we focus on *decision trees*, which is one of the most popular interpretable models described in Section 3.3.2. As mentioned in Section 3.3.4, there often exist multiple interpretable models that are approximately equally accurate

but provide different predictions and explanations for the same prediction task, which phenomenon is called "*Rashomon effect*" [48]. In real deployments of machine learning models, we may face the problem of the Rashomon effect; that is, when decision-makers are required to update their models due to some reasons, e.g., dataset shift [276] or issues with fairness [127, 264] or robustness [15, 321], they retrain their models and often obtain new models that are unnecessarily significantly different in their prediction and interpretation from their initial models [122]. As shown in Section 3.3.4, such differences between the initial model and retrained model can cause serious problems regarding the reliability of deployed models, such as the risk of *fairwashing* [12, 13, 129]. To address this issue, we introduce a new framework for stabilizing the predictions and explanations of interpretable models under given constraints. In particular, we formulate a post-processing task of modifying a given decision tree so as to satisfy fairness constraints without significantly changing both its prediction and interpretation. Then, we propose a solution method for the formulated task based on MILO.

Chapter 4

Distribution-Aware Counterfactual Explanation

Post-hoc explanation methods for machine learning models have been widely used to support decision-making. *Counterfactual Explanation (CE)*, also known as *Actionable Recourse*, is one of the post-hoc explanation methods that provide a perturbation vector that alters the prediction result obtained from a classifier. Users can directly interpret the perturbation as an "*action*" to obtain their desired decision results. However, actions extracted by existing methods often become unrealistic for users because they do not adequately consider the characteristics corresponding to the data distribution, such as feature-correlations and outlier risk. To suggest an executable action for users, we propose a new framework of CE, which we refer to as *Distribution-Aware Counterfactual Explanation (DACE)*, that extracts a realistic action by evaluating its reality on the empirical data distribution. Here, the key idea is to define a new cost function based on the Mahalanobis distance and the local outlier factor. Then, we propose a mixed-integer linear optimization approach to extracting an optimal action by mini-

mizing the defined cost function. Experiments conducted on real datasets demonstrate the effectiveness of the proposed method compared with existing CE methods.

4.1 Introduction

Complex machine learning models, such as deep neural networks and tree ensembles, are widely applied to critical decision-making tasks (e.g., medical diagnoses, judicial decisions, and loan approvals). While these models have achieved high prediction accuracy, they often suffer from a lack of *explainability* [1, 233]. If the decisions based on machine learning models might have a significant impact on individuals, decision-makers must provide explanations of the decisions to assure users of their correctness [119]. Consequently, post-hoc methods for extracting explanations of individual predictions from complex machine learning models have attracted considerable attention [189, 223, 280, 281]. Such extracted explanations help users understand given decisions based on complex model predictions and accept them with confidence [121].

To provide users with better insights, post-hoc methods need to show both why their undesirable predictions are given and how users should act to obtain their desired prediction results [85,233]. *Counterfactual Explanation (CE)* [344], which is also known as *Actionable Recourse* [335], is one of the post-hoc explanation methods that satisfy these requirements. Consider an example on the FICO dataset [97], which is a dataset of actual Home Equity Line of Credit (HELOC) applications. The task of the dataset is to predict whether individuals will default on their HELOC based on their livelihood, such as credit histories. Imagine a situation that a user with current status x° receives an undesired prediction $f(x^\circ) \neq y^*$ for a target label y^* from a trained classifier f, which means that the user is predicted as "high risk of default" by the classifier. CE

attempts to provide the user with a perturbation vector a as an "*action*" so that current status x° can be changed to obtain the desired outcome, i.e., $f(x^\circ + a) = y^*$, which means "low risk of default."

To achieve this, existing CE methods find an optimal solution $a^* \in \mathcal{A}$ of the following optimization problem:

$$a^* := \underset{a \in \mathcal{A}}{\arg\min} \quad C(a \mid x^\circ) \quad \text{subject to} \quad f(x^\circ + a) = y^*,$$

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 an action a. This problem is related to *adversarial examples* [321] in the sense that it finds a perturbation a that alters the output of a classifier f. In the context of CE, a perturbation a is interpreted as a required action for a user x° to obtain the desired prediction result y^*. Therefore, the action suggested by CE should be executable for users. In this chapter, we focus on this characteristic property of CE and discuss how to provide a realistic action that users can directly refer to and execute.

A key to extracting realistic actions is to design a cost function C that considers the characteristics of the data distribution [30, 202, 259]. While several useful cost functions, such as the total log-percentile shift (TLPS) [335], have been proposed, we demonstrate that such previously proposed cost functions often extract unrealistic actions. Figure 4.1 shows actions a (yellow arrows) extracted using the TLPS from random forest classifiers trained on the FICO dataset, and these modified instances $x^\circ + a$ (yellow triangles) on the feature space of the dataset. The features "MSinceOldestTradeOpen," "AverageMInFile," "ExternalRiskEstimate," and "PercentInstallTrades" indicate the months passed from the oldest trade, the average history length of trades, the consolidated risk estimation from other credit bureaus, and the percentage of trades paid in installments, respectively. Table 4.1(a) shows the actual values of the extracted

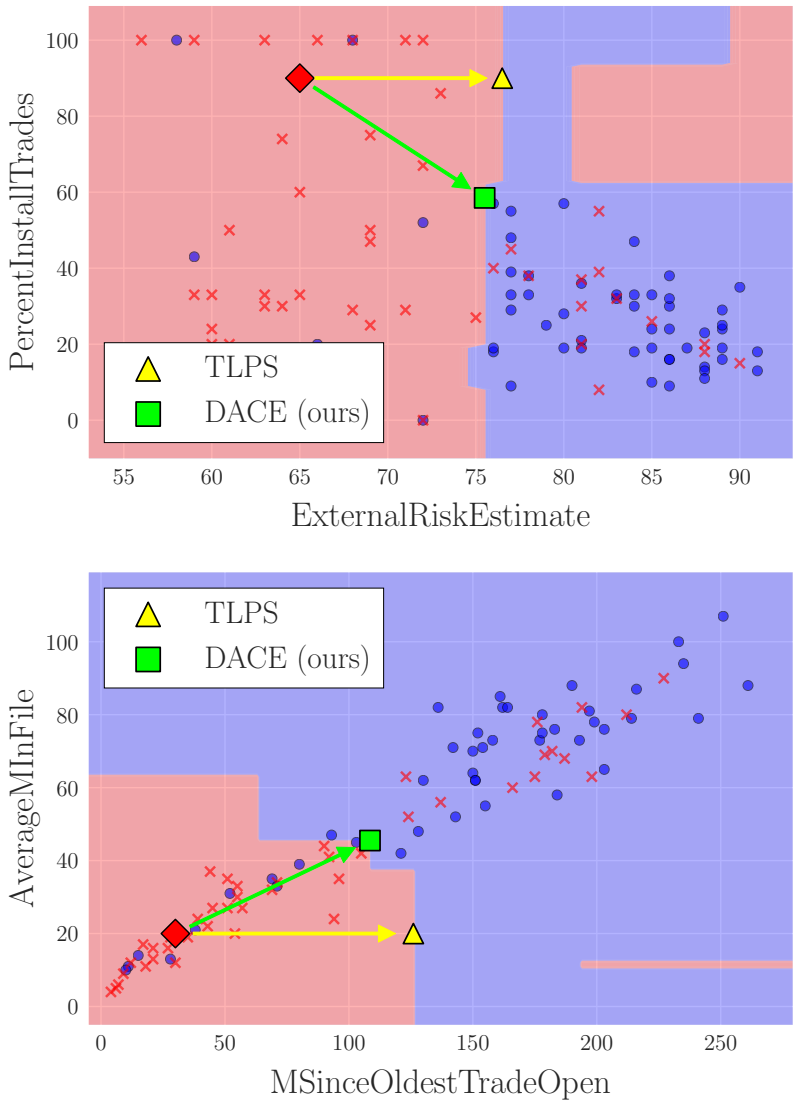


Figure 4.1: Two-dimensional illustrations of actions extracted on the FICO dataset. Blue (resp. red) regions represent decision regions of random forest classifiers whose prediction results are "low (resp. high) risk of credit". The yellow and green arrows represent actions for the input instances (red diamonds) extracted by TLPS and the proposed DACE, respectively.

Table 4.1: Examples of CE extracted on the FICO dataset. These actions provide users how to change their feature values such that they are predicted as "low risk of default" from the classifier learned on the dataset.

(a) TLPS [335]

Feature to Change	**Action**	**MD**	**LOF**
"MSinceOldestTradeOpen"	30 → 126 (+96)	**4.39**	1.93
"ExternalRiskEstimate"	65→ 77 (+12)	1.35	**7.92**

(b) DACE (ours)

Feature to Change	**Action**	**MD**	**LOF**
"MSinceOldestTradeOpen"	30 → 109 (+79)	**1.11**	1.17
"AverageMInFile"	20 → 46 (+26)		
"ExternalRiskEstimate"	65 → 76 (+11)	1.40	**1.04**
"PercentInstallTrades"	90 → 58 (-32)		

actions. It is evident that these modified instances $x^\circ + a$ are located in the region predicted as "low risk of default" by the classifiers in Figure 4.1; therefore, users can obtain their desired outcome by changing each feature according to the suggested actions in Table 4.1(a). However, we argue that these actions are not realistic for users from the following two perspectives corresponding to the characteristics of the empirical data distribution:

- **Feature-correlation:** Because each feature is often dependent on other features, i.e., having non-zero correlation, the cost of changing a value with respect to a feature should be evaluated depending on both the amount of its difference and relation to other features. In Figure 4.1 (top), it seems unnatural to increase only "MSinceOldestTradeOpen" without increasing "AverageMInFile" because these features are correlated.

- **Outlier risk:** By simply minimizing the cost of a, there is a risk that its modified instance $x^\circ + a$ becomes an outlier of the data distribution. In Figure 4.1 (bottom), it seems unrealistic to increase "ExternalRiskEstimate" without decreasing "PercentInstallTrades", because there are no training instances near $x^\circ + a$.

Based on the above observations, our research goals are (i) to model the reality of an action as a cost function C, and (ii) to propose a method to optimize C for extracting realistic actions.

4.1.1 Contributions

In this chapter, we propose a new framework of CE, named *Distribution-Aware Counterfactual Explanation (DACE)*, that extracts a realistic action for users. Our contributions can be summarized as follows:

- We propose a new cost function based on the *Mahalanobis distance (MD)* [227] and *Local Outlier Factor (LOF)* [52] to evaluate the reality of actions. MD is a metric that captures the relationships between features, and LOF is a popular outlier score that measures how unusual a given instance is by using k-nearest neighbors (k-NN).

- We formulate the problem of finding an optimal action according to our cost function as a *mixed-integer linear optimization (MILO)* problem, which can be solved by modern MILO solvers, such as CPLEX [153]. For computational efficiency, we show that if we use ℓ_1-norm based MD and 1-NN based LOF for the cost function, the number of variables and constraints of the problem can be reduced dramatically.

- By experiments on real datasets, we demonstrate the effectiveness of DACE compared to existing methods, including TLPS [335], MAD [292,344], and PCC [28].

Table 4.1(b) and the green arrows in Figure 4.1 show the actions extracted by our DACE. These results suggest that the MD and LOF of actions well reflect the feature-correlations and outlier risks, respectively, and that DACE can extract realistic actions in the sense of these properties.

4.1.2 Related Work

Several post-hoc methods for generating explanations from complex models have been proposed [189,223,239,280,281]. *Counterfactual Explanation (CE)* [344], also known as *Actionable Recourse* [335], is one of the post-hoc methods that has attracted attention in recent years. Existing CE methods can be categorized as gradient-based [178, 241, 344], autoencoder [81, 226, 259], SAT [175], heuristic search [200], or mixed-integer

linear optimization (MILO) [76, 174, 292, 335]. In this chapter, we focus on MILO-based methods, which can directly handle non-differentiable models over possibly non-continuous features, such as random forests over categorical features. Our results extend existing MILO-based methods to deal with nonlinear cost functions such as MD and LOF.

Some recent studies have pointed out issues with existing post-hoc explanation methods, e.g., lack of robustness [113, 286], privacy risks [14, 234], and unrealistic assumptions [30, 260, 339]. With respect to CE, various metrics to evaluate the reliability of actions have been proposed. Laugel et al. introduced the notion of the *connectedness* of an action to exclude a meaningless action that transfers a given instance to an empty decision region containing no training instances [202]. They also pointed out that most existing CE methods can generate non-connected actions. Note however that their connectedness and our criterion are incomparable. Actually, in Figure 4.1, all actions are connected; however, some are classified as bad according to our criterion. Another criterion, called *proximity* [201] for CE, is related to the LOF. To the best of our knowledge, our method is a first attempt to optimize proximity, compared to previous studies that only proposed the criterion.

4.2 Preliminaries

4.2.1 Mahalanobis Distance

The *Mahalanobis distance (MD)* [227] is a popular metric in statistics and metric learning [191]. For two vectors $x, x' \in \mathbb{R}^D$ and a positive semi-definite matrix $M \in$

$\mathbb{R}^{D\times D}$, MD $d_{\mathrm{M}}\colon \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_{\geq 0}$ between x and x' is defined as follows:

$$d_{\mathrm{M}}(x, x' \mid M) := \sqrt{(x'-x)^\top M (x'-x)}.$$

Since M is positive semi-definite, M can be decomposed as $M = U^\top U$, where $U \in \mathbb{R}^{D\times D}$. Thus, $d_{\mathrm{M}}(x, x' \mid M)$ can be also expressed as follows:

$$d_{\mathrm{M}}(x, x' \mid M) = \|U(x'-x)\|_2,$$

where $\|\cdot\|_p$ denotes the ℓ_p-norm.

In statistics, the inverse matrix of the covariance matrix Σ of the distribution where x and x' follow is often used as M. The MD $d_{\mathrm{M}}(x, x' \mid \Sigma^{-1})$ with Σ^{-1} is scale-invariant and considers the feature-correlations [225].

4.2.2 Local Outlier Factor

We assume a metric space $(\mathcal{X}, \Delta)$ and a set of N instances $X \subseteq \mathcal{X}$. As a metric $\Delta\colon \mathcal{X}\times\mathcal{X} \to \mathbb{R}_{\geq 0}$ on $\mathcal{X}$, we consider $\Delta(x, x') = \sum_{d=1}^{D} \Delta_d(x_d, x'_d)$, where $\Delta_d\colon \mathcal{X}_d\times\mathcal{X}_d \to \mathbb{R}_{\geq 0}$ is some dissimilarity measure of the feature $d \in [D]$. We omit them if it is clear from the context.

The *local outlier factor (LOF)* [52] is a prominent outlier score based on the local densities of instances [7]. To define the LOF, we first introduce some notations. For an instance $x \in \mathcal{X}$ and a positive integer $k \in [N]$, let $N_k(x) \subseteq X$ be its *k-nearest neighbors* (k-NN) on X. Let $d_k(x)$ be the distance Δ between x and its k-th nearest instance, i.e., $d_k(x) = \max_{x'\in N_k(x)} \Delta(x, x')$. For $x, x' \in \mathcal{X}$, the *k-reachability distance* of x with respect to x' is defined by $rd_k(x, x') := \max\{\Delta(x, x'), d_k(x')\}$. The *$k$-local reachability density* of x is defined by $lrd_k(x) := |N_k(x)| / \sum_{x'\in N_k(x)} rd_k(x, x')$. Then,

the k-*LOF* of x on X is defined as follows:

$$q_k(x \mid X) := \frac{1}{|N_k(x)|} \sum_{x' \in N_k(x)} \frac{lrd_k(x')}{lrd_k(x)}.$$

4.3 Problem Statement

4.3.1 Action and Action Set

Let $f\colon \mathcal{X} \to \mathcal{Y}$ and $x^\circ = (x_1^\circ, \ldots, x_D^\circ) \in \mathcal{X}$ be a classifier and a given instance, respectively, such that $f(x^\circ) = -1$. We define an *action* for x° as a perturbation vector $a \in \mathbb{R}^D$ such that $x^\circ + a \in \mathcal{X}$ and $f(x^\circ + a) = +1$. A *candidate action set* $\mathcal{A} = A_1 \times \cdots \times A_D$ is a finite set of feasible actions such that $0 \in A_d$ and $A_d \subseteq \{a_d \in \mathbb{R} \mid x_d^\circ + a_d \in \mathcal{X}_d\}$ for $d \in [D]$. For $d \in [D]$, we denote $I_d = |A_d|$.

Depending on the type of the feature $d \in [D]$ and the classifier f, we can often determine each A_d automatically [76, 335]. For example, if x_d is a feature representing "age," then we construct A_d such that $a_d \in \mathbb{N} \cup \{0\}$ holds for any $a_d \in A_d$ because "age" is represented as a positive integer value and its value cannot be decreased. If x_d is an immutable feature (e.g., "gender"), then we set $A_d = \{0\}$. If x_d is a real-valued feature, we first set a bounded interval $[a_d^{\min}, a_d^{\max}]$ such that $[x_d^\circ + a_d^{\min}, x_d^\circ + a_d^{\max}] \subseteq \mathcal{X}_d$ based on the training dataset. Then, we construct A_d as a grid with some discretization step $\varepsilon > 0$[1] such that $A_d \subseteq [a_d^{\min}, a_d^{\max}]$ and $a_d^{\min}, a_d^{\max} \in A_d$. When f is an LM, this discretization procedure is guaranteed to have no effect on the feasibility of the actions [335]. It is also noteworthy that when f is a TE, we can automatically determine A_d depending on the branching thresholds b of the internal nodes of the decision trees $f_1, \ldots, f_T$ whose branching feature is d [76].

[1] In our implementation, we choose ε such that $|A_d| = 100$.

4.3.2 Cost Function

We introduce a new cost function $C_{\mathrm{DACE}}\colon \mathcal{A} \to \mathbb{R}_{\geq 0}$ as a score to evaluate whether an action is realistic for users. Given a positive semi-definite matrix $M \in \mathbb{R}^{D\times D}$, set of N instances $X \subseteq \mathcal{X}$, positive integer $k \in [N]$, and trade-off parameter $\lambda \geq 0$, we define C_{DACE} with respect to an input instance $x^\circ \in \mathcal{X}$ as follows:

$$C_{\mathrm{DACE}}(a \mid x^\circ) := d^2_{\mathrm{M}}(x^\circ, x^\circ + a \mid M) + \lambda \cdot q_k(x^\circ + a \mid X),$$

where

- $d^2_{\mathrm{M}}(x^\circ, x^\circ + a \mid M)$ is the squared MD between the input instance x° and its modified instance $x^\circ + a$, and
- $q_k(x^\circ + a \mid X)$ is the k-LOF of $x^\circ + a$ on X.

We can adjust a trade-off between the MD d^2_{M} and k-LOF q_k of an obtained action a by tuning the parameter λ.

4.3.3 Problem Definition

Our aim is to find an action $a \in \mathcal{A}$ that alters the prediction result $f(x^\circ)$ and minimizes $C_{\mathrm{DACE}}(a \mid x^\circ)$. This problem can be defined as follows.

Problem 1. Given a classifier $f\colon \mathcal{X} \to \mathcal{Y}$, input instance $x^\circ \in \mathcal{X}$ such that $f(x^\circ) = -1$, action set $\mathcal{A}$, positive semi-definite matrix $M \in \mathbb{R}^{D\times D}$, set of N instances $X \subseteq \mathcal{X}$, positive integer $k \in [N]$, and trade-off parameter $\lambda \geq 0$, find an action $a^* \in \mathcal{A}$ that is an optimal solution for the following optimization problem:

$$\underset{a\in\mathcal{A}}{\text{minimize}} \quad C_{\mathrm{DACE}}(a \mid x^\circ) \quad \text{subject to} \quad f(x^\circ + a) = +1.$$

By solving the above optimization problem, we obtain an action a that takes care of feature-correlations and outlier risk on the empirical distribution.

4.4 MILO Formulation

In this section, we propose an MILO formulation to solve Problem 1[2].

4.4.1 Basic Ideas

While our proposed cost function C_{DACE} is nonlinear and has a discrete structure due to $N_k(x^\circ + a)$, we can exactly formulate Problem 1 as an MILO problem using standard modeling techniques on integer optimization [359]. However, to linearize the nonlinearity and discreteness, this naive formulation requires $\mathcal{O}(I^2 + N^2)$ auxiliary variables and constraints, where $I := \sum_{d=1}^{D} |A_d|$, and preliminary experiments demonstrated that this formulation was computationally infeasible on standard datasets.

To avoid introducing $\mathcal{O}(I^2 + N^2)$ auxiliary variables and constraints, we introduce a surrogate objective function $\hat{C}_{\mathrm{DACE}}$ that approximates our cost function C_{DACE} and optimize it instead of C_{DACE}. Our main ideas are (i) fixing $k = 1$ for the LOF $q_k(x^\circ + a)$ of C_{DACE}, and (ii) replacing the MD d^2_{M} of C_{DACE} with *ℓ_1-norm based Mahalanobis distance (ℓ_1-MD)* $\hat{d}_{\mathrm{M}}$ defined as

$$\hat{d}_{\mathrm{M}}(x, x' \mid M) := \|U(x' - x)\|_1,$$

which is based on the fact that d^2_{M} can be expressed as $d^2_{\mathrm{M}}(x, x' \mid M) = \|U(x' - x)\|_2^2$. Overall, we define a surrogate objective function $\hat{C}_{\mathrm{DACE}}$ of C_{DACE} as follows:

$$\hat{C}_{\mathrm{DACE}}(a \mid x^\circ) := \hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M) + \lambda \cdot q_1(x^\circ + a \mid X).$$

In the rest of this chapter, we formulate the following problem as an MILO problem instead of Problem 1:

$$\underset{a \in \mathcal{A}}{\text{minimize}} \quad \hat{C}_{\mathrm{DACE}}(a \mid x^\circ) \quad \text{subject to} \quad f(x^\circ + a) = +1.$$

[2]For details of the general techniques on integer optimization used in this chapter, see, e.g., [359].

First, 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 ($\pi_{d,i} = 0$). Then, $\pi_{d,i}$ must satisfy the following constraint:

$$\sum_{i=1}^{I_d} \pi_{d,i} = 1, \qquad \forall d \in [D]. \tag{4.1}$$

By using $\pi_{d,i}$, each element of an action $a = (a_1, \ldots, a_D) \in \mathcal{A}$ can be expressed as $a_d = \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}$. In the following, we formulate the objective $\hat{C}_{\mathrm{DACE}}(a \mid x^\circ)$ and constraint $f(x^\circ + a) = +1$ using linear constraints of $\pi_{d,i}$.

4.4.2 Surrogate Objective Function

ℓ_1-norm based MD (ℓ_1-MD). From the definition of the ℓ_1-MD, $\hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M)$ can be expressed as follows:

$$\hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M) = \|Ua\|_1 = \sum_{d=1}^{D} |\langle U_d, a \rangle| ,$$

where $U_d = (U_{d,1}, \ldots, U_{d,D})$ is the d-th row vector of U. We introduce variables $\delta_d \geq 0$ for $d \in [D]$ such that $\delta_d = |\langle U_d, a \rangle|$. Then, $\hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M)$ can be expressed as

$$\hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M) = \sum_{d=1}^{D} \delta_d,$$

with the following constraints:

$$-\delta_d \leq \sum_{d'=1}^{D} U_{d,d'} \cdot \sum_{i=1}^{I_d} a_{d',i} \cdot \pi_{d',i} \leq \delta_d, \qquad \forall d \in [D]. \tag{4.2}$$

The ℓ_1-MD $\hat{d}_{\mathrm{M}}$ is not exactly the same as the original MD d^2_{M}, however, we show the approximation ratio of $\hat{d}_{\mathrm{M}}$ with respect to d^2_{M} in the following proposition.

Proposition 1 (Approximation Ratio)**.** *Let* $a^*, \hat{a} \in \mathbb{R}^D$ *be two vectors such that* $a^* = \arg\min_{a \in \mathbb{R}^D} d_{\mathrm{M}}^2(x^\circ, x^\circ + a \mid M)$ *and* $\hat{a} = \arg\min_{a \in \mathbb{R}^D} \hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M)$*, respectively. Then,* $d_{\mathrm{M}}(x^\circ, x^\circ + \hat{a} \mid M) \leq \sqrt{D} \cdot d_{\mathrm{M}}(x^\circ, x^\circ + a^* \mid M)$ *holds.*

Proof. Let $U \in \mathbb{R}^{D \times D}$ be a matrix such that $M = U^\top U$. By the definitions, $d_{\mathrm{M}}(x^\circ, x^\circ + a \mid M)$ and $\hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M)$ are expressed as $d_{\mathrm{M}}(x^\circ, x^\circ + a \mid M) = \|Ua\|_2$ and $\hat{d}_{\mathrm{M}}(x^\circ, x^\circ + a \mid M) = \|Ua\|_1$, respectively. From the properties of ℓ_p-norm, $\|U\hat{a}\|_2 \leq \|U\hat{a}\|_1$ and $\|Ua^*\|_1 \leq \sqrt{D} \cdot \|Ua^*\|_2$ hold. From the definitions of a^* and $\hat{a}$, $\|U\hat{a}\|_1 \leq \|Ua^*\|_1$ holds. By combining these inequalities, we have $\|U\hat{a}\|_2 \leq \sqrt{D} \cdot \|Ua^*\|_2$, which is equivalent to $d_{\mathrm{M}}(x^\circ, x^\circ + \hat{a} \mid M) \leq \sqrt{D} \cdot d_{\mathrm{M}}(x^\circ, x^\circ + a^* \mid M)$. □

1-Local Outlier Factor (1-LOF). From the definitions of q_k and lrd_k, 1-LOF of $x^\circ + a$ can be expressed as

$$q_1(x^\circ + a \mid X) = lrd_1(x^{(n^*)}) \cdot rd_1(x^\circ + a, x^{(n^*)}),$$

where $n^* = \arg\min_{n \in [N]} \Delta(x^\circ + a, x^{(n)})$, i.e., $N_1(x^\circ + a) = \{x^{(n^*)}\}$. Because n^* and $rd_1(x^\circ + a, x^{(n^*)})$ vary depending on $x^\circ + a$, i.e., the variables $\pi_{d,i}$, we need to formulate n^* and the value of $rd_1(x^\circ + a, x^{(n^*)})$ by linear constraints of $\pi_{d,i}$. We introduce variables $\nu_n \in \{0, 1\}$ and $\rho_n \geq 0$ for $n \in [N]$ such that $\nu_n = \mathbb{I}\left[x^{(n)} \in N_1(x^\circ + a)\right]$ and $\rho_n = rd_1(x^\circ + a, x^{(n)}) \cdot \mathbb{I}\left[x^{(n)} \in N_1(x^\circ + a)\right]$, respectively. Then, $q_1(x^\circ + a)$ can be expressed as a linear form of ρ_n as follows:

$$q_1(x^\circ + a \mid X) = \sum_{n=1}^{N} lrd_1(x^{(n)}) \cdot \rho_n,$$

with the following constraints:

$$\sum_{n=1}^{N} \nu_n = 1, \tag{4.3}$$

$$\sum_{d=1}^{D}\sum_{i=1}^{I_d}(c_{d,i}^{(n)} - c_{d,i}^{(m)}) \cdot \pi_{d,i} \leq C_n \cdot (1 - \nu_n), \qquad \forall n, m \in [N], \tag{4.4}$$

$$\rho_n \geq d_1(x^{(n)}) \cdot \nu_n, \qquad \forall n \in [N], \tag{4.5}$$

$$\rho_n \geq \sum_{d=1}^{D}\sum_{i=1}^{I_d} c_{d,i}^{(n)} \cdot \pi_{d,i} - C_n \cdot (1 - \nu_n), \qquad \forall n \in [N], \tag{4.6}$$

where $c_{d,i}^{(n)} = \Delta_d(x_d^\circ + a_{d,i}, x_d^{(n)})$ and C_n is a constant value for the *big-M constraint* [359] such that $C_n \geq \max_{a\in\mathcal{A}} \Delta(x^\circ + a, x^{(n)})$, respectively[3]. Note that $lrd_1(x^{(n)})$, $c_{d,i}^{(n)}$, C_n, and $d_1(x^{(n)})$ are constant values because they can be computed when X and $\mathcal{A}$ are given. Constraints (4.3) and (4.4), which are based on the statement $\nu_m = 1 \Rightarrow \forall n \in [N] : \Delta(x^\circ + a, x^{(m)}) \leq \Delta(x^\circ + a, x^{(n)})$, are the constraints for expressing the nearest instance of $x^\circ + a$. Constraints (4.5) and (4.6) represent k-reachability distance $rd_k(x^\circ + a, x^{(n)})$.

4.4.3 Base Learner Constraints

We introduce variables $\xi_t \in \mathbb{R}$ for $t \in [T]$ such that $\xi_t = f_t(x^\circ + a)$, where f_t is the t-th base learner of f. From the definition of the AC, the constraint $f(x^\circ + a) = +1$ is equivalent to the following linear constraint of ξ_t:

$$\sum_{t=1}^{T} w_t \cdot \xi_t \geq b. \tag{4.7}$$

We express the constraint $\xi_t = f_t(x^\circ + a)$ by linear constraints of ξ_t and $\pi_{d,i}$ because $f_t(x^\circ + a)$ depends on the value of a, i.e., the variables $\pi_{d,i}$. In the following, we show how to express ξ_t when f is a linear model (LM), tree ensemble (TE), or multilayer perceptron (MLP).

[3]Since we assume $\Delta_d(x, x') = \sum_{d=1}^{D} \Delta_d(x_d, x'_d)$, we can determine the value of C_n as $C_n = \max_{a\in\mathcal{A}} \Delta(x^\circ + a, x^{(n)}) = \sum_{d=1}^{D} \max_{a_d \in A_d} \Delta(x_d^\circ + a_d, x_d^{(n)}) = \sum_{d=1}^{D} \max_{i\in[I_d]} c_{d,i}^{(n)}$.

Linear Models. From the definition of the LM, $T = D$ and $f_d(x^\circ + a) = x^\circ_d + a_d$ for $d \in [D]$. Thus, we can simply express the base learner of the LM as follows:

$$\xi_d = x^\circ_d + \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}, \qquad \forall d \in [D]. \tag{4.8}$$

Tree Ensembles. Each base learner f_t of the TE is a decision tree. Recall that a decision tree f_t with L_t leaves represents a partition $\{r_{t,1}, \ldots, r_{t,L_t}\}$ of the input domain $\mathcal{X}$ and can be expressed as $f_t(x) = \sum_{l=1}^{L} \hat{y}_{t,l} \cdot \mathbb{I}[x \in r_{t,l}]$. To express $\xi_t = f_t(x^\circ + a)$, we can utilize the following *decision logic constraint* proposed by Cui et al. [76]:

$$\phi_{t,l} \in \{0, 1\}, \qquad \forall t \in [T], \forall l \in [L_t], \tag{4.9}$$

$$D \cdot \phi_{t,l} \leq \sum_{d=1}^{D} \sum_{i \in \mathcal{I}^{(d)}_{t,l}} \pi_{d,i}, \qquad \forall t \in [T], \forall l \in [L_t], \tag{4.10}$$

$$\sum_{l=1}^{L_t} \phi_{t,l} = 1, \qquad \forall t \in [T], \tag{4.11}$$

$$\xi_t = \sum_{l=1}^{L_t} \hat{y}_{t,l} \cdot \phi_{t,l}, \qquad \forall t \in [T], \tag{4.12}$$

where $\mathcal{I}^{(d)}_{t,l} = \{i \in [I_d] \mid x^\circ_d + a_{d,i} \in r^{(d)}_{t,l}\}$ and $r^{(d)}_{t,l}$ is the subspace of $\mathcal{X}_d$ such that $r_{t,l} = r^{(1)}_{t,l} \times \cdots \times r^{(D)}_{t,l}$. Here, the variable $\phi_{t,l}$ is an indicator variable such that $\phi_{t,l} = \mathbb{I}[x^\circ + a \in r_{t,l}]$ represented by Constraint (4.10). Constraint (4.11) ensures that the modified instance $x^\circ + a$ is assigned to exactly one leaf of f_t. Because $\mathcal{I}^{(d)}_{t,l}$, L_t, and $\hat{y}_{t,l}$ can be computed when f and $\mathcal{A}$ are given, they are constants.

Multilayer Perceptrons. Each base learner f_t of the MLP is an output of the t-th neuron with the ReLU activation function, i.e., $f_t(x + a) = \max\{0, g_t(x + a)\}$, where $g_t(x) := w^{(t)} \cdot x + b^{(t)}$. Therefore, we need to extract the positive part of $f_t(x^\circ + a)$ as

Table 4.2: Numbers of variables (#Vars) and constraints (#Consts) required for the exact formulation of Problem 1 (left) and our formulation (4.17) (right). Note that $I \geq D$ from the definition of I, and that our formulation (4.17) optimizes $\hat{C}_{\mathrm{DACE}}$, i.e., ℓ_1-MD $\hat{d}_{\mathrm{M}}$ and 1-LOF q_1 rather than C_{DACE}, i.e., d^2_{M} and q_k for $k > 1$.

	#Vars	**#Consts**		**#Vars**	**#Consts**
d^2_{M}	$\mathcal{O}(I^2)$	$\mathcal{O}(I^2)$	$\hat{d}_{\mathrm{M}}$	$\mathcal{O}(D)$	$\mathcal{O}(D)$
N_k	$\mathcal{O}(N^2)$	$\mathcal{O}(N^2)$	N_1	$\mathcal{O}(N)$	$\mathcal{O}(N^2)$
q_k	$\mathcal{O}(N^2)$	$\mathcal{O}(N^2)$	q_1	$\mathcal{O}(N)$	$\mathcal{O}(N)$

ξ_t. To express $\xi_t = f_t(x^\circ + a)$, we can utilize the MILO formulation to count the linear regions of the MLP as proposed by Serra et al. [300]. We can extend their constraints to address our problem as follows:

$$\mu_t \in \{0,1\}, \bar{\xi}_t \geq 0, \qquad \forall t \in [T], \tag{4.13}$$

$$\xi_t \leq G_t \cdot \mu_t, \qquad \forall t \in [T], \tag{4.14}$$

$$\bar{\xi}_t \leq -\bar{G}_t \cdot (1 - \mu_t), \qquad \forall t \in [T], \tag{4.15}$$

$$\xi_t = \bar{\xi}_t + \sum_{d=1}^{D} w_d^{(t)} \cdot \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i} + F_t, \qquad \forall t \in [T], \tag{4.16}$$

where F_t, G_t, and $\bar{G}_t$ are constants such that $F_t = f_t(x^\circ)$, $G_t \geq \max_{a \in \mathcal{A}} f_t(x^\circ + a)$, and $\bar{G}_t \leq \min_{a \in \mathcal{A}} f_t(x^\circ + a)$, respectively. These values can be computed when f and $\mathcal{A}$ are given. The variable ν_t indicates whether $f_t(x^\circ + a)$ is positive, and $\bar{\xi}_t$ represents the negative part of $f_t(x^\circ + a)$. Constraint (4.16) represents $\xi_t = \max\{0, f_t(x^\circ + a)\}$. Note that the above constraints can be extended to a general MLP with more than two hidden layers as with the work by Serra et al. [300].

4.4.4 Overall Formulation

Finally, we show our overall formulation as follows:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{d=1}^{D} \delta_d + \lambda \cdot \sum_{n=1}^{N} lrd_1(x^{(n)}) \cdot \rho_n \\
\text{subject to} \quad & \text{Constraint } (4.1 - 4.7), \\
& \begin{cases} \text{Constraint } (4.8), & \text{if } H \text{ is a LM}, \\ \text{Constraint } (4.9 - 4.12), & \text{if } H \text{ is a TE}, \\ \text{Constraint } (4.13 - 4.16), & \text{if } H \text{ is a MLP}, \end{cases} \\
& \pi_{d,i} \in \{0, 1\}, && \forall d \in [D], \forall i \in [I_d], \\
& \delta_d \geq 0, && \forall d \in [D], \\
& \nu_n \in \{0, 1\}, && \forall n \in [N], \\
& \rho_n \geq 0, && \forall n \in [N].
\end{aligned}
\tag{4.17}
$$

Table 4.2 presents the total numbers of variables and constraints required for the naive exact formulation of Problem 1 and our formulation (4.17). Note that the former formulation optimizes our proposed cost function C_{DACE} exactly, while the latter formulation optimizes our surrogate objective function $\hat{C}_{\mathrm{DACE}}$ instead of C_{DACE}. Table 4.2 shows that our formulation (4.17) reduces the number of variables and constraints dramatically compared to the exact formulation. In our preliminary experiments, we confirmed that our formulation (4.17) succeeded to obtain optimal solutions for the instances that the exact formulation did not, and the obtained solutions by the formulation (4.17) had better quality in terms of the original objective function C_{DACE}. Therefore, we utilize the formulation (4.17) to extract actions for computational efficiency.

As with existing MILO-based methods [292, 335], our formulation can be (i) efficiently solved by off-the-shelf MILO solvers, such as CPLEX [153], (ii) applied to an algorithm for enumerating distinct actions as its subroutine, and (iii) customized by adding constraints desired by users. To summarize the above advantages, we can obtain actions that satisfy user-defined constraints without implementing designated algorithms. In the following, we show two concrete examples of additional constraints to improve the practicality of the obtained action.

Categorical Features. Because datasets often include categorical features, actions suggested by CE must also consider them. We assume that all the categorical features are one-hot encoded by preprocessing. Let $G \subseteq [D]$ be a set of features that represents a one-hot encoded categorical feature with $|G|$ categories, i.e., $\mathcal{X}_d = \{0, 1\}$ for $d \in G$ and $\sum_{d \in G} x_d = 1$ for any $x \in \mathcal{X}$. To maintain the structure of a categorical feature, the modified instance $x^\circ + a$ also must satisfy it, i.e., $\sum_{d \in G}(x^\circ_d + a_d) = 1$. Since $\sum_{d \in G} x^\circ_d = 1$ and $a_d = \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}$ for $d \in G$, we have

$$\begin{aligned}\sum_{d \in G}(x^\circ_d + a_d) &= \sum_{d \in G} x^\circ_d + \sum_{d \in G} a_d \\ &= 1 + \sum_{d \in G} \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}.\end{aligned}$$

Therefore, the constraint $\sum_{d \in G}(x^\circ_d + a_d) = 1$ can be expressed by the variables $\pi_{d,i}$ as follows:

$$\sum_{d \in G} \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i} = 0.$$

Sparsity Constraint. In the context of CE, sparse actions are preferred from the perspective of interpretability [344]. To improve the sparsity of the obtained action,

we can add constraints corresponding to the number of features changed by the action. For a positive integer $B \in [D]$, we consider the constraint $\|a\|_0 \leq B$, which states that the maximum number of features to be changed by an action a is less than or equal to B. This constraint can be expressed as the following linear constraint on $\pi_{d,i}$:

$$\sum_{d=1}^{D} \sum_{i \in \bar{\mathcal{I}}_d} \pi_{d,i} \leq B,$$

where $\bar{\mathcal{I}}_d = \{i \in [I_d] \mid a_{d,i} \neq 0\}$, which can be computed when $\mathcal{A}$ is given.

4.5 Experiments

To investigate the effectiveness of DACE, we conducted experiments on real datasets in which we compared its performance with that of existing CE methods. All the code was implemented in Python 3.7 with scikit-learn and IBM ILOG CPLEX v12.10[4]. All experiments were conducted on 64-bit macOS Catalina 10.15.6 with Intel Core i9 2.4GHz CPU and 64 GB memory. In addition, a 600 second time limit was imposed for obtaining the solution.

4.5.1 Experimental Setting

We used four real datasets: FICO ($D = 23$) [97], German ($D = 61$), WineQuality ($D = 12$), and Diabetes ($D = 10$) [88] datasets, where D is the number of features. For the German dataset, we transformed each categorical feature into a one-hot encoded vector. We randomly split each dataset into training (75%) and test (25%) instances, and trained ℓ_2-regularized logistic regression classifiers (LR), random forest classifiers

[4]All the code and scripts are available at the following repository: `https://github.com/kelicht/dace`.

(RF) with $T = 100$ decision trees, and two-layer ReLU network classifiers (MLP) with $T = 200$ neurons, on each training dataset. Then, we extracted actions for test instances that had received undesired prediction results, such as predicted as "high risk of default" from each classifier. For the interpretability of extracted actions a, we imposed the sparsity constraint $\|a\|_0 \leq B$ with $B = 4$ for all datasets and methods.

Baseline Methods. We compared our proposed method (**DACE**), which approximately optimizes our proposed cost function C_{DACE} by optimizing our surrogate objective function $\hat{C}_{\mathrm{DACE}}$, to three existing methods. The main difference between DACE and existing methods is a cost function to be optimized. One cost function is the total log-percentile shift (**TLPS**) [335] that evaluates actions based on the cumulative distribution functions estimated from training instances. Another cost is the weighted ℓ_1-norm based on the inverse of median absolute deviation (**MAD**) [292, 344]. In addition to these costs, we also compared our DACE with the weighted ℓ_2-norm based on the Pearson's correlation coefficients (**PCC**) proposed by [28] to generate imperceptible adversarial examples. Note that we excluded an exact MILO formulation for optimizing C_{DACE} from baselines because it often failed to obtain good solutions within 600 seconds in our preliminary experiments.

Evaluation Scores. To compare the qualities of obtained actions a, we measured (i) the MD $d_{\mathrm{M}}(x^\circ, x^\circ + a \mid \Sigma^{-1})$, where Σ is the covariance matrix estimated from the training instances X, (ii) the 10-LOF $q_{10}(x^\circ + a \mid X_+)$ on the training instances $X_+ \subseteq X$ labeled as the target label (e.g., "low risk of default"), and (iii) running time to solve each MILO problem. The MD $d_{\mathrm{M}}(x^\circ, x^\circ + a \mid \Sigma^{-1})$ can measure the effort to execute an action a by considering the feature-correlations [225]. k-LOF $q_k(x^\circ + a \mid X_+)$ represents the risk that the action a leads x° to be an outlier on the instances with the

target label. We evaluated whether actions extracted by the baselines and DACE are realistic for users in terms of the above criteria.

4.5.2 Comparison with Existing Methods

We compared the actions extracted by DACE with those extracted by the baselines. We set $\lambda = 1.0$ for the FICO and Diabetes datasets and $\lambda = 0.01$ for the German and WineQuality datasets. These parameters were selected based on the sensitivity analyses described below.

Table 4.3 and Table 4.4 present the average MD, 10-LOF, and running time for each classifier and dataset. As shown in Table 4.3, we can observe that DACE achieved lower MD and 10-LOF than those of the baselines regardless of classifiers and datasets. Figure 4.2 presents scatter plots between MD and 10-LOF of each obtained action. We can see that DACE stably achieved lower MD and 10-LOF than the baseline methods. These results suggest that DACE stably obtain more realistic actions than the baseline methods by considering feature-correlation and the risk of leading to an outlier.

Regarding the average running time shown in Table 4.4, DACE was definitely slower than the baselines. However, Table 4.3 and Figure 4.2 also indicates that DACE stably found actions of better quality in terms of both MD and 10-LOF within 600 seconds, which is a reasonable calculation time. From these results, the effectiveness of DACE has been confirmed on real datasets, and we also argue that our proposed method is favorable when decision-makers and their customers require the quality of an action.

4.5.3 Sensitivity Analysis of Trade-off Parameter

Finally, we show the sensitivity of λ in $\hat{C}_{\mathrm{DACE}}$ on LR classifiers. Figure 4.3 presents the average MD and 10-LOF of obtained actions a for each λ. We can see that there

Table 4.3: Experimental results on the real datasets.

Dataset	Method	Logistic Regression		Random Forest		Multilayer Perceptron	
		MD	10-LOF	MD	10-LOF	MD	10-LOF
FICO	TLPS	5.44 ± 4.6	1.49 ± 1.0	2.12 ± 1.1	1.33 ± 0.63	4.25 ± 4.2	1.97 ± 1.3
	MAD	8.49 ± 5.8	1.49 ± 1.1	2.11 ± 1.2	1.39 ± 0.73	1.02 ± 0.79	1.42 ± 0.61
	PCC	9.34 ± 5.1	1.5 ± 1.1	2.45 ± 1.4	1.29 ± 0.26	8.11 ± 1.1e+01	1.5 ± 0.96
	DACE	**2.5 ± 1.5**	**1.32 ± 0.43**	**1.9 ± 0.97**	**1.24 ± 0.23**	**0.819 ± 0.72**	**1.4 ± 0.5**
German	TLPS	6.61 ± 2.0	1.23 ± 0.39	8.25 ± 1.9	1.23 ± 0.38	9.1 ± 0.37	1.27 ± 0.47
	MAD	9.51 ± 4.9	1.24 ± 0.4	9.02 ± 3.1	1.48 ± 0.86	9.95 ± 2.9	1.29 ± 0.55
	PCC	6.52 ± 2.5	1.23 ± 0.4	8.07 ± 3.3	1.53 ± 0.92	11.9 ± 4.0	1.29 ± 0.55
	DACE	**3.76 ± 1.1**	**1.18 ± 0.34**	**3.91 ± 1.1**	**1.09 ± 0.23**	**3.73 ± 1.1**	**1.04 ± 0.24**
WineQuality	TLPS	2.03 ± 1.6	4.54 ± 11	1.3 ± 0.89	1.55 ± 1.1	1.45 ± 1.5	1.82 ± 2.4
	MAD	2.06 ± 1.7	1.44 ± 0.99	1.3 ± 0.94	1.46 ± 1.0	1.09 ± 0.97	1.77 ± 2.4
	PCC	8.43 ± 5.5	1.45 ± 0.98	7.27 ± 4.1	1.47 ± 0.98	4.94 ± 3.1	1.43 ± 0.62
	DACE	**1.18 ± 0.92**	**1.33 ± 0.81**	**0.84 ± 0.5**	**1.39 ± 0.86**	**0.919 ± 0.69**	**1.35 ± 0.66**
Diabetes	TLPS	26.4 ± 13	5.34 ± 4.1	1.71 ± 1.5	1.24 ± 0.28	26.4 ± 15	6.46 ± 5.8
	MAD	3.45 ± 2.9	1.33 ± 0.42	1.6 ± 1.3	1.27 ± 0.44	3.38 ± 2.6	1.36 ± 0.4
	PCC	5.12 ± 3.6	1.24 ± 0.4	4.09 ± 3.0	1.28 ± 0.44	6.97 ± 3.9	1.27 ± 0.45
	DACE	**1.1 ± 0.89**	**1.18 ± 0.24**	**0.963 ± 0.66**	**1.22 ± 0.28**	**1.11 ± 0.83**	**1.18 ± 0.22**

Table 4.4: Average running times on the real datasets.

Classifier	Method	FICO	German	WineQuality	Diabetes
Logistic Regression	TLPS	0.0601 ± 0.0077	0.0203 ± 0.0025	0.0382 ± 0.0079	0.027 ± 0.0057
	MAD	0.0507 ± 0.0072	0.018 ± 0.003	0.0347 ± 0.0042	0.0337 ± 0.0055
	PCC	0.0461 ± 0.0059	0.0184 ± 0.0032	0.0431 ± 0.0052	0.0352 ± 0.0053
	DACE	41.2 ± 26	1.77 ± 0.58	15.0 ± 4.8	9.83 ± 2.8
Random Forest	TLPS	25.6 ± 50	36.7 ± 31	9.96 ± 5.8	13.6 ± 12
	MAD	16.0 ± 23	24.5 ± 17	11.2 ± 5.9	13.8 ± 12
	PCC	25.9 ± 21	7.06 ± 4.1	6.52 ± 3.4	9.54 ± 6.9
	DACE	207 ± 71	107 ± 67	492 ± 180	203 ± 150
Multilayer Perceptron	TLPS	3.67 ± 7.5	0.0888 ± 0.025	0.236 ± 0.14	0.0732 ± 0.032
	MAD	3.68 ± 7.1	0.0851 ± 0.032	0.247 ± 0.16	0.126 ± 0.062
	PCC	1.95 ± 8.5	0.0717 ± 0.028	0.232 ± 0.13	0.192 ± 0.12
	DACE	305 ± 260	1.7 ± 0.77	109 ± 150	35.5 ± 34

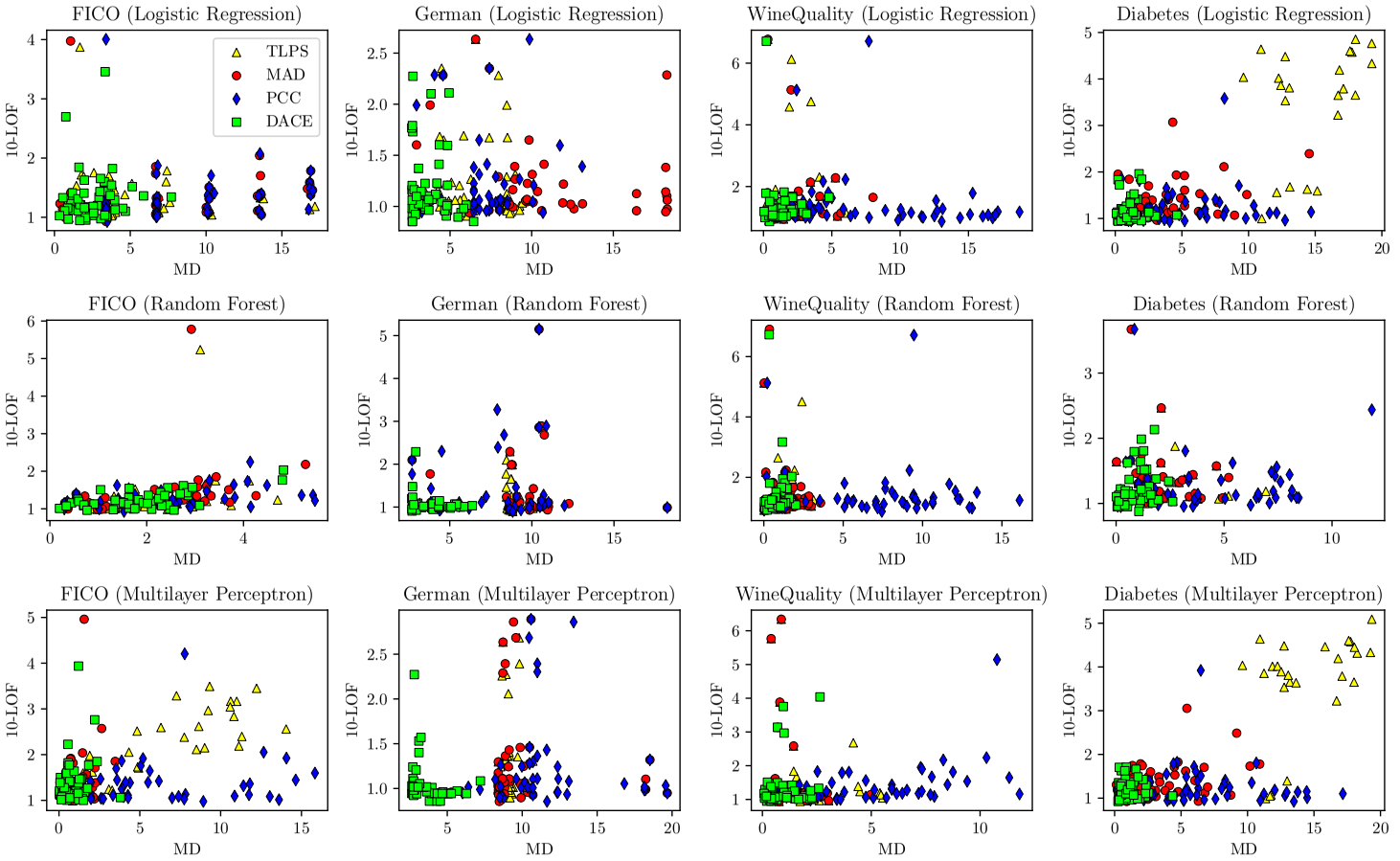


Figure 4.2: Scatter plots between MD and 10-LOF of the actions extracted by baseline methods and our DACE.

is a trade-off between MD and LOF of actions obtained by DACE with respect to the value of λ. Consequently, we need to select λ depending on whether a user emphasizes the preference or reliability of an action. In other words, by varying the value of λ, we can obtain several distinct actions that have diverse characteristics in terms of MD and LOF. As mentioned in [242, 335, 344], suggesting multiple actions may help users for referring to as their future guidelines. Figure 4.3 indicates that by varying λ, we can obtain several actions that have diverse characteristics in terms of MD and LOF.

4.6 Conclusion

In this chapter, we proposed a new framework of CE, named Distribution-Aware CE (DACE), for extracting a realistic action by considering the empirical distribution on labeled examples. We introduced a new cost function based on the Mahalanobis distance (MD) and the local outlier factor (LOF), and then proposed an efficient MILO formulation to approximately optimize the cost function. By experiments, we confirmed the effectiveness of our proposed method by comparing with existing methods.

Discussion. A critical challenge of CE for providing realistic actions is to adequately design a cost function C. However, it is difficult to select the cost function that satisfies all requirements that users desire [339]. As demonstrated in Figure 4.1, our cost function C_{DACE} can quantitatively evaluate the reality of actions by taking into account feature-correlations and outlier risk. On the other hand, it has the following limitations:

- The MD included in C_{DACE} is a metric based on linear relationships between features [191, 225]. Therefore, C_{DACE} cannot accurately capture nonlinear rela-

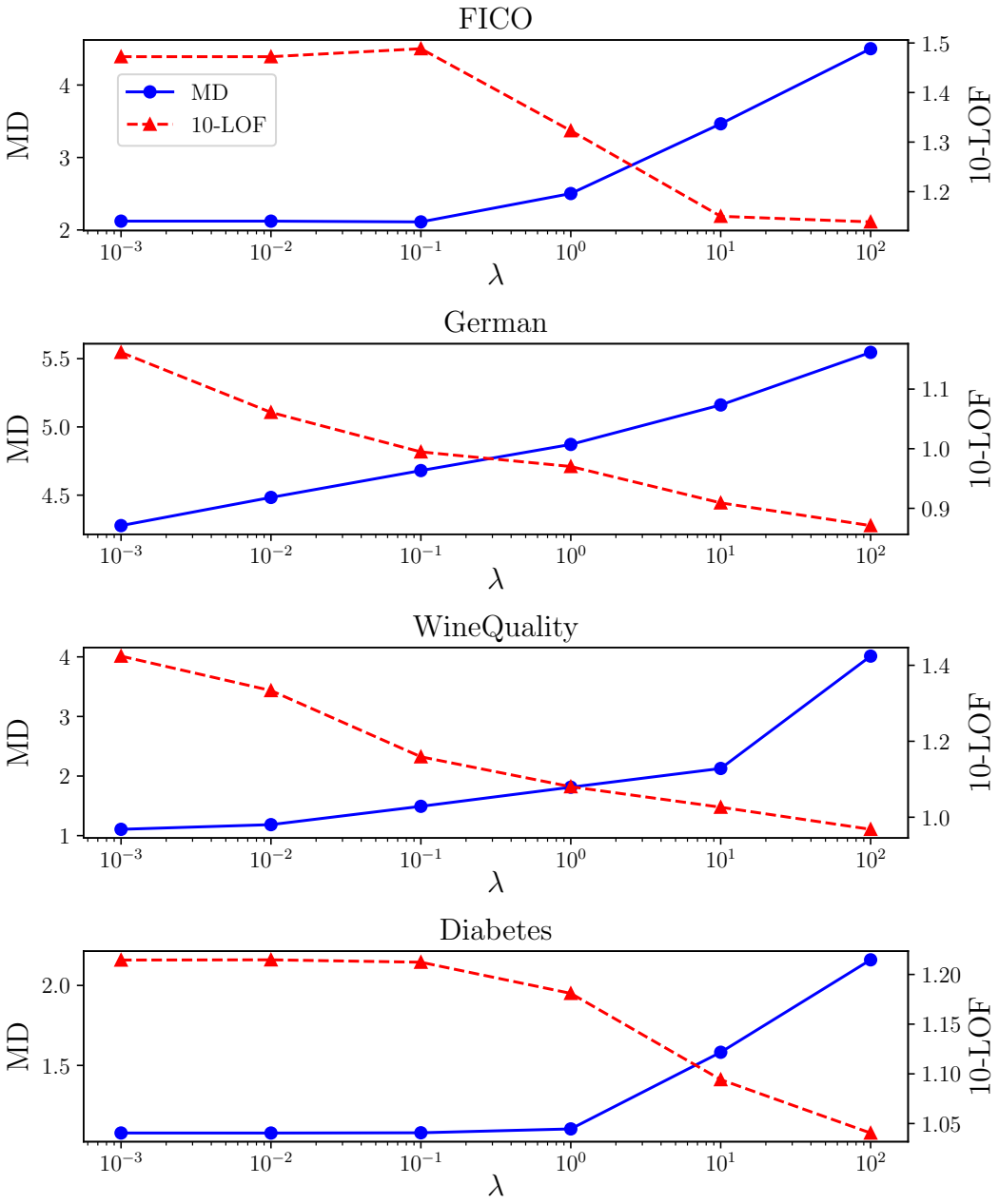


Figure 4.3: Sensitivity analyses of the trade-off parameter λ of DACE between the average MD and LOF.

tionships and asymmetric interactions, such as *causal effect*. In particular, the causal effect is one of the most important topics in recent studies on CE, and several methods that take into account causal effects between features have been proposed [174, 178, 226].

- Recently, a framework for enumerating distinct actions has been proposed to allow users to choose an action among the enumerated actions according to their preferences [242, 292, 335]. To enumerate actions, we must solve the optimization problem for extracting an action multiple times as a subroutine. Because C_{DACE} often requires more running time to be optimized than existing cost functions as shown in Table 4.4, we need to develop another efficient enumeration algorithm designated for C_{DACE}.

Future Work. In the future, we plan to devise a more efficient MILO formulation, and extend it to deal with other reliability criteria of CE, such as connectedness [202]. It would also be interesting to learn a matrix M that well reflects the properties of a given instance and data distribution [191]. To clarify in which situation it would be most effective to use our DACE, we also plan to conduct further detailed experiments on the situations where our assumptions do not hold, e.g., where correlations between features are low or nonlinear. It is also important for future work to conduct user-experiments to qualitatively evaluate the usability of actions extracted by our DACE.

Chapter 5

Ordered Counterfactual Explanation

Post-hoc explanation methods for machine learning models have been widely used to support decision-making. One of the popular methods is *Counterfactual Explanation (CE)*, also known as *Actionable Recourse*, which provides a user with a perturbation vector of features that alters the prediction result. Given a perturbation vector, a user can interpret it as an "*action*" for obtaining one's desired decision result. In practice, however, showing only a perturbation vector is often insufficient for users to execute the action. The reason is that if there is an asymmetric interaction among features, such as causality, the total cost of the action is expected to depend on the order of changing features. Therefore, practical CE methods are required to provide an appropriate order of changing features in addition to a perturbation vector. For this purpose, we propose a new framework called *Ordered Counterfactual Explanation (OrdCE)*. We introduce a new objective function that evaluates a pair of an action and an order based on feature interaction. To extract an optimal pair, we propose a mixed-integer linear

optimization approach with our objective function. Numerical experiments on real datasets demonstrated the effectiveness of our OrdCE in comparison with unordered CE methods.

5.1 Introduction

Complex machine learning models such as neural networks and tree ensembles are widely used in critical decision-making tasks (e.g., medical diagnosis and loan approval). Thus, post-hoc methods for extracting explanations from an individual prediction of these models have been attracting much attention for the last few years [189, 223, 280, 281]. To provide a user with a better insight into future improvement, a post-hoc method needs to show not only why undesirable predictions are given, but also how to act to obtain a desired prediction result [85, 233].

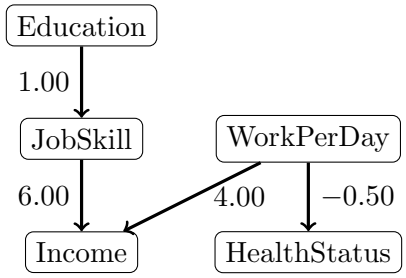
One of the post-hoc methods that show an action to obtain the desired outcome is *Counterfactual Explanation (CE)* [344], also known as *Actionable Recourse* [335]. Consider an example of a synthetic credit loan approval dataset shown in Figure 5.1. Imagine a situation that a user receives an undesired prediction $f(x^\circ) \neq y^\star$ for a target label $y^\star$ from a trained model f, which means denial of credit loan, on an instance x° related to one's current livelihood. We want to provide the user with advice a on changes of features such as "Income" and "JobSkill" so that the user can change one's current status x° to obtain the desired outcome $f(x^\circ + a) = y^\star$ [335].

To achieve this goal, most of the existing CE methods find a perturbation vector $a^* \in \mathcal{A}$, called an *action*, as an optimal solution of the following optimization problem:

$$a^* := \operatorname*{arg\,min}_{a \in \mathcal{A}} \; C_{\mathrm{dist}}(a \mid x^\circ) \quad \text{subject to } f(x^\circ + a) = y^\star,$$

	Name	Range
x_1	"Education"	$1, \ldots, 5$
x_2	"JobSkill"	$1, \ldots, 10$
x_3	"Income"	$10, \ldots, 100$
x_4	"WorkPerDay"	$4, \ldots, 10$
x_5	"HealthStatus"	$1, \ldots, 10$

(a) Features



(b) Causal DAG

Figure 5.1: Features and the causal DAG of our synthetic credit loan approval dataset. The task is to predict whether an individual's credit loan will be approved. We labeled each individual depending on one's values of "Income" and "HealthStatus".

where $\mathcal{A}$ is a set of feasible perturbation vectors, and C_{dist} is a cost function that measures the required efforts of a, such as TLPS [335] and DACE [171]. Table 5.1 presents examples of actions extracted from a logistic regression classifier trained on our credit approval dataset in Figure 5.1. A user can obtain one's desired outcome by changing each feature according to the suggested perturbation vectors in Table 5.1.

In practice, however, showing only a perturbation vector a^* is often insufficient for users to execute the action due to interaction among features [270]. In the previous example, as shown in the causal DAG in Figure 5.1(b), we have an asymmetric interaction "JobSkill" $\rightarrow$ "Income", which means that increasing one's "JobSkill" has a positive effect on increasing "Income" while the opposite does not. From these observations, we see that it is more reasonable to increase first "JobSkill" and then "Income" than the reverse order. Thus, practical CE methods are required to provide an appropriate order of changing features in addition to a perturbation vector a^*.

To achieve this requirement, we propose a novel CE framework that returns a pair (a^*, σ^*), called an *ordered action*, of a perturbation vector a^* and a permutation σ^*

of features that advises a user to change features in that order. We assume that the feature interaction is represented by an *interaction matrix* M, whose element indicates the linear interaction between two features, such as correlations, causal effects, or given by some prior knowledge. Roughly speaking, we consider the following optimization problem:

$$(a^*, \sigma^*) := \underset{a \in \mathcal{A}, \sigma \in \Sigma}{\arg\min}\ C_{\text{dist}}(a \mid x^\circ) + \gamma \cdot C_{\text{ord}}(a, \sigma \mid M)$$
$$\text{subject to } f(x^\circ + a) = y^\star,$$

where C_{ord} is a new cost function for determining an order of changing features, Σ is the set of all permutations of the features perturbed by a, and $\gamma > 0$ is a trade-off parameter.

5.1.1 Contributions

Our goal is to extend the CE framework so that it provides an ordered action by taking into account feature interaction. Our contributions are summarized as follows:

1. As a new framework for CE, we propose *Ordered Counterfactual Explanation (OrdCE)* that provides an *ordered action*, i.e., a pair of a perturbation vector and an order of changing features. For that purpose, we introduce a new objective function that evaluates the cost of an ordered action based on a given *interaction matrix* M.

2. We formulate the problem of finding an optimal ordered action as a *mixed-integer linear optimization (MILO)* problem, which can be efficiently solved by modern MILO solvers such as CPLEX [153]. Our formulation works on popular classifiers, such as linear models, tree ensembles, and multilayer perceptrons. In addition,

Method	Action		
	"Income"	"WorkPerDay"	"HealthStatus"
DACE	+4	+1	+3
TLPS	+5	0	0

Table 5.1: Examples of actions extracted by the existing CE methods on the credit approval dataset in Figure 5.1.

Method	Order	Feature	Action
OrdCE + DACE	1st	"HealthStatus"	+3
	2nd	"WorkPerDay"	+1
	3rd	"Income"	+4
OrdCE + TLPS	1st	"JobSkill"	+1
	2nd	"Income"	+6

Table 5.2: Examples of ordered actions extracted by our OrdCE on the credit approval dataset in Figure 5.1.

the formulation can be combined with any existing cost function used in MILO-based CE methods, such as TLPS [335] or DACE [171].

3. We conducted numerical experiments on real datasets and compared the performance of our OrdCE with previous CE methods. We confirmed that (i) our MILO approach obtained better ordered actions than baselines within practical computation time, and (ii) the obtained orders were reasonable from the perspective of feature interaction.

Table 5.2 presents examples of ordered actions extracted by OrdCE on the credit approval dataset in Figure 5.1. These orders of changing features are consistent with the causal DAG shown in Figure 5.1(b). For example, the ordered action extracted by OrdCE + DACE indicates increasing "WorkPerDay" before "Income". This order is more reasonable than its reverse order because "WorkPerDay" has a positive effect on "Income". In addition, in the result of OrdCE + TLPS, the total number of the changing features increases from that of the unordered TLPS in Table 5.1. However, because "JobSkill" has an effect 6.00 on "Income" as shown in Figure 5.1(b), changing "JobSkill" by +1 naturally causes an increase of "Income" by +6. Thus, it is expected that the user completes the ordered action suggested by OrdCE + TLPS with only changing "JobSkill" at the 1st step. Our OrdCE can find such an appropriate order by optimizing a perturbation vector and an order simultaneously. In summary, we see that our method presents a reasonable ordered action, which helps a user act to obtain the desired outcome.

5.1.2 Related Work

The existing CE methods can be categorized into gradient-based [178, 241, 344], autoencoder [81, 226], SAT [175], or mixed-integer linear optimization (MILO) [76, 171, 292, 335]. Since our cost function is non-differentiable due to the discrete nature of a permutation σ over features, we focus on MILO-based methods, which can directly handle such functions.

Most CE methods provide only a perturbation vector as an action. To the best of our knowledge, FACE [270] and OAS [277] are the only exceptions that consider a sequence of actions. FACE provides a sequence of training instances as a path from a given instance to the target instances in a neighborhood graph. However, FACE does not take into account feature interaction and does not determine the execution order of features. On the other hand, OAS provides a sequence of actions, which is related to classical planning [246]. There, the costs of candidate actions are static and irrelevant to their order, while in OrdCE, the costs dynamically depend on the previously chosen actions due to their interaction. It is also noteworthy that Bertsimas et al. studied how to determine an order of features to improve explainability in linear regression [37].

5.2 Problem Statement

5.2.1 Action and Ordered Action

Let $f\colon \mathcal{X} \to \mathcal{Y}$ and $x^\circ = (x_1^\circ, \ldots, x_D^\circ) \in \mathcal{X}$ be a classifier and a given instance such that $f(x^\circ) = -1$, respectively. We define an *action* for x° as a perturbation vector $a \in \mathbb{R}^D$ such that $f(x^\circ + a) = +1$. An *action set* $\mathcal{A} = A_1 \times \cdots \times A_D$ is a finite set of feasible actions such that $0 \in A_d$ and $A_d \subseteq \{a_d \in \mathbb{R} \mid x_d^\circ + a_d \in \mathcal{X}_d\}$ for $d \in [D]$.

As mentioned in Chapter 4, we can determine each A_d depending on the type and constraint of the feature $d \in [D]$. For example, a feature representing "Age" must be a positive integer and cannot be decreased. We define the *perturbing features* of an action a as $\mathrm{supp}(a) := \{d \in [D] \mid a_d \neq 0\}$. For $K \in [D]$, we write $\mathcal{A}_{\leq K} := \{a \in \mathcal{A} \mid |\mathrm{supp}(a)| \leq K\}$.

We introduce an *ordered action* for OrdCE. An ordered action is a pair of a perturbation vector $a \in \mathcal{A}_{=K}$ for some $K \in [D]$ and a permutation $\sigma = (\sigma_1, \ldots, \sigma_K) \in [D]^K$ of the perturbing features $\mathrm{supp}(a)$, which suggests perturbing the features $\mathrm{supp}(a)$ in that order. We denote by $\Sigma(a)$ the set of all permutations of $\mathrm{supp}(a)$, and call $\sigma \in \Sigma(a)$ a *perturbing order* of a.

5.2.2 Interaction Matrix

Practically, causal relationships are usually unknown in advance, and we need to estimate them. Since linear causal models can be estimated properly in practical settings where hidden common causes are included [303], we assume the feature interaction is linear. We assume that the feature interaction is represented by a matrix $M = (M_{i,j})_{1 \leq i,j \leq D}$, which we call an *interaction matrix*. Each element $M_{i,j}$ represents the linear interaction from i to j, that is, when we perturb a feature x_i to $x_i + a_i$, then x_j is perturbed to $x_j + M_{i,j} \cdot a_i$. We can compute $M_{i,j}$ explicitly with causal effect estimation methods [159, 262, 303] or some prior knowledge of domain experts.

From a given causal DAG estimated by, for example, DirectLiNGAM [152, 303], we can compute an interaction matrix as follows. Let $B = (B_{i,j})_{1 \leq i,j \leq D}$ be the adjacency matrix of the estimated causal DAG. By reordering the order of the nodes, we can assume that B is a strictly upper triangular matrix. Here, LiNGAM considers a model expressed the following structural equations: $x_j = \sum_{i \in \mathrm{pa}_B(j)} B_{i,j} \cdot x_i + e_j$, where e_j is

a continuous random variable that has non-Gaussian distributions and is independent of each other, and $\mathrm{pa}_B(j) \subseteq [D]$ is the set of features that are the ancestors of j on the estimated causal DAG. Then, we obtain

$$M = I + \sum_{k=1}^{D-1} B^k.$$

5.2.3 Cost Function

As a score to evaluate the required effort of an ordered action (a, σ), we introduce a new cost function C_{OrdCE} as follows. Given an input instance $x^\circ \in \mathcal{X}$, an interaction matrix M, and a trade-off parameter $\gamma \geq 0$, we define C_{OrdCE} for a pair of a perturbation $a \in \mathcal{A}$ and its order $\sigma \in \Sigma(a)$ as

$$C_{\mathrm{OrdCE}}(a, \sigma \mid x^\circ, M, \gamma) := C_{\mathrm{dist}}(a \mid x^\circ) + \gamma \cdot C_{\mathrm{ord}}(a, \sigma \mid M),$$

where C_{dist} and C_{ord} are *distance-based* and *ordering cost functions*, respectively. The former evaluates the required effort of a perturbation vector a, and the latter determines a perturbing order σ of a.

Distance-based Cost Function

As with the existing CE methods, we utilize a distance-based cost function C_{dist} to evaluate the required effort of an entire perturbation a. For simplicity, we assume C_{dist} as the following form:

$$C_{\mathrm{dist}}(a \mid x^\circ) = \sum_{d=1}^{D} \mathrm{dist}_d(a_d \mid x_d^\circ),$$

where $\mathrm{dist}_d \colon A_d \to \mathbb{R}_{\geq 0}$ is a cost measure of the feature d that evaluates the effort to change x_d° to $x_d^\circ + a_d$, such as the total-log percentile shift [335]. Note that our

optimization approach can adapt to other types of existing cost functions, such as ℓ_1-Mahalanobis' distance [171]. While these useful distance-based cost functions have been proposed, they do not deal with a perturbing order $\sigma \in \Sigma(a)$.

Ordering Cost Function

To extend CE so as to deal with a perturbing order $\sigma = (\sigma_1, \ldots, \sigma_K) \in \Sigma(a)$, we introduce an ordering cost function C_{ord} as the following form:

$$C_{\mathrm{ord}}(a, \sigma \mid M) = \sum_{k=1}^{K} \mathrm{cost}^{(k)}(a_{\sigma_1,\ldots,\sigma_k} \mid M),$$

where $a_{\sigma_1,\ldots,\sigma_k} := (a_{\sigma_1}, \ldots, a_{\sigma_k})$ and $\mathrm{cost}^{(k)}$ is a cost for each k-th perturbation a_{σ_k}. We define $\mathrm{cost}^{(k)}$ as a function that depends not only on the perturbation of σ_k at the k-th step but also on the previously perturbed features $\sigma_1, \ldots, \sigma_{k-1} \in \mathrm{supp}(a)$ since the actual amount of a perturbation of a feature is affected by the values of other features that interact with it. Note that we assume that the previously perturbed features are unaffected by the following perturbation, which is based on the *intervention* in causal models [262].

To define $\mathrm{cost}^{(k)}$, we introduce a parameter $\Delta^{(k)} = \Delta^{(k)}(a_{\sigma_1,\ldots,\sigma_k} \mid M)$, called the *actual perturbation*, as the amount of change on x_{σ_k} that we actually need to obtain a desired perturbation. Then, the resulting perturbation a_{σ_k} is equal to the sum of $\Delta^{(k)}$ and the effect of the previous perturbations $\Delta^{(1)}, \ldots, \Delta^{(k-1)}$. Formalizing this idea, we have the following conditions on $\Delta^{(k)}$ for $k = 1, 2, 3$:

$$\begin{aligned}
a_{\sigma_1} &= \Delta^{(1)}, \\
a_{\sigma_2} &= \Delta^{(2)} + M_{\sigma_1,\sigma_2} \cdot \Delta^{(1)}, \\
a_{\sigma_3} &= \Delta^{(3)} + M_{\sigma_2,\sigma_3} \cdot \Delta^{(2)} + M_{\sigma_1,\sigma_3} \cdot \Delta^{(1)}.
\end{aligned}$$

Generally, we have $a_{\sigma_k} = \Delta^{(k)} + \sum_{l=1}^{k-1} M_{\sigma_l,\sigma_k} \cdot \Delta^{(l)}$. For any $k \in [K]$, we inductively obtain

$$\Delta^{(k)}(a_{\sigma_1,\ldots,\sigma_k} \mid M) = a_{\sigma_k} - \sum_{l=1}^{k-1} M_{\sigma_l,\sigma_k} \cdot \Delta^{(l)}(a_{\sigma_1,\ldots,\sigma_l} \mid M).$$

By using $\Delta^{(k)}$, we define $\mathrm{cost}^{(k)}$ as follows:

$$\mathrm{cost}^{(k)}(a_{\sigma_1,\ldots,\sigma_k} \mid M) := \left|\Delta^{(k)}(a_{\sigma_1,\ldots,\sigma_k} \mid M)\right|.$$

In practice, since each feature has a different scale, we multiply each $\Delta^{(k)}$ by a scaling factor $s_{\sigma_k} > 0$, such as the inverse of its standard deviation.

5.2.4 Problem Definition

Our aim is to find an ordered action (a, σ) that minimizes the cost C_{OrdCE}. This problem can be defined as follows.

Problem 2. Given an additive classifier $f\colon \mathcal{X} \to \mathcal{Y}$, an input instance $x^\circ \in \mathcal{X}$ such that $f(x^\circ) = -1$, an action set $\mathcal{A}$, an interaction matrix $M \in \mathbb{R}^{D\times D}$, $K \in [D]$, and $\gamma \geq 0$, find an ordered action (a, σ) that is an optimal solution for the following optimization problem:

$$\begin{array}{ll} \underset{a\in\mathcal{A}_{\leq K},\sigma\in\Sigma(a)}{\text{minimize}} & C_{\mathrm{OrdCE}}(a, \sigma \mid x^\circ, M, \gamma) \\ \text{subject to} & f(x^\circ + a) = +1. \end{array}$$

By solving the above optimization problem, we obtain an ordered action (a, σ) that accords with feature interaction.

5.2.5 Concrete Examples

To study the cost function C_{ord} and objective function C_{OrdCE}, we present concrete examples to observe behavior of these functions in the same setting as Section 5.1.

Our synthetic credit loan approval dataset consists of five features $x_1, \dots x_5$ presented in Figure 5.1 in Section 5.1. As an interaction matrix, we take the matrix M whose element $M_{i,j}$ represents the average causal effect from a feature i to j. As described previously, $M = (M_{i,j})_{1 \leq i,j \leq 5}$ is calculated from the adjacency matrix of the causal DAG in Figure 5.1(b) as follows:

$$M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 1 & 1 & 6 & 0 & 0 \\ 0 & 1 & 6 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 4 & 1 & -0.5 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}.$$

In the examples below, we use C_{ord} with scaling factors $s_d > 0$ ($d \in [D]$):

$$C_{\text{ord}}(a, \sigma \mid M) = \sum_{k=1}^{K} s_{\sigma_k} \cdot \left| \Delta^{(k)}(a_{\sigma_1, \dots, \sigma_k} \mid M) \right|.$$

First, we show an example to observe the behavior of the ordering cost function C_{ord} in the following Example 1.

Example 1. Consider a perturbation $a = (0, 0, 4, 1, 3)$ and its perturbing orders $\sigma = (4, 3, 5)$ and $\sigma^\circ = (5, 4, 3)$. We compare the values of our ordering cost function C_{ord} for the two ordered actions (a, σ) and (a, σ°).

For (a, σ), the actual perturbations $\Delta^{(k)}$ can be calculated as follows:

$$\begin{aligned} \Delta^{(1)} &= 1 - 0 = 1, \\ \Delta^{(2)} &= 4 - M_{4,3} \cdot \Delta^{(1)} = 0, \\ \Delta^{(3)} &= 3 - M_{3,5} \cdot \Delta^{(2)} - M_{4,5} \cdot \Delta^{(1)} = 3.5. \end{aligned}$$

Thus, the value of C_{ord} for (a, σ) can be calculated as

$$\begin{aligned} C_{\mathrm{ord}}(a, \sigma \mid M) &= s_{\sigma_1} \cdot |\Delta^{(1)}| + s_{\sigma_2} \cdot |\Delta^{(2)}| + s_{\sigma_3} \cdot |\Delta^{(3)}| \\ &= s_4 + 3.5 s_5. \end{aligned}$$

Similarly, the value of C_{ord} for (a, σ°) can be calculated as

$$\begin{aligned} C_{\mathrm{ord}}(a, \sigma^\circ \mid M) &= s_{\sigma^\circ_1} \cdot |\Delta^{(1)}| + s_{\sigma^\circ_2} \cdot |\Delta^{(2)}| + s_{\sigma^\circ_3} \cdot |\Delta^{(3)}| \\ &= s_4 + 3 s_5. \end{aligned}$$

Because $s_d > 0$ for all $d \in \{1, \ldots, 5\}$, $C_{\mathrm{ord}}(a, \sigma \mid M) > C_{\mathrm{ord}}(a, \sigma^\circ \mid M)$ holds. □

In the above example, the ordered action (a, σ) suggests increasing the values of "WorkPerDay", "Income", and "HealthStatus" in this order. The other ordered action (a, σ°) suggests increasing the values of "HealthStatus", "WorkPerDay", and "Income" in this order. Since "WorkPerDay" has a negative causal effect to "HealthStatus", the perturbing order σ° is more reasonable than σ from the perspective of the feature interaction. The above example indicates that we can obtain an appropriate order of its perturbing features by minimizing C_{ord}.

Next, we show an example to observe the behavior of the objective cost function C_{OrdCE} in the following Example 2.

Example 2. Consider the same setting as Example 1 and two feasible ordered actions (a, σ) and (a°, σ°) with

$$\begin{aligned} a &= (0, 0, 6, 0, 0), \quad \sigma = (3), \\ a^\circ &= (0, 1, 6, 0, 0), \quad \sigma^\circ = (2, 3). \end{aligned}$$

We compare the values of our objective function C_{OrdCE} for the two ordered actions (a, σ) and (a°, σ°).

For (a, σ), the actual perturbations $\Delta^{(k)}$ can be calculated as follows:

$$\Delta^{(1)} = 6 - 0 = 6.$$

Thus, the value of C_{OrdCE} for (a, σ) can be calculated as

$$\begin{aligned} C_{\mathrm{OrdCE}}(a, \sigma \mid x^\circ, M, \gamma) &= c_3 + \gamma \cdot (s_{\sigma_1} \cdot |\Delta^{(1)}|) \\ &= c_3 + \gamma \cdot 6 s_3, \end{aligned}$$

where $c_3 = \mathrm{dist}_3(6 \mid x^\circ_3) > 0$.

Similarly, the value of C_{OrdCE} for (a°, σ°) can be calculated as

$$\begin{aligned} C_{\mathrm{OrdCE}}(a^\circ, \sigma^\circ \mid x^\circ, M, \gamma) &= c_2 + c_3 + \gamma \cdot (s_{\sigma^\circ_1} \cdot |\Delta^{(1)}| + s_{\sigma^\circ_2} \cdot |\Delta^{(2)}|) \\ &= c_2 + c_3 + \gamma \cdot s_2, \end{aligned}$$

where $c_2 = \mathrm{dist}_2(1 \mid x^\circ_2) > 0$.

Now we assume $6s_3 - s_2 > 0$. Then, we obtain the following result:

$$\begin{aligned} & C_{\mathrm{OrdCE}}(a, \sigma \mid x^\circ, M, \gamma) > C_{\mathrm{OrdCE}}(a^\circ, \sigma^\circ \mid x^\circ, M, \gamma) \\ \iff & \gamma \cdot (6s_3 - s_2) - c_2 > 0 \\ \iff & \gamma > \frac{c_2}{6s_3 - s_2}. \end{aligned}$$

Hence, if $6s_3 - s_2 > 0$ and $\gamma > c_2/(6s_3 - s_2)$, then $C_{\mathrm{OrdCE}}(a, \sigma \mid x^\circ, M, \gamma) > C_{\mathrm{OrdCE}}(a^\circ, \sigma^\circ \mid x^\circ, M, \gamma)$. □

In the above example, the ordered action (a, σ) suggests increasing only "Income". The other ordered action (a°, σ°) suggests increasing the values of "JobSkill" and "Income" in this order. The total number of the changing features of the latter ordered action is greater than that of the former. However, a user is expected to complete the

latter ordered action with only changing "JobSkill" since increasing one ' s "JobSkill " has a positive effect on increasing "Income " as mentioned in Section 5.1. From the above example, we see that we can adjust a trade-off between the required effort of an entire perturbation a and each step $\Delta^{(k)}$ by tuning the parameter γ.

5.3 MILO Formulation

In this section, we formulate our problem for finding an optimal ordered action as an MILO problem.

5.3.1 Basic Constraints

For $d \in [D]$, we set $A_d = \{a_{d,1}, \ldots, a_{d,I_d}\}$ and $a_{d,1} = 0$. First, we introduce binary variables $\pi_{d,i} \in \{0,1\}$ for $d \in [D]$ and $i \in [I_d]$, which indicate that $a_{d,i} \in A_d$ is selected ($\pi_{d,i} = 1$) or not ($\pi_{d,i} = 0$) as in the previous MILO-based methods [171, 335]. Then, $\pi_{d,i}$ must satisfy the following constraints:

$$\sum_{i=1}^{I_d} \pi_{d,i} = 1, \qquad \forall d \in [D]. \tag{5.1}$$

By using $\pi_{d,i}$, we can express a perturbation for each feature d as $a_d = \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}$. Note that $\pi_{d,1} = 1$ indicates that a feature d is not perturbed since $a_{d,1} = 0$.

To express an order of perturbing features, we introduce binary variables $\pi_{d,i}^{(k)} \in \{0,1\}$ for $d \in [D], i \in [I_d]$, and $k \in [K]$, which indicate $a_{d,i} \in A_d$ is selected as the k-th perturbation; that is, $\pi_{d,i}^{(k)} = 1$ if $a_{d,i} \in A_d$ is selected in the k-th step, and $\pi_{d,i}^{(k)} = 0$ otherwise. Then, $\pi_{d,i}^{(k)}$ must satisfy the following constraints:

$$\pi_{d,i} = \sum_{k=1}^{K} \pi_{d,i}^{(k)}, \qquad \forall d \in [D], \forall i \in [I_d]. \tag{5.2}$$

We also introduce binary variables $\sigma_{k,d}$ for $k \in [K]$ and $d \in [D]$ that indicate whether the feature d is perturbed in the k-th step. We impose the following constraints on $\sigma_{k,d}$:

$$\sigma_{k,d} = 1 - \pi^{(k)}_{d,1}, \qquad \forall k \in [K], \forall d \in [D], \tag{5.3}$$

$$\sum_{k=1}^{K} \sigma_{k,d} \leq 1, \qquad \forall d \in [D], \tag{5.4}$$

$$\sum_{d=1}^{D} \sigma_{k,d} \leq 1, \qquad \forall k \in [K], \tag{5.5}$$

$$\sum_{d=1}^{D} \sigma_{k,d} \geq \sum_{d=1}^{D} \sigma_{k+1,d}, \qquad \forall k \in [K-1]. \tag{5.6}$$

Constraint (5.4) allows that we can perturb each feature at most once. Constraint (5.5) imposes that we can perturb at most one feature in each step. Constraint (5.6) is a symmetry breaking constraint on $\sigma_{k,d}$.

5.3.2 Objective Function

Our objective function C_{OrdCE} consists of C_{dist} and C_{ord}, which we express with the program variables $\pi^{(k)}_{d,i}$ and $\sigma_{k,d}$.

Distance-based Cost Function

From our assumption of C_{dist}, it can be expressed as follows:

$$C_{\text{dist}}(a \mid x^{\circ}) = \sum_{d=1}^{D} \sum_{i=1}^{I_d} \sum_{k=1}^{K} c_{d,i} \cdot \pi^{(k)}_{d,i},$$

where $c_{d,i}$ is a constant value such that $c_{d,i} = \text{dist}_d(a_{d,i} \mid x^{\circ}_d)$. Note that our MILO formulation can adapt to other existing cost functions such as DACE [171] and SCM [226].

Ordering Cost Function

Since C_{ord} is non-linear due to a permutation σ, we need to express it by linear constraints of the variables. We introduce variables ζ_k for $k \in [K]$ such that $\zeta_k = |\Delta^{(k)}|$. Then, C_{ord} can be expressed as follows:

$$C_{\mathrm{ord}}(a, \sigma \mid M) = \sum_{k=1}^{K} \zeta_k.$$

Moreover, for $k \in [K]$ and $d \in [D]$, we introduce variables $\delta_{k,d} \in \mathbb{R}$ to express $\zeta_k = |\sum_{d=1}^{D} \delta_{k,d}|$. Then, $\delta_{k,d}$ must satisfy $\delta_{k,d} = \Delta^{(k)}$ if $\sigma_{k,d} = 1$, and $\delta_{k,d} = 0$ if $\sigma_{k,d} = 0$. We can linearize these non-linear constraints as follows:

$$\delta_{k,d} \geq \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}^{(k)} - \epsilon_{k,d} - U_{k,d} \cdot (1 - \sigma_{k,d}), \qquad \forall k \in [K], \forall d \in [D], \tag{5.7}$$

$$\delta_{k,d} \leq \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}^{(k)} - \epsilon_{k,d} - L_{k,d} \cdot (1 - \sigma_{k,d}), \qquad \forall k \in [K], \forall d \in [D], \tag{5.8}$$

$$L_{k,d} \cdot \sigma_{k,d} \leq \delta_{k,d} \leq U_{k,d} \cdot \sigma_{k,d}, \qquad \forall k \in [K], \forall d \in [D], \tag{5.9}$$

$$\epsilon_{k,d} = \sum_{l=1}^{k-1} \sum_{d'=1}^{D} M_{d',d} \cdot \delta_{l,d'}, \qquad \forall k \in [K], \forall d \in [D], \tag{5.10}$$

$$-\zeta_k \leq \sum_{d=1}^{D} \delta_{k,d} \leq \zeta_k, \qquad \forall k \in [K], \tag{5.11}$$

where $\epsilon_{k,d}$ is an auxiliary variable such that $\epsilon_{k,d} = \sum_{l=1}^{k-1} M_{\sigma_l,d} \cdot \Delta^{(l)}$ for $k \in [K]$ and $d \in [D]$. The constant values $L_{k,d}$ and $U_{k,d}$ are the lower and upper bounds of $\delta_{k,d}$. These values can be recursively computed from the interaction matrix M and the action set $\mathcal{A}$ as follows:

$$L_{k+1,d} = L_{k,d} - \max_{d' \in [D] \setminus \{d\}} \max_{\Delta \in \{L_{k,d'}, U_{k,d'}\}} M_{d',d} \cdot \Delta,$$

$$U_{k+1,d} = U_{k,d} - \min_{d' \in [D] \setminus \{d\}} \min_{\Delta \in \{L_{k,d'}, U_{k,d'}\}} M_{d',d} \cdot \Delta,$$

where $L_{1,d} = \min_{a_d \in A_d} a_d$ and $U_{1,d} = \max_{a_d \in A_d} a_d$.

5.3.3 Base Learner Constraints

We introduce variables $\xi_t \in \mathbb{R}$ for $t \in [T]$ such that $\xi_t = f_t(x^\circ + a)$, where f_t is the t-th base learner of f. From the definition of additive classifiers, the constraint $f(x^\circ + a) = +1$ is equivalent to the following linear constraint of ξ_t:

$$\sum_{t=1}^{T} w_t \cdot \xi_t \geq b. \tag{5.12}$$

We express the constraint $\xi_t = f_t(x^\circ + a)$ by linear constraints of ξ_t and $\pi_{d,i}$ because $f_t(x^\circ + a)$ depends on the value of a, i.e., the variables $\pi_{d,i}$. In the following, we show how to express ξ_t when f is a linear model (LM), tree ensemble (TE), or multilayer perceptron (MLP).

Linear Models

From the definition of LM, $T = D$ and $f_d(x^\circ + a) = x^\circ_d + a_d$ for $d \in [D]$. Hence, we can simply express the base learner of the LM as follows:

$$\xi_d = x^\circ_d + \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i}, \qquad \forall d \in [D]. \tag{5.13}$$

Tree Ensembles

Each base learner f_t of the TE is a decision tree. To express $\xi_t = f_t(x^\circ + a)$, we can utilize the following *decision logic constraint* [76, 171]:

$$\phi_{t,l} \in \{0, 1\}, \qquad \forall t \in [T], \forall l \in [L_t], \tag{5.14}$$

$$\sum_{l=1}^{L_t} \phi_{t,l} = 1, \qquad \forall t \in [T], \tag{5.15}$$

$$D \cdot \phi_{t,l} \leq \sum_{d=1}^{D} \sum_{i \in I_{t,l}^{(d)}} \pi_{d,i}, \qquad \forall t \in [T], \forall l \in [L_t], \tag{5.16}$$

$$\xi_t = \sum_{l=1}^{L_t} \hat{y}_{t,l} \cdot \phi_{t,l}, \qquad \forall t \in [T], \tag{5.17}$$

where $I_{t,l}^{(d)} = \{i \in [I_d] \mid x_d^\circ + a_{d,i} \in r_{t,l}^{(d)}\}$ and $r_{t,l}^{(d)}$ is the subspace of $\mathcal{X}_d$ such that $r_{t,l} = r_{t,l}^{(1)} \times \cdots \times r_{t,l}^{(D)}$.

Multilayer Perceptrons

Each base learner f_t of the MLP is an output of the t-th neuron with the ReLU activation function, i.e., $f_t(x + a) = \max\{0, w^{(t)}(x + a) + b^{(t)}\}$. Hence, we need to extract the positive part of $w^{(t)}(x + a) + b^{(t)}$ as the output of the t-th base learner ξ_t. To express it, we can utilize the following constraints proposed by Serra et al. [300]:

$$\nu_t \in \{0, 1\}, \bar{\xi}_t \geq 0, \qquad \forall t \in [T], \tag{5.18}$$

$$\xi_t \leq G_t \cdot \nu_t, \qquad \forall t \in [T], \tag{5.19}$$

$$\bar{\xi}_t \leq \bar{G}_t \cdot (1 - \nu_t), \qquad \forall t \in [T], \tag{5.20}$$

$$\xi_t = \bar{\xi}_t + \sum_{d=1}^{D} w_d^{(t)} \cdot \sum_{i=1}^{I_d} a_{d,i} \cdot \pi_{d,i} + F_t, \qquad \forall t \in [T], \tag{5.21}$$

where F_t, G_t, and $\bar{G}_t$ are constants such that $F_t = w^{(t)}x^\circ + b^{(t)}$, $G_t \geq \max_{a \in \mathcal{A}} w^{(t)}(x + a) + b^{(t)}$, and $\bar{G}_t \geq -\min_{a \in \mathcal{A}} w^{(t)}(x + a) + b^{(t)}$. The variable ν_t indicates whether $w^{(t)}(x + a) + b^{(t)}$ is positive, and $\bar{\xi}_t$ represents negative part of $w^{(t)}(x + a) + b^{(t)}$. Note that the above constraints can be extended to a general MLP with more than two hidden layers [300].

5.3.4 Overall Formulation

Finally, we show our overall formulation as follows:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{d=1}^{D}\sum_{i=1}^{I_d}\sum_{k=1}^{K} c_{d,i}\cdot\pi_{d,i}^{(k)} + \gamma\cdot\sum_{k=1}^{K}\zeta_k \\
\text{subject to} \quad & \text{Constraint (5.1–5.12)}, \\
& \begin{cases} \text{Constraint (5.13)}, & \text{if } H \text{ is a LM}, \\ \text{Constraint (5.14–5.17)}, & \text{if } H \text{ is a TE}, \\ \text{Constraint (5.18–5.21)}, & \text{if } H \text{ is a MLP}, \end{cases} \\
& \pi_{d,i}^{(k)} \in \{0,1\}, && \forall k \in [K], \forall d \in [D], \forall i \in [I_d], \\
& \sigma_{k,d} \in \{0,1\}, && \forall k \in [K], \forall d \in [D], \\
& \delta_{k,d}, \in \mathbb{R}, && \forall k \in [K], \forall d \in [D], \\
& \zeta_k \in \mathbb{R}, && \forall k \in [K].
\end{aligned}
\tag{5.22}
$$

As with the existing MILO-based methods [171, 292, 335], our formulation can be (i) handled by off-the-shelf MILO solvers, such as CPLEX [153], and (ii) customized by additional user-defined constraints, such as one-hot encoded categorical features and hard constraints for ordering (e.g., *precondition* [277]). In summary, we can obtain ordered actions that satisfy user-defined constraints without implementing designated algorithms.

For computational complexity, Problem (5.22) includes K times more variables and constraints than in the unordered one. Thus, solving (5.22) is equal to or more difficult than unordered ones. However, in the context of CE, sparse actions are preferred from

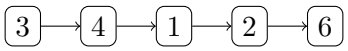
the perspective of interpretability [344]. Therefore, it is sufficient to choose small K for obtaining sparse actions.

5.3.5 Post-processing and Partially Ordered Actions

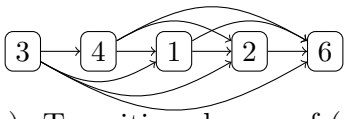
When some perturbing features have no interaction, changing the order of such features does not affect the cost C_{ord}. For example in Figure 5.1(b), the cost of the ordered action ["Education" $\rightarrow$ "WorkPerDay" $\rightarrow$ "JobSkill"] is the same as that of ["WorkPerDay" $\rightarrow$ "Education" $\rightarrow$ "JobSkill"] because "WorkPerDay" has no effect to the others. Thus the ordered action can be reduced to the partially ordered action ["WorkPerDay"] and ["Education" $\rightarrow$ "JobSkill"]. Suggesting such a partial order helps a user to execute an ordered action. We provide a post-processing algorithm that computes a partial order of the perturbing features from an ordered action and an interaction matrix.

An ordered action may be reduced to a *partially ordered action*, which is a pair $(a, \leq)$ of a perturbation vector $a \in \mathcal{A}$ and a partial order $\leq$ on $\mathrm{supp}(a)$. Here, we give a procedure to construct a partially ordered structure $\leq$ from an interaction matrix M and an ordered action (a, σ). If a perturbing order σ' of a is consistent with the obtained partial order $\leq$, the ordered action (a, σ') has the same ordering cost C_{ord} with that of (a, σ).

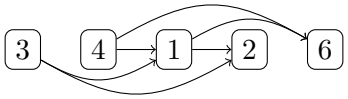
An ordered action can be expressed in the form of a path structure like Figure 5.2(a), where each node indicates a perturbing feature. In this path structure, changing a feature i should be executed after changing all its ancestors $\mathrm{pa}(i)$ and not after any its descendant. A partially ordered action is expressed in the form of a DAG like Figure 5.2(d), and a change in a feature satisfies the same condition as above. Note that even if a causal DAG is given, the DAG of a partially ordered action is not necessarily a subgraph of the causal DAG.



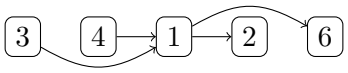
(a): Path structure for a given ordered action



(b): Transitive closure of (a)



(c): Remove the edges with no interaction from (b)



(d): Transitive reduction of (c)

Figure 5.2: Algorithm for obtaining a partially ordered action.

Algorithm

We give an algorithm to obtain a partially ordered action from an ordered action and an interaction matrix M. The procedure is as follows:

1. Construct a path that represents the perturbing order of a given ordered action.
2. Compute the transitive closure that represents the total order of the perturbing features.
3. Remove an edge from i to j if there is no interaction between i and j, that is, $M_{i,j} = M_{j,i} = 0$.
4. Compute the transitive reduction that represents a partially ordered action.

Here, a *transitive closure* of a directed graph G is a directed graph that has an edge from i to j if and only if there is a directed path from i to j in G. Also, a *transitive*

reduction of a directed graph G is a directed graph with the fewest number of edges whose transitive closure is the same as the transitive closure of G. For a finite DAG, its transitive closure and its transitive reduction are uniquely determined and can be computed in polynomial time [11, 120, 243].

We show that our algorithm can output a desired partial order of perturbing features for a given ordered action. For this purpose, we see that the ordering cost C_{ord} of an ordered action only depends on its partially ordered structure obtained by the above procedure. If i is not an ancestor of j on the DAG of a partially ordered action, then $M_{i,j} = M_{j,i} = 0$. Thus, the actual perturbation in the k-th step is calculated as follows:

$$\begin{aligned}\Delta^{(k)}(a_{\sigma_1,\ldots,\sigma_k} \mid M) &= a_{\sigma_k} - \sum_{l=1}^{k-1} M_{\sigma_l,\sigma_k} \cdot \Delta^{(l)}(a_{\sigma_1,\ldots,\sigma_l} \mid M) \\ &= a_{\sigma_k} - \sum_{\substack{l=1,\ldots,k-1 \\ \sigma_l \in \mathrm{pa}(\sigma_k)}} M_{\sigma_l,\sigma_k} \cdot \Delta^{(l)}(a_{\sigma_1,\ldots,\sigma_l} \mid M),\end{aligned}$$

where $\mathrm{pa}(j) \subseteq \mathrm{supp}(a)$ denotes the set of ancestors of a feature j on the DAG of the partially ordered action. That is, the ordered action (a, σ') has the same ordering cost C_{ord} as that of (a, σ) if σ' is consistent with the partially ordered structure obtained from (a, σ).

5.4 Experiments

In this section, we conducted experiments on real datasets to investigate the effectiveness and behavior of our **OrdCE**. All the code was implemented in Python 3.7 with scikit-learn and IBM ILOG CPLEX v12.10[1]. All the experiments were conducted on 64-bit macOS Catalina 10.15.6 with Intel Core i9 2.4GHz CPU and 64GB memory,

[1] All the code is available at `https://github.com/kelicht/ordce`.

and we imposed a 300 second time limit for solving.

5.4.1 Experimental Setting

We randomly split each dataset into the train (75%) and test (25%) instances, and trained ℓ_2-regularized logistic regression classifiers (LR), random forest classifiers (RF) with $T = 100$ decision trees, and two-layer ReLU network classifiers (MLP) with $T = 200$ neurons, on each training dataset. Then, we extracted ordered actions for test instances that had been received undesired prediction results, such as predicted as "high risk of default" from each classifier.

Distance-based Cost Functions

As a distance-based cost function C_{dist}, we used four existing cost functions: the total log-percentile shift (TLPS) [335], weighted ℓ_1-norm of median absolute deviation (MAD) [292, 344], ℓ_1-Mahalanobis' distance (DACE) [171], and distance based on structural causal models (SCM) [226]. The former two are *norm-based* cost functions that evaluate actions for each perturbing feature independently. The latter two are *interaction-aware* cost functions that evaluate actions by considering feature-correlation and causality, respectively.

Baseline Method

To the best of our knowledge, there is no existing method that determines a minimum-cost perturbing order. Even if an ordered action is consistent with a given causal DAG M, it is not necessarily optimal with respect to our objective function $C_{\text{OrdCE}}(a, \sigma \mid x^\circ, M, \gamma) = C_{\text{dist}}(a \mid x^\circ) + \gamma \cdot C_{\text{ord}}(a, \sigma \mid M)$, since C_{ord} depends not only on the causal

direction but also on the amount of the resultant perturbation. As a baseline, we proposed a greedy algorithm (**Greedy**), which consists of the following two steps:

1. Extract a perturbation vector $a^\star$ by optimizing C_{dist}.

2. Determine a perturbing order σ of a^* by solving the following optimization problem for k iteratively:

$$\sigma_k = \underset{d \in \text{supp}(a^\star) \setminus \{\sigma_1, \ldots, \sigma_{k-1}\}}{\arg\min} \left| a^\star_d - \sum_{l=1}^{k-1} M_{\sigma_l, d} \cdot \Delta^{(l)} \right|.$$

   This procedure greedily selects a perturbing feature that has the smallest cost in each step.

To compare **OrdCE** with **Greedy**, we measured the average values of the distance-based cost C_{dist}, the ordering cost C_{ord}, and the objective function C_{OrdCE} over the ordered actions obtained by those two methods.

5.4.2 Experimental Results

Comparison with Baseline

We used four real datasets: FICO ($D = 23$) [97], German ($D = 40$), WineQuality ($D = 12$), and Diabetes ($D = 8$) [88] datasets, where D is the number of features. For German dataset, we transformed each categorical feature into a one-hot encoded vector. For each dataset, we estimated the adjacency matrix of a causal DAG by the DirectLiNGAM algorithm [152, 303], and computed an interaction matrix M from the adjacency matrix (see Section 3). We set $\gamma = 1.0$ and $K = 4$.

Tables 5.3 and 5.4 present the average of the objective function C_{OrdCE} and the ordering cost C_{ord} for extracted ordered actions, respectively. From Tables 5.3 and 5.4,

C_{dist}	Dataset	Logistic Regression		Random Forest		Multilayer Perceptron	
		Greedy	**OrdCE**	**Greedy**	**OrdCE**	**Greedy**	**OrdCE**
TLPS	FICO	3.96 ± 2.6	**3.27 ± 2.1**	3.26 ± 2.9	**3.22 ± 3.2**	3.35 ± 3.0	**1.57 ± 1.4**
	German	4.9 ± 5.8	**4.81 ± 5.7**	3.23 ± 2.9	**3.2 ± 2.9**	5.38 ± 4.7	**5.03 ± 4.5**
	WineQuality	1.78 ± 1.8	**1.57 ± 1.5**	0.901 ± 0.55	**0.875 ± 0.52**	0.969 ± 0.83	**0.761 ± 0.61**
	Diabetes	2.91 ± 2.5	**2.47 ± 2.0**	2.3 ± 1.8	**2.26 ± 1.8**	1.12 ± 1.5	**0.668 ± 0.99**
DACE	FICO	10.6 ± 7.3	**9.61 ± 6.7**	6.78 ± 4.8	**6.67 ± 4.7**	3.5 ± 3.5	**3.41 ± 3.3**
	German	6.19 ± 5.3	**5.88 ± 4.9**	5.54 ± 4.6	**5.42 ± 4.5**	7.0 ± 5.8	**6.7 ± 5.4**
	WineQuality	2.93 ± 2.0	**2.42 ± 1.6**	1.65 ± 1.2	**1.51 ± 1.1**	1.91 ± 1.5	**1.66 ± 1.3**
	Diabetes	2.56 ± 1.7	**2.43 ± 1.6**	2.38 ± 1.7	**2.21 ± 1.6**	0.832 ± 1.2	**0.766 ± 1.1**
MAD	FICO	3.19 ± 2.1	**3.07 ± 2.0**	2.94 ± 2.7	**2.85 ± 1.9**	1.09 ± 0.97	**1.07 ± 0.98**
	German	5.90 ± 5.0	**2.52 ± 1.6**	6.70 ± 8.8	**1.94 ± 1.6**	6.53 ± 6.8	**5.14 ± 4.2**
	Wine	1.44 ± 1.0	**1.43 ± 1.0**	0.982 ± 0.59	**0.966 ± 0.58**	0.872 ± 0.68	**0.865 ± 0.68**
	Diabetes	**1.82 ± 1.2**	**1.82 ± 1.2**	1.34 ± 1.2	**1.31 ± 1.1**	0.695 ± 0.71	**0.683 ± 0.70**
SCM	FICO	3.96 ± 3.1	**3.41 ± 2.5**	3.80 ± 2.9	**3.53 ± 2.8**	2.23 ± 2.0	**2.09 ± 1.8**
	German	3.60 ± 3.2	**2.95 ± 2.4**	2.05 ± 2.0	**1.73 ± 1.6**	3.99 ± 3.2	**3.42 ± 2.8**
	Wine	2.32 ± 1.6	**1.75 ± 1.2**	1.43 ± 0.95	**1.32 ± 0.90**	1.21 ± 0.9	**1.03 ± 0.75**
	Diabetes	**1.78 ± 1.2**	**1.78 ± 1.2**	1.34 ± 1.2	**1.21 ± 1.0**	0.629 ± 0.87	**0.549 ± 0.74**

Table 5.3: Average objective value on the real datasets.

C_{dist}	Dataset	Logistic Regression		Random Forest		Multilayer Perceptron	
		Greedy	**OrdCE**	**Greedy**	**OrdCE**	**Greedy**	**OrdCE**
TLPS	FICO	2.21 ± 1.6	**1.33 ± 0.87**	1.72 ± 1.5	**1.49 ± 1.2**	3.05 ± 2.8	**0.84 ± 0.74**
	German	1.85 ± 1.5	**1.71 ± 1.4**	1.5 ± 1.2	**1.47 ± 1.2**	2.27 ± 1.8	**1.76 ± 1.6**
	WineQuality	1.0 ± 0.98	**0.765 ± 0.59**	0.475 ± 0.31	**0.439 ± 0.29**	0.69 ± 0.63	**0.446 ± 0.35**
	Diabetes	1.74 ± 1.6	**1.01 ± 0.81**	0.939 ± 0.67	**0.883 ± 0.64**	0.862 ± 1.3	**0.318 ± 0.58**
DACE	FICO	3.79 ± 2.6	**2.41 ± 1.8**	2.14 ± 1.5	**1.59 ± 1.2**	1.24 ± 1.2	**0.918 ± 0.86**
	German	1.92 ± 1.8	**1.43 ± 1.1**	1.6 ± 1.3	**1.46 ± 1.2**	2.23 ± 2.0	**1.87 ± 1.5**
	WineQuality	1.31 ± 0.94	**0.781 ± 0.53**	0.716 ± 0.54	**0.503 ± 0.34**	0.796 ± 0.69	**0.509 ± 0.41**
	Diabetes	1.2 ± 0.83	**1.02 ± 0.7**	1.13 ± 0.84	**0.912 ± 0.67**	0.425 ± 0.63	**0.322 ± 0.47**
MAD	FICO	1.41 ± 0.91	**1.28 ± 0.80**	1.45 ± 1.1	**1.37 ± 0.84**	0.562 ± 0.50	**0.526 ± 0.50**
	German	5.33 ± 4.8	**1.52 ± 1.1**	5.99 ± 8.5	**0.975 ± 1.2**	3.48 ± 4.7	**1.64 ± 1.3**
	Wine	0.757 ± 0.54	**0.750 ± 0.54**	0.497 ± 0.32	**0.470 ± 0.30**	0.449 ± 0.35	**0.439 ± 0.35**
	Diabetes	**0.890 ± 0.60**	**0.890 ± 0.60**	0.636 ± 0.55	**0.588 ± 0.51**	0.303 ± 0.4	**0.283 ± 0.39**
SCM	FICO	1.98 ± 1.6	**1.32 ± 0.84**	1.90 ± 1.5	**1.49 ± 1.0**	1.08 ± 1.0	**0.813 ± 0.73**
	German	2.02 ± 1.9	**1.35 ± 1.1**	1.11 ± 1.2	**0.796 ± 0.73**	2.16 ± 1.8	**1.59 ± 1.3**
	Wine	1.34 ± 0.95	**0.769 ± 0.54**	0.802 ± 0.57	**0.583 ± 0.41**	0.645 ± 0.5	**0.455 ± 0.35**
	Diabetes	0.890 ± 0.60	**0.889 ± 0.60**	0.716 ± 0.66	**0.577 ± 0.50**	0.349 ± 0.5	**0.264 ± 0.36**

Table 5.4: Average ordering cost on the real datasets.

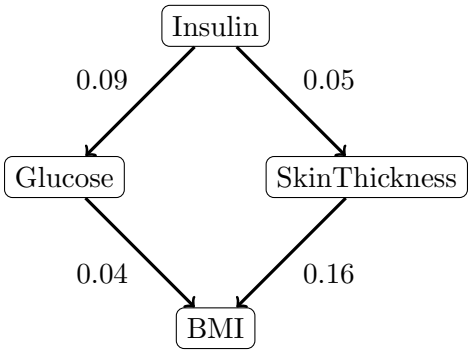


Figure 5.3: Subgraph of the causal DAG of the Diabetes dataset estimated by the DirectLiNGAM algorithm [152, 303].

we can observe that **OrdCE** always achieved lower C_{OrdCE} and C_{ord} than **Greedy** for all datasets and classifiers. Especially, in MLP and TLPS on FICO dataset, the averages of C_{OrdCE} and C_{dist} given by **OrdCE** are 1.57 and 0.84, respectively, which were less than half of those obtained by **Greedy**.

Next, we examine the ordered actions given by **OrdCE** to confirm the practicality. Table 5.5 presents examples of ordered actions extracted from the RF classifier on the Diabetes dataset, and Figure 5.3 presents a subgraph of the estimated causal DAG of the dataset. In both cases of TLPS and DACE, our **OrdCE** output ordered actions that accords with the directed edges in the causal DAG in Figure 5.3. On the other hand, the action extracted by **Greedy** with DACE, the order is not consistent with the causal DAG. From these results, we confirmed that **OrdCE** succeeded in obtaining a reasonable perturbing order from the perspective of the causal DAG. In addition, for TLPS, the perturbation given by **OrdCE** is different from that given by **Greedy**. This difference is caused by the effect that **OrdCE** optimizes a perturbation vector and its order simultaneously.

Regarding the computation time, **OrdCE** was certainly slower than **Greedy** be-

Method	**Order**	**Feature**	**Action**	C_{dist}	C_{ord}
Greedy	1st	"BMI"	-6.25	**0.778**	0.828
OrdCE	1st	"Glucose"	-3.0	0.825	**0.749**
	2nd	"BMI"	-5.05		

(a) TLPS [335]

Method	**Order**	**Feature**	**Action**	C_{dist}	C_{ord}
Greedy	1st	"BMI"	-0.8	**0.716**	0.825
	2nd	"SkinThickness"	-2.5		
	3rd	"Glucose"	-8.5		
	4th	"Insulin"	-32.0		
OrdCE	1st	"Insulin"	-32.0	**0.716**	**0.528**
	2nd	"Glucose"	-8.5		
	3rd	"SkinThickness"	-2.5		
	4th	"BMI"	-0.8		

(b) DACE [171]

Table 5.5: Examples of ordered actions extracted from the RF classifier on the Diabetes dataset.

C_{dist}	Dataset	Logistic Regression		Random Forest		Multilayer Perceptron	
		Greedy	**OrdCE**	**Greedy**	**OrdCE**	**Greedy**	**OrdCE**
TLPS	FICO	0.110 ± 0.023	12.3 ± 15	21.1 ± 50	125 ± 130	12.6 ± 48	183 ± 140
	German	0.0226 ± 0.0052	0.528 ± 0.57	10.6 ± 3.8	15.6 ± 8.8	0.154 ± 0.08	1.44 ± 0.91
	Wine	0.0682 ± 0.013	2.32 ± 4.5	8.82 ± 3.8	25.9 ± 48	0.354 ± 0.16	4.88 ± 13
	Diabetes	0.0339 ± 0.0046	0.720 ± 0.52	6.18 ± 7.4	42.4 ± 60	0.290 ± 0.1	4.81 ± 12
DACE	FICO	21.6 ± 22	277 ± 75	146 ± 78	287 ± 48	190 ± 140	233 ± 120
	German	0.260 ± 0.15	3.33 ± 5.3	30.2 ± 15	138 ± 180	0.778 ± 0.37	6.25 ± 9.8
	Wine	2.04 ± 0.81	150 ± 120	137 ± 110	254 ± 89	11.9 ± 21	121 ± 120
	Diabetes	0.245 ± 0.077	22.8 ± 52	25.4 ± 26	217 ± 120	2.27 ± 3.1	50.2 ± 100
MAD	FICO	0.0981 ± 0.021	8.05 ± 9.1	18.3 ± 27	117 ± 120	13.3 ± 32	151 ± 140
	German	0.0169 ± 0.0034	0.579 ± 0.54	8.09 ± 1.3	81.2 ± 110	0.0892 ± 0.058	5.15 ± 7.6
	Wine	0.0764 ± 0.019	1.16 ± 0.49	12.6 ± 5.5	40.2 ± 68	0.319 ± 0.12	8.13 ± 29
	Diabetes	0.0274 ± 0.0041	0.449 ± 0.13	8.03 ± 8.2	37.4 ± 61	0.253 ± 0.083	3.77 ± 8.3
SCM	FICO	0.564 ± 0.35	212 ± 120	96.1 ± 80	275 ± 70	175 ± 140	231 ± 120
	German	0.0360 ± 0.0085	0.829 ± 0.73	13.2 ± 8.7	63.4 ± 48	0.133 ± 0.09	1.61 ± 1.6
	Wine	0.475 ± 0.19	106 ± 110	123 ± 100	263 ± 82	1.92 ± 1.2	91.4 ± 100
	Diabetes	0.0450 ± 0.021	2.10 ± 5.1	18.1 ± 14	143 ± 120	0.806 ± 0.53	42.4 ± 98

Table 5.6: Computational times on the real datasets.

cause **OrdCE** exactly solved Problem 2. The result of computation time for each distance-based cost function, i.e., TLPS, DACE, MAD, and SCM, is shown in Table 5.6. In MLP and TLPS on FICO dataset, the average computation times of **OrdCE** and **Greedy** are 183 and 12.6 seconds, respectively. For other datasets, **OrdCE** is within 1.38–120 times slower than **Greedy**. However, Tables 5.3 to 5.5 indicate that **OrdCE** found better ordered actions in terms of C_{OrdCE} and C_{ord} than **Greedy** within 300 seconds, which is a reasonable computation time.

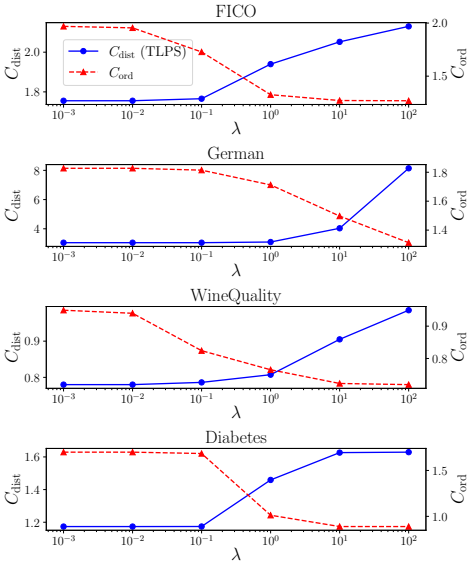
Sensitivity Analysis of Trade-off Parameter

To examine the sensitivity of the trade-off parameter γ, we observed C_{dist} and C_{ord} of ordered actions extracted from LR classifiers by varying γ. Figure 5.4 presents the average values of C_{dist} and C_{ord} for each value of γ and each dataset. We can see trade-off relationship between C_{dist} and C_{ord}. As mentioned in previous work on CE [242,344], suggesting multiple actions is helpful for diversity. By varying γ, we can obtain several distinct ordered actions that have diverse characteristics in terms of C_{dist} and C_{ord}.

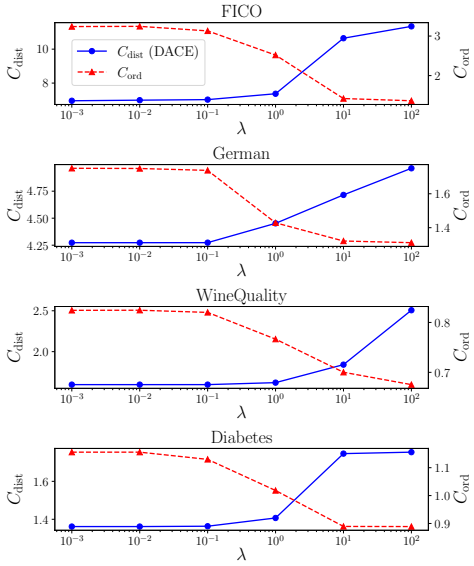
5.5 Conclusion

We proposed Ordered Counterfactual Explanation (OrdCE) that provides an optimal pair of a perturbation vector and an order of the features to be perturbed. We introduced a new objective function that evaluates the required cost of a perturbation vector and an order, and proposed a MILO formulation for optimizing it. By experiments on real datasets, we confirmed the effectiveness of our method by comparing it with a greedy method.

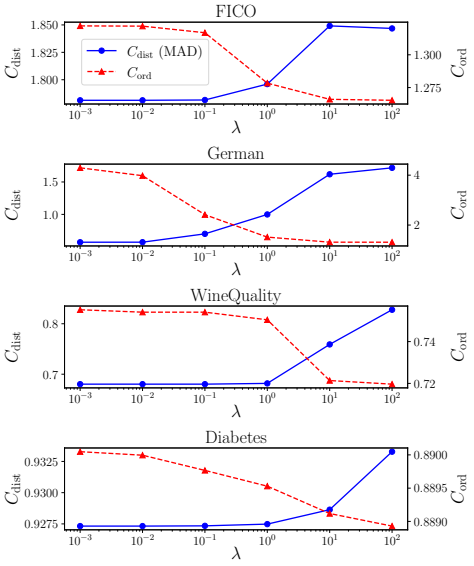
As future work, we plan to conduct user-experiments to evaluate the usability



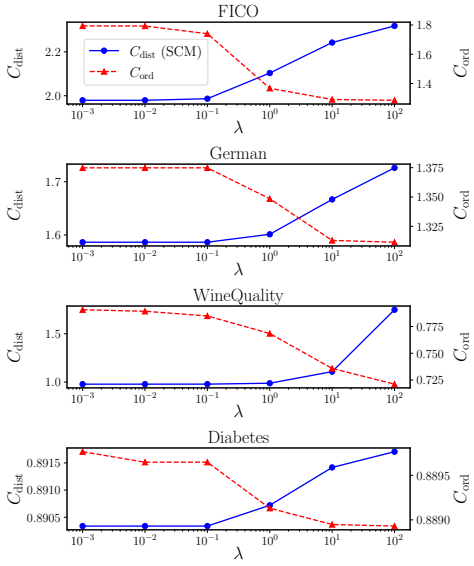
(a) TLPS [335]



(b) DACE [171]



(c) MAD [292, 344]



(d) SCM [226]

Figure 5.4: Sensitivity analyses of the trade-off parameter γ of **OrdCE** between the average C_{dist} and C_{ord}.

of our OrdCE. In this study, while we assume the causal relationship is linear, our cost function has the potential to deal with non-linear relationships, which sometimes appear in the real world [262]. Therefore, it is also interesting for future work to develop a method for optimizing our cost function with non-linear causal relationships.

Chapter 6

Fairness-Aware Decision Tree Editing

In the application of machine learning models to decision-making tasks (e.g., loan approval), *fairness* of their predictions has emerged as an important topic in recent years. If decision-makers detect unfairness in their models during deployment, they must modify the models to satisfy constraints on a specific discrimination criterion. However, simply retraining a model from scratch under fairness constraints may raise serious reliability issues caused by differences in prediction and interpretation between the initial model and retrained model. In this chapter, we propose a post-processing framework, named *Fairness-Aware Decision tree Editing (FADE)*, that converts a given biased decision tree into a fair decision tree without significantly changing it in terms of its prediction and interpretation. For this purpose, we introduce two dissimilarity measures between decision trees based on the prediction discrepancy and edit distance. We propose a mixed-integer linear optimization formulation for minimizing the dissimilarity measures under fairness constraints. Numerical experiments on real datasets

demonstrate the effectiveness of our method in comparison with existing methods.

6.1 Introduction

In the application of machine learning models to decision-making tasks, such as loan approval, there have been increasing concerns about the *fairness* of the predictions [127]. If the predictions are unfair, which means that there is *disparity* of the predictions between groups defined by *sensitive attributes* (e.g., gender or race) [29], the models are no longer usable in decision-making tasks, even if they achieve high accuracy. To avoid such disparity of predictions and achieve model fairness, a number of methods for evaluating the disparity [4, 55, 136, 366] and training fair models [167, 168, 365] have been proposed.

If decision-makers identify unfairness in their deployed model, they must modify the model so as to satisfy constraints on specific discrimination criteria, such as *demographic parity (DP)* [55]. However, in various situations of real applications (e.g., *dataset shift* [276]), there still remain many difficulties in eliminating the disparity from deployed models [348]. In the following, we demonstrate that simply retraining a model from scratch under fairness constraints raises serious reliability issues. We consider an *out-of-sample deployment scenario* [335], which is a special case of the dataset shift, on the German dataset [88]. A decision-maker deploys the decision tree presented in Figure 6.1(a) to predict whether individuals will default on their loan. However, because the training dataset does not include individuals under 25 years old, the decision tree turns out to be unfair to young individuals in terms of DP. The decision-maker then retrains it from scratch to be fair with respect to individuals' age using an existing method [167]. Although the retrained decision tree presented in Figure 6.1(b) achieves

lower DP than the initial decision tree, we argue that there are serious reliability issues from the following two perspectives:

- **Prediction:** To satisfy fairness constraints, the prediction results of some individuals must be changed. Previous research indicated a risk of prediction conflicts between an initial model and retrained model [229, 332]. In Figure 6.1, 26% of individuals receive different prediction results from the retrained decision tree than the initial decision tree.

- **Interpretation:** There are often representational differences between the initial decision tree and retrained decision tree, which implies differences in their interpretations. These differences may cause problems regarding the vulnerability of model interpretation [122], such as *fairwashing* [12]. In Figure 6.1, the prediction rules of four out of seven nodes in the retrained decision tree differ from those of the initial decision tree.

Based on the above observations, we aim to edit a given biased decision tree h into a fair decision tree h^* without significantly changing it in terms of its prediction and interpretation. Roughly speaking, we consider the following optimization problem:

$$\underset{h^*}{\text{minimize}} \quad \Gamma(h, h^*) \quad \text{subject to} \quad \Delta_z(h^*) \leq \theta,$$

where Γ is a dissimilarity measure between decision trees, Δ_z is a discrimination criterion, and θ is a discrimination threshold. Our research goals are (i) to introduce dissimilarity measures Γ between decision trees based on the above two perspectives, and (ii) to develop a method for minimizing $\Gamma(h, h^*)$ under fairness constraints on $\Delta_z(h^*)$.

6.1.1 Contributions

In this chapter, we propose a post-processing framework, named *Fairness-Aware Decision tree Editing (FADE)*, that edits the prediction rules of nodes of a given trained decision tree to satisfy fairness constraints without significantly changing the decision tree. Our contributions can be summarized as follows:

- To evaluate the degree of change from a given decision tree to its edited one, we introduce two dissimilarity measures between decision trees based on the *prediction discrepancy* [229] and *edit distance* [322, 372].

- We formulate the problem of finding an optimal edit operation according to the proposed dissimilarity measures under fairness constraints as a *mixed-integer linear optimization (MILO)* problem, which can be efficiently solved by off-the-shelf MILO solvers, such as CPLEX [153].

- By experiments on real datasets, we confirmed that decision trees edited by our method achieve comparable accuracy and fairness with a few edits to that of decision trees trained by an existing method for learning fair decision trees [167].

Figure 6.1(c) presents a decision tree edited by the proposed FADE method. The ratio of conflicting predictions is reduced to 15%, and only one node in the initial decision tree (Figure 6.1(a)) is edited. This result suggests that (i) FADE can transform a biased decision tree into a fair one with several changes in terms of its prediction and interpretation, and (ii) the edited decision tree achieves comparable accuracy and fairness to the retrained decision tree.

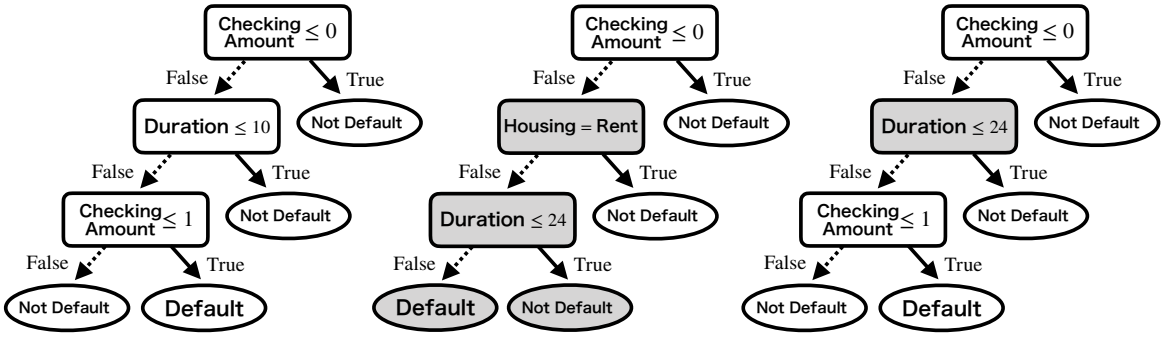
6.1.2 Related Work

Fairness in Machine Learning. According to Hajian [127], existing methods for achieving model fairness can be divided into three groups: *pre-processing* [57, 166], *in-processing* [167, 168, 365], and *post-processing* [56, 348] approaches. Our method is most closely related to post-processing approaches, which adjust already trained models under fairness constraints based on a discrimination criterion, such as *demographic parity* [55] or *equal opportunity* [136]. In particular, Kamiran et al. proposed a post-processing algorithm that modifies the predictive labels of leaves of a trained decision tree to satisfy fairness constraints [167]. Our approach can be regarded as an extension of their approach, as we edit not only leaves, but also internal nodes.

Decision Trees. With respect to *interpretability*, a decision tree is one of the most standard interpretable models. Top-down greedy approaches (e.g., CART [49]) are popular methods for training decision trees. In contrast, several non-greedy methods that are either based on MILO [8, 38, 341] or branch-and-bound algorithms [10, 149] have recently been proposed. Our approach is based on MILO methods, and can be regarded as their extension to determine how to edit an already trained decision tree to be fair. It should also be noted that Carreira et al. proposed a modifying method for decision trees that aims to enhance their accuracy and interpretability [58]. However, in contrast, our method aims to enhance their fairness.

6.2 Preliminaries

In this chapter, as in most studies on decision trees [10, 149], we assume that all features are binary-valued; that is, $\mathcal{X} = \{0, 1\}^D$. For a continuous feature, we can



Acc . : 0.719 DP : 0.147 Acc . : 0.714 DP : 0.021 Acc . : **0 . 720** DP : **0 . 004**

(*a*) Initial DT (*b*) Retrained DT (*c*) Edited DT

Figure 6.1: Decision trees (DTs) obtained for the German dataset. Acc. and DP denote the accuracy and demographic parity with respect to individuals' age on the dataset, respectively.

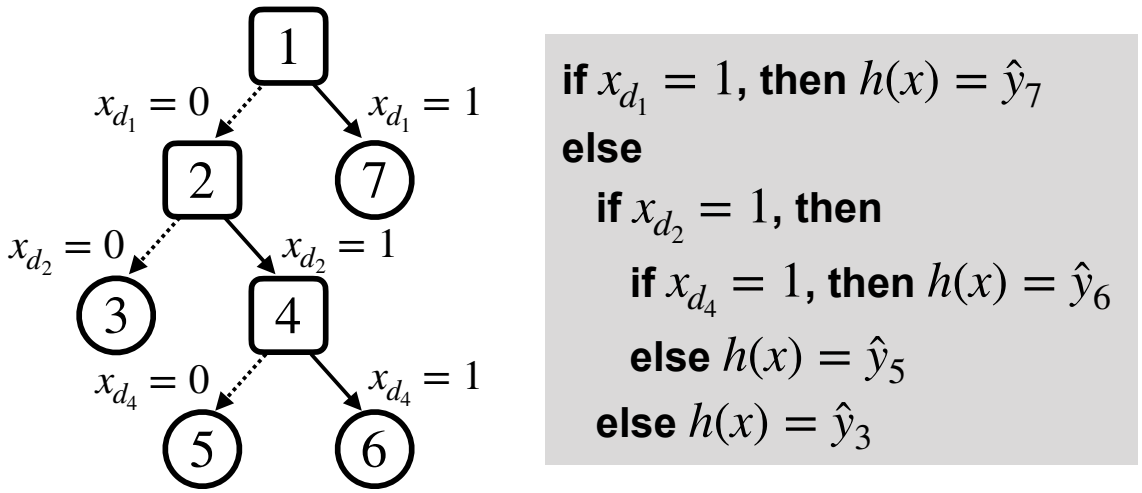


Figure 6.2: Example of decision tree h and its prediction rule, where $\mathcal{I} = \{1, 2, 4\}$ and $\mathcal{L} = \{3, 5, 6, 7\}$. For a leaf $l = 5$, $A_5 = \{1, 2, 4\}$, $A_5^{(\mathrm{L})} = \{1, 4\}$, and $A_5^{(\mathrm{R})} = \{2\}$.

encode it into some binary features by the existing discretization techniques, such as *bucketization* [10, 149].

6.2.1 Another Expression of Decision Trees

As shown in Chapter 2, a *decision tree* $h\colon \mathcal{X} \to \mathcal{Y}$ is a classifier that consists of a set of if-then-else rules expressed as a binary tree structure [49]. It makes a prediction according to the *predictive label* $\hat{y} \in \mathcal{Y}$ of the leaf that an input instance $x \in \mathcal{X}$ reaches. The corresponding leaf is determined by traversing the tree from the root depending on whether the statement $x_d = 1$ is true or not, where $d \in [D]$ is a *branching feature* of each internal node. Figure 6.2 presents an example of a decision tree.

We denote the set of all nodes in a decision tree h by $\mathcal{T} = \{1, \ldots, T\}$, where the nodes are indexed by a pre-order traversal. Let $\mathcal{L}, \mathcal{I} \subseteq \mathcal{T}$ represent sets of internal nodes and leaves in h, respectively. Note that $|\mathcal{L}| = |\mathcal{I}| + 1$ holds since h is a binary tree. For a node $t \in \mathcal{T}$, we denote the set of internal nodes on the path from the root to t by $A_t \subseteq \mathcal{I}$. Then, h can be expressed as follows:

$$h(x) = \sum_{l \in \mathcal{L}} \hat{y}_l \cdot \phi_l(x),$$

where $\hat{y}_l \in \mathcal{Y}$ is the *predictive label* of a leaf l and $\phi_l\colon \mathcal{X} \to \{0, 1\}$ is the *leaf indicator function* that indicates whether an input x reaches l. We can express ϕ_l as

$$\phi_l(x) = \mathbb{I}\left[\left(\bigwedge\nolimits_{i \in A_l^{(\mathrm{L})}} (x_{d_i} = 0)\right) \wedge \left(\bigwedge\nolimits_{i \in A_l^{(\mathrm{R})}} (x_{d_i} = 1)\right)\right],$$

where $d_i \in [D]$ is the *branching feature* of an internal node i, and $A_l^{(\mathrm{L})}, A_l^{(\mathrm{R})} \subseteq A_l$ are the sets of left and right branching nodes on the path from the root to l, respectively. We denote a set of decision trees $\mathcal{H} \subseteq \{h \mid h\colon \mathcal{X} \to \mathcal{Y}\}$.

6.2.2 Discrimination Scores

In the context of fairness-aware machine learning, we consider a *sensitive attribute* $z \in \{0, 1\}$, such as gender or race [127], in addition to the input domain $\mathcal{X}$ and output domain $\mathcal{Y}$. In this chapter, let a tuple (x, y, z) of an input instance $x = (x_1, \ldots, x_D) \in \mathcal{X}$, output label $y \in \mathcal{Y}$, and sensitive attribute $z \in \{0, 1\}$ be an *example*, and the set $S = \{(x^{(n)}, y^{(n)}, z^{(n)})\}_{n=1}^{N}$ be a *sample* with N examples. The aim of fairness-aware machine learning is to obtain a classifier $h \colon \mathcal{X} \to \mathcal{Y}$ whose predictions are fair with respect to the sensitive attribute z.

To evaluate the fairness of a classifier h with respect to the sensitive attribute z, we focus on *demographic parity (DP)* [55], which is a major discrimination criterion based on statistical parity. Our method can also be extended to other criteria based on statistical parity, such as *equal opportunity* [136]. The values of these criteria approach 1 if the predictions of h tend to be unfair for z, and they approach 0 if the predictions tend to be fair.

We say that a classifier h satisfies DP if $P(H(x) = 1 \mid z = 1) = P(H(x) = 1 \mid z = 0)$, where P is the probability over the joint distribution on $(z, h(x))$. Because we cannot observe, we instead define the DP score $\Delta_z \colon \mathcal{H} \to [0, 1]$ using the empirical probability $\hat{P}$ on a given sample S as follows:

$$\Delta_z(h \mid S) := \left| \hat{P}(h(x) = 1 \mid z = 1) - \hat{P}(h(x) = 1 \mid z = 0) \right|.$$

6.3 Problem Statement

In this section, we formally define our *Fairness-Aware Decision tree Editing (FADE)* problem.

6.3.1 Dissimilarity Measure Between Decision Trees

To evaluate the dissimilarity between a given decision tree and its edited one in terms of its prediction and interpretation, we introduce two dissimilarity measures $\Gamma\colon \mathcal{H} \times \mathcal{H} \to \mathbb{R}_{\geq 0}$ for decision trees.

Prediction Discrepancy

One dissimilarity measure is the *prediction discrepancy (PD)* proposed by Marx et al. to evaluate predictive multiplicity in a classification task [229]. PD represents the ratio of individuals whose prediction results are changed if the initial classifier h_1 is replaced by another classifier h_2. In equation, PD between h_1 and h_2 is defined as follows:

$$\Gamma_{\mathrm{PD}}(h_1, h_2 \mid S) = \frac{1}{|S|} \sum_{(x,y,z) \in S} \mathbb{I}\left[h_1(x) \neq h_2(x)\right],$$

where S is a given sample. PD can be regarded to evaluate the functional dissimilarity between two classifiers on S.

Edit Distance

The second dissimilarity measure is the *edit distance (ED)* [322, 372], which is a standard dissimilarity measure between labeled ordered trees. The ED between two labeled ordered trees $\mathcal{T}_1, \mathcal{T}_2$ is defined as the minimal length of the sequence of edit operations required to transform $\mathcal{T}_1$ into $\mathcal{T}_2$. Three edit operations are available: *insertion*, *deletion*, and *relabeling*.

By regarding the branching features $d_i \in [D]$ and predictive labels $\hat{y}_l \in \mathcal{Y}$ as node labels, decision trees can be viewed as labeled ordered binary trees with node labels $\Sigma = [D] \cup \mathcal{Y}$. Let $\boldsymbol{e}(h_1, h_2) = \langle e_1, \ldots, e_M \rangle$ be a sequence of edit operations to transform

h_1 into h_2. Then, the ED between h_1 and h_2 is defined as follows:

$$\Gamma_{\mathrm{ED}}(h_1, h_2) = \min_{\boldsymbol{e}(h_1,h_2)} \left(\sum_{e \in \boldsymbol{e}(h_1,h_2)} \mathrm{cost}(e) \right),$$

where $\mathrm{cost}(e) \geq 0$ is the cost of the edit operation e.

Let ϵ be a special blank symbol not in Σ, and let $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$. Then, each edit operation e corresponding to nodes t_1 and t_2 can be expressed as a pair of their labels (s_1, s_2), where $s_1, s_2 \in \Sigma \cup \{\epsilon\}$ are the node labels of t_1 and t_2, respectively [190]. We can express e by (i) (ϵ, s_2) for inserting a node with a label s_2, (ii) (s_1, ϵ) for deleting a node with a label s_1, and (iii) (s_1, s_2) for relabeling a node with a label s_1 by another label s_2. We define the cost of e by $\mathrm{cost}(e) = \mathrm{dist}(t_1, t_2)$, where $\mathrm{dist}(t_1, t_2)$ is some dissimilarity measure between the labels of nodes t_1 and t_2. In this chapter, for simplicity, we assume that the unit cost $\mathrm{dist}(t_1, t_2) = \mathbb{I}[s_1 \neq s_2]$.

6.3.2 Edit Operation for Decision Tree

As edit operations for a given initial decision tree $h \in \mathcal{H}$, we consider deletion and relabeling. For h, let $\mathcal{H}^*(h) \subseteq \mathcal{H}$ be the set of edited decision trees h^* from h that satisfy the following five conditions:

C1. There exists an edit sequence $\boldsymbol{e}(h, h^*)$ consisting of deletion and relabeling operations to transform h into h^*.

C2. All internal nodes i and leaves l of h^* have a branching feature $d_i \in [D]$ and predictive label $\hat{y}_l \in \mathcal{Y}$ as their node labels, respectively.

C3. All internal nodes i of h^* have two child nodes each: $\mathrm{left}(i)$ and $\mathrm{right}(i)$.

C4. For any leaf l of h^*, there are no duplicate branching features for the internal nodes in A_l.

C5. For any internal node i of h^*, not all leaves included in its subtree have the same predictive label.

C2 and **C3** serve to maintain the model structure of an edited decision tree h^*, while **C4** and **C5** prevent h^* from including redundant nodes.

6.3.3 Problem Definition

Our goal is to reduce the discrimination score Δ_z of a given decision tree h by editing its branching features and predictive labels without significantly changing it. Using Γ and $\mathcal{H}^*(h)$, this problem can be defined as follows.

Problem 3 (FADE problem)**.** Given a sample S, a decision tree h, a dissimilarity measure Γ, and a discrimination threshold $\theta \in [0, 1]$, find a decision tree h^* that is an optimal solution of the following optimization problem:

$$\underset{h^* \in \mathcal{H}^*(h)}{\text{minimize}} \quad \Gamma(h, h^*) \quad \text{subject to} \quad \Delta_z(h^* \mid S) \leq \theta.$$

By solving the above optimization problem, we obtain an edited decision tree h^* that satisfies the fairness constraint $\Delta_z(h^* \mid S) \leq \theta$ with minimum change with respect to the dissimilarity measure $\Gamma(h, h^*)$.

6.4 MILO Formulation

In this section, we propose an MILO formulation of our FADE problem[1].

[1]For the details of techniques of MILO formulation in this chapter, see, e.g., [359].

6.4.1 Decision Tree Constraints

To express the edit operations for each node, i.e., deletion and relabeling (**C1**), and the edited decision tree $h^* \in \mathcal{H}^*(h)$, we first introduce some program variables. We introduce variables $\zeta_t \in \{0,1\}$ for $t \in \mathcal{T}$, which indicate whether a node t is deleted ($\zeta_t = 0$) or not ($\zeta_t = 1$). For an internal node $i \in \mathcal{I}$ and leaf $l \in \mathcal{L}$, let $d_i^* \in [D]$ and $\hat{y}_l^* \in \mathcal{Y}$ be their node labels in h^*, respectively. For $i \in \mathcal{I}$, we introduce variables $\alpha_{i,d} \in \{0,1\}$ for $d \in [D]$, which indicate whether $d_i^* = d$. Similarly, for $l \in \mathcal{L}$, we introduce a variable $\lambda_l^+ \in \{0,1\}$ (resp. $\lambda_l^- \in \{0,1\}$), which indicates whether $\hat{y}_l^* = 1$ (resp. $\hat{y}_l^* = 0$). We also introduce a variable $\lambda_i^+ \in \{0,1\}$ (resp. $\lambda_i^- \in \{0,1\}$) for $i \in \mathcal{I}$, which indicates whether all leaves included in the subtree of i have the predictive label 1 (resp. 0). To satisfy the conditions mentioned in Section 6.3.2, we impose the following constraints on these variables:

$$\sum_{d=1}^{D} \alpha_{i,d} = \zeta_i, \qquad \forall i \in \mathcal{I}, \tag{6.1}$$

$$\lambda_l^+ + \lambda_l^- = \zeta_l, \qquad \forall l \in \mathcal{L}, \tag{6.2}$$

$$\zeta_i \leq \zeta_{j_i}, \qquad \forall i \in \mathcal{I}, j_i = \max(A_i^{(\mathrm{R})} \cup \{0\}), \tag{6.3}$$

$$\zeta_l = \zeta_{i_l}, \qquad \forall l \in \mathcal{L}, i_l = \max(A_l^{(\mathrm{R})} \cup \{0\}), \tag{6.4}$$

$$\sum_{i \in A_l} \alpha_{i,d} \leq 1, \qquad \forall l \in \mathcal{L}, d \in [D], \tag{6.5}$$

$$0 \leq \zeta_{\mathrm{right}(i)} + 2 \cdot \lambda_i^* - \lambda_{\mathrm{left}(i)}^* - \lambda_{\mathrm{right}(i)}^* \leq 1, \qquad \forall i \in \mathcal{I}, * \in \{+,-\}, \tag{6.6}$$

$$\zeta_i \leq 1 - (\lambda_i^+ + \lambda_i^-), \qquad \forall i \in \mathcal{I}, \tag{6.7}$$

$$\zeta_0 = 1, \tag{6.8}$$

where $\mathrm{left}(i) \in \mathcal{T}$ (resp. $\mathrm{right}(i) \in \mathcal{T}$) is a left (resp. right) child node of i, and ζ_0 is a dummy variable for notational convenience. Constraints (6.1–6.4) represent **C2** and

C3, while constraints (6.5–6.7) represent **C4** and **C5**. Note that $j_i, \mathrm{left}(i), \mathrm{right}(i)$ for $i \in \mathcal{I}$ and i_l for $l \in \mathcal{L}$ can be computed when h is given.

6.4.2 Fairness Constraints

To express the fairness constraint $\Delta_z(h^* \mid S) \leq \theta$, we must first express the output $h^*(x^{(n)})$ of an edited decision tree h^* for each input instance $x^{(n)}$ in a given sample S. We introduce indicator variables $\phi_{l,n} \in \{0,1\}$ for $l \in \mathcal{L}$ and $n \in [N]$ such that $\phi_{l,n} = \phi_l^*(x^{(n)})$, where $\phi_l^* \colon \mathcal{X} \to \{0,1\}$ is the leaf indicator function of a leaf l in h^*. We also introduce auxiliary variables $\psi_{i,n} \in \{0,1\}$ indicating whether $x^{(n)}$ satisfies the branching rule of $i \in \mathcal{I}$, i.e., $\psi_{i,n} = \mathbb{I}\left[x_{d_i^*} = 1\right]$. From the definition of the leaf indicator function in Section 6.2.1, we can express $\phi_{l,n}$ as

$$\phi_{l,n} = \prod\nolimits_{i \in A_l^{(\mathrm{L})}} (1 - \psi_{i,n}) \cdot \prod\nolimits_{i \in A_l^{(\mathrm{R})}} \psi_{i,n}.$$

These non-linear constraints can be linearized as follows:

$$\psi_{i,n} = \sum_{d=1}^{D} \alpha_{i,d} \cdot x_d^{(n)}, \qquad \forall i \in \mathcal{I}, n \in [N], \tag{6.9}$$

$$I_l \cdot \phi_{l,n} \leq \sum_{i \in A_l^{(\mathrm{L})}} (1 - \psi_{i,n}) + \sum_{i \in A_l^{(\mathrm{R})}} \psi_{i,n}, \qquad \forall l \in \mathcal{L}, n \in [N], \tag{6.10}$$

$$\sum_{i \in A_l^{(\mathrm{L})}} (1 - \psi_{i,n}) + \sum_{i \in A_l^{(\mathrm{R})}} \psi_{i,n} \leq I_l \cdot \phi_{l,n} + I_l - 1, \qquad \forall l \in \mathcal{L}, n \in [N], \tag{6.11}$$

where I_l is a constant value such that $I_l = |A_l|$, which can be computed when h is given. Constraint (6.9) represents $\psi_{i,n} = \mathbb{I}\left[x_{d_i^*} = 1\right]$, while Constraints (6.10) and (6.11) represent $\phi_{l,n} = \prod_{i \in A_l^{(\mathrm{L})}} (1 - \psi_{i,n}) \cdot \prod_{i \in A_l^{(\mathrm{R})}} \psi_{i,n}$. For maintaining the model structure of h^* with respect to S, the variables $\phi_{l,n}$ must also satisfy the following

constraints:

$$\zeta_l \leq \sum_{n \in [N]} \phi_{l,n}, \qquad \forall l \in \mathcal{L}, \tag{6.12}$$

$$\phi_{l,n} \leq \zeta_l, \qquad \forall l \in \mathcal{L}, n \in [N], \tag{6.13}$$

$$\sum_{l \in \mathcal{L}} \phi_{l,n} = 1, \qquad \forall n \in [N], \tag{6.14}$$

Constraints (6.12) and (6.13) are the constraints for preventing h^* from having redundant leaves that no input instance in S reaches. Constraint (6.14) imposes that each input instance is assigned to exactly one leaf of h^*.

Here, we express the fairness constraint using $\phi_{l,n}$, and show that $\Delta_z(h^* \mid S)$ can be expressed as the summation of the values of the DP score in each leaf of h^*.

Proposition 2. *For $\bar{z} \in \{0, 1\}$ and S, let $N_{\bar{z}} = \{n \in [N] \mid z^{(n)} = \bar{z}\}$. Then, we have*

$$\Delta_z(h^* \mid S) = \left| \sum_{l \in \mathcal{L}} \lambda_l^+ \cdot \sum_{n=1}^{N} c_n \cdot \phi_{l,n} \right|,$$

where $c_n = \frac{z^{(n)}}{|N_1|} - \frac{1 - z^{(n)}}{|N_0|}$.

Proof. From the definitions of DP Δ_z and decision tree h^*, we have

$$\begin{aligned}
\Delta_z(h^* \mid S) &= \left| \frac{\sum_{n \in N_1} h^*(x^{(n)})}{|N_1|} - \frac{\sum_{n \in N_0} h^*(x^{(n)})}{|N_0|} \right| \\
&= \left| \sum_{l \in \mathcal{L}} \mathbb{I}\left[\hat{y}_l^* = 1\right] \cdot \sum_{n=1}^{N} c_n \cdot \phi_l^*(x^{(n)}) \right| \\
&= \left| \sum_{l \in \mathcal{L}} \lambda_l^+ \cdot \sum_{n=1}^{N} c_n \cdot \phi_{l,n} \right|.
\end{aligned}$$

□

We introduce variables $\delta_l \in [-1, 1]$ for $l \in \mathcal{L}$ such that $\delta_l = \lambda_l^+ \cdot \sum_{n=1}^N c_n \cdot \phi_{l,n}$, which contain non-linear constraints on the variables δ_l, λ_l^+, and $\phi_{l,n}$. From Constraint (6.2) and $-1 \leq \sum_{n=1}^N c_n \cdot \phi_{l,n} \leq 1$, we can linearize them by the following linear constraints:

$$-\lambda_l^+ \leq \delta_l \leq \lambda_l^+, \qquad \forall l \in \mathcal{L}, \tag{6.15}$$

$$\sum_{n=1}^N c_n \cdot \phi_{l,n} - \lambda_l^- \leq \delta_l \leq \sum_{n=1}^N c_n \cdot \phi_{l,n} + \lambda_l^-, \qquad \forall l \in \mathcal{L}. \tag{6.16}$$

Note that c_n is a constant value because it can be computed when S is given. From Proposition 2, the constraint $\Delta_z(h^* \mid S) \leq \theta$ is equivalent to $\left|\sum_{l \in \mathcal{L}} \delta_l\right| \leq \theta$. It can be expressed as the following linear constraint of δ_l:

$$-\theta \leq \sum_{l \in \mathcal{L}} \delta_l \leq \theta. \tag{6.17}$$

6.4.3 Objective Function

We introduce a variable $\gamma \geq 0$ to express $\gamma = \Gamma(h, h^*)$, which is the value of the objective function of Problem 3 to be minimized. In the following, we express PD and ED by linear constraints on program variables.

Prediction Discrepancy

As with $\Delta_z(h^* \mid S)$, we show that PD $\Gamma_{\mathrm{PD}}(h, h^* \mid S)$ can be expressed as the summation of the PD values in each leaf of h^* as follows:

$$\begin{aligned}
\Gamma_{\mathrm{PD}}(h, h^* \mid S) &= \frac{1}{N} \sum_{n=1}^N \sum_{l \in \mathcal{L}} \mathbb{I}\left[h(x^{(n)}) \neq \hat{y}_l^*\right] \cdot \phi_l(x^{(n)}) \\
&= \frac{1}{N} \sum_{l \in \mathcal{L}} \left((1 - \hat{y}_l^*) \cdot \sum_{n=1}^N h(x^{(n)}) \cdot \phi_l(x^{(n)}) + \hat{y}_l^* \cdot \sum_{n=1}^N (1 - h(x^{(n)})) \cdot \phi_l(x^{(n)}) \right) \\
&= \frac{1}{N} \sum_{l \in \mathcal{L}} (\lambda_l^- \cdot N_l^+ + \lambda_l^+ \cdot N_l^-),
\end{aligned}$$

where $N_l^+ = \sum_{n=1}^N h(x^{(n)}) \cdot \phi_l(x^{(n)})$ and $N_l^- = \sum_{n=1}^N (1 - h(x^{(n)})) \cdot \phi_l(x^{(n)})$ for $l \in \mathcal{L}$.

Based on the above result, we introduce variables $\gamma_l \geq 0$ for $l \in \mathcal{L}$ such that $\gamma_l = N_l^+ \cdot \lambda_l^- + N_l^- \cdot \lambda_l^+$, which is the value of PD in l. Since the values of N_l^+ and N_l^- depend on the value of $\phi_l(x^{(n)})$, they include non-linear terms of the variables $\phi_{l,n}$, λ_l^+, and λ_l^-. We can linearize these non-linear constraints as follows:

$$N_l^- - N \cdot \lambda_l^- \leq \gamma_l \leq N_l^- + N \cdot \lambda_l^-, \qquad \forall l \in \mathcal{L}, \tag{6.18}$$

$$N_l^+ - N \cdot \lambda_l^+ \leq \gamma_l \leq N_l^+ + N \cdot \lambda_l^+, \qquad \forall l \in \mathcal{L}, \tag{6.19}$$

$$N_l^+ = \sum_{n=1}^N h(x^{(n)}) \cdot \phi_{l,n}, \qquad \forall l \in \mathcal{L}, \tag{6.20}$$

$$N_l^- = \sum_{n=1}^N (1 - h(x^{(n)})) \cdot \phi_{l,n}, \qquad \forall l \in \mathcal{L}. \tag{6.21}$$

Note that $h(x^{(n)})$ is a constant value because they can be computed when h and S are given. Finally, we can express $\gamma = \Gamma_{\mathrm{PD}}(h, h^* \mid S)$ by the following constraints:

$$\gamma = \frac{1}{N} \sum_{l \in \mathcal{L}} \gamma_l. \tag{6.22}$$

Edit Distance

To formulate the ED $\Gamma_{\mathrm{ED}}(h, h^*)$, we extend the formulation proposed by Kondo et al. to address our FADE problem [190]. They proposed an integer linear optimization formulation for computing ED between given two trees by utilizing *Tai mappings* [322]. A Tai mapping M between two labeled ordered trees $\mathcal{T}_1$ and $\mathcal{T}_2$ is a subset of $\mathcal{T}_1 \times \mathcal{T}_2$ satisfying the following three conditions for any $(t_1, t_2), (u_1, u_2) \in M$:

C6. (One-to-one mapping) $t_1 = u_1 \iff t_2 = u_2$,

C7. (Preserving ancestor) $t_1 <_{\mathrm{anc}} u_1 \iff t_2 <_{\mathrm{anc}} u_2$,

C8. (Preserving sibling) $t_1 <_{\text{sib}} u_1 \iff t_2 <_{\text{sib}} u_2$,

where $t <_{\text{anc}} u$ (resp. $t <_{\text{sib}} u$) indicates that a node t is an ancestor (resp. to the left) of a node u. For a Tai mapping M, the cost of M is defined as

$$\text{cost}(M) = \sum_{(t_1,t_2)\in M} \text{dist}(t_1, t_2) + \sum_{t_1 \in M_{\text{del}}} \text{dist}(t_1, \epsilon) + \sum_{t_2 \in M_{\text{ins}}} \text{dist}(\epsilon, t_2),$$

where $M_{\text{del}} = \{t_1 \in \mathcal{T}_1 \mid \forall t_2 \in \mathcal{T}_2 : (t_1, t_2) \notin M\}$ and $M_{\text{ins}} = \{t_2 \in \mathcal{T}_2 \mid \forall t_1 \in \mathcal{T}_1 : (t_1, t_2) \notin M\}$, which are the sets of deleted and inserted nodes, respectively. Tai proved that the ED between $\mathcal{T}_1$ and $\mathcal{T}_2$ is equivalent to the minimum Tai mapping cost, which indicates that the ED can be computed by finding a minimum-cost Tai mapping instead of finding a minimum-cost sequence of edit operations transforming $\mathcal{T}_1$ into $\mathcal{T}_2$ [322]. Based on this result, we formulate $\Gamma_{\text{ED}}(h, h^*)$ by expressing $\text{cost}(M)$ by program variables and minimizing it.

To express a Tai mapping M between h and h^*, we first introduce variables $\mu_{t_1,t_2} \in \{0, 1\}$ for $t_1, t_2 \in \mathcal{T}$ that indicate whether $(t_1, t_2) \in M$. Since M is a Tai mapping, μ_{t_1,t_2} must satisfy the following constraints [190]:

$$\mu_{t_1,t_2} \in \{0, 1\}, \qquad \forall t_1, t_2 \in \mathcal{T}, \tag{6.23}$$

$$\sum_{t_2\in\mathcal{T}} \mu_{t_1,t_2} \leq 1, \qquad \forall t_1 \in \mathcal{T}, \tag{6.24}$$

$$\sum_{t_1\in\mathcal{T}} \mu_{t_1,t_2} = \zeta_{t_2} \qquad \forall t_2 \in \mathcal{T}, \tag{6.25}$$

$$\mu_{t_1,t_2} + \mu_{u_1,u_2} \leq 1, \qquad \forall((t_1, t_2), (u_1, u_2)) \in \mathcal{T}_{<_{\text{anc}}}, \tag{6.26}$$

$$\mu_{t_1,t_2} + \mu_{u_1,u_2} \leq 1, \qquad \forall((t_1, t_2), (u_1, u_2)) \in \mathcal{T}_{<_{\text{sib}}}, \tag{6.27}$$

where $\mathcal{T}_{\prec} = \{((t_1, t_2), (u_1, u_2)) \in \mathcal{T}^2 \times \mathcal{T}^2 \mid t_1 \prec u_1 \iff t_2 \prec u_2\}$ for a binary relation $\prec \in \{<_{\text{anc}}, <_{\text{sib}}\}$ on $\mathcal{T}$, which can be computed when h is given. Constraints (6.24)

and (6.25) represents **C6** (one-to-one mapping), while Constraint (6.26) and Constraint (6.27) represents **C7** (preserving ancestor) and **C8** (preserving sibling), respectively.

Next, we show that $\mathrm{cost}(M)$ can be expressed as the summation of the costs in each node. In our Problem 3, since we consider deletion and relabeling as our available edit operation (**C1**), $M_{\mathrm{ins}} = \emptyset$ and $\sum_{t_2 \in M_{\mathrm{ins}}} \mathrm{dist}(\epsilon, t_2) = 0$ hold. From the definitions of M_{del} and μ_{t_1,t_2}, we obtain

$$\begin{aligned}
\mathrm{cost}(M) &= \sum_{(t_1,t_2)\in M} \mathrm{dist}(t_1,t_2) + \sum_{t_1 \in M_{\mathrm{del}}} \mathrm{dist}(t_1,\epsilon) \\
&= \sum_{t_1\in\mathcal{T}} \sum_{t_2\in\mathcal{T}} \mathrm{dist}(t_1,t_2)\cdot\mu_{t_1,t_2} + \sum_{t_1\in\mathcal{T}} \mathrm{dist}(t_1,\epsilon)\cdot\left(1 - \sum_{t_2\in\mathcal{T}} \mu_{t_1,t_2}\right) \\
&= \sum_{t_1\in\mathcal{T}} \left(1 + \sum_{t_2\in\mathcal{T}} (\mathrm{dist}(t_1,t_2) - 1)\cdot\mu_{t_1,t_2}\right).
\end{aligned}$$

Note that $\mathrm{dist}(t_1, \epsilon) = 1$ from the definition of dist.

We introduce variables $\gamma_{t_1} \geq 0$ for $t_1 \in \mathcal{T}$ such that $\gamma_{t_1} = 1 + \sum_{t_2\in\mathcal{T}}(\mathrm{dist}(t_1,t_2) - 1)\cdot\mu_{t_1,t_2}$. They include non-linear constraints because the value of $\mathrm{dist}(t_1,t_2)$ depends on the node labels of t_2, i.e., the variables $\alpha_{t_2,d}$ if $t_2 \in \mathcal{I}$, and $\lambda^+_{t_2}$ and $\lambda^-_{t_2}$ if $t_2 \in \mathcal{L}$. From the definition of μ_{t_1,t_2} and Constraint (6.25), we can express γ_{t_1} as follows:

$$\gamma_{t_1} = \begin{cases} 1 & \text{if } \sum_{t_2\in\mathcal{T}} \mu_{t_1,t_2} = 0, \\ \mathrm{dist}(t_1,t^*) & \text{otherwise,} \end{cases}$$

where $t^* \in \mathcal{T}$ such that satisfies $\mu_{t_1,t^*} = 1$. From these observations, we can linearize the above constraints on γ_{t_1} by the following constraints:

$$\gamma_{t_1} \geq 1 - \sum_{t_2\in\mathcal{T}} \mu_{t_1,t_2}, \qquad \forall t_1 \in \mathcal{T} \qquad (6.28)$$

$$\gamma_{t_1} \geq \mathrm{dist}(t_1, t_2) - U \cdot (1 - \mu_{t_1,t_2}), \qquad \forall t_1, t_2 \in \mathcal{T} \tag{6.29}$$

$$\mathrm{dist}(t_1, t_2) = \begin{cases} \hat{y}_{t_1} \cdot \lambda^-_{t_2} + (1 - \hat{y}_{t_1}) \cdot \lambda^+_{t_2} & \text{if } t_1, t_2 \in \mathcal{L}, \\ \sum_{d \in [D]} \mathbb{I}\left[d_t \neq d\right] \cdot \alpha_{t_2,d} & \text{if } t_1, t_2 \in \mathcal{I}, \\ 1 & \text{otherwise,} \end{cases} \qquad \forall t_1, t_2 \in \mathcal{T}, \tag{6.30}$$

where $U \geq 0$ is the upper bound of $\mathrm{dist}(t_1, t_2)$, and we can set $U = 1$ from the definition of dist. Finally, we can express $\gamma = \Gamma_{\mathrm{ED}}(h, h^*)$ by the following constraints:

$$\gamma = \sum_{t \in \mathcal{T}} \gamma_t. \tag{6.31}$$

6.4.4 Overall Formulation

Finally, our overall MILO formulation can be expressed as follows:

$$\begin{aligned} \text{minimize} \quad & \gamma \\ \text{subject to} \quad & \text{Constraint (6.1–6.17)}, \\ & \begin{cases} \text{Constraint (6.18–6.22)}, & \text{if } \Gamma = \Gamma_{\mathrm{PD}}, \\ \text{Constraint (6.23–6.31)}, & \text{if } \Gamma = \Gamma_{\mathrm{ED}}, \end{cases} \\ & \alpha_{i,d} \in \{0, 1\}, && \forall i \in \mathcal{I}, d \in [D], \\ & \zeta_t \in \{0, 1\}, && \forall t \in \mathcal{T}, \\ & \lambda^+_t \in \{0, 1\}, && \forall t \in \mathcal{T}, \\ & \lambda^-_t \in \{0, 1\}, && \forall t \in \mathcal{T}, \\ & \gamma_t \geq 0, && \forall t \in \mathcal{T}, \\ & \phi_{l,n} \in \{0, 1\}, && \forall l \in \mathcal{L}, n \in [N], \\ & \delta_l \in [-1, 1], && \forall l \in \mathcal{L}, \\ & \gamma \geq 0. \end{aligned}$$

The total numbers of variables and constraints excluding those for Γ are both $\mathcal{O}(|\mathcal{L}| \cdot (N + D))$. For $\Gamma = \Gamma_{\mathrm{PD}}$ (resp. $\Gamma = \Gamma_{\mathrm{ED}}$), $\mathcal{O}(|\mathcal{L}|)$ variables and constraints (resp. $\mathcal{O}(|\mathcal{T}|^2)$ variables and $\mathcal{O}(|\mathcal{T}|^4)$ constraints) are additionally required.

As with existing MILO-based methods [8,38,341], our formulation can be efficiently solved by off-the-shelf MILO solvers, such as CPLEX [153], and customized by adding constraints that users desire. Consequently, we can obtain an optimal edited decision tree under not only fairness constraints but also user-defined additional constraints without implementing designated algorithms.

Additional Constraints. Our FADE problem includes no constraints on the prediction error of an edited decision tree h^*. To improve the practicality of the obtained decision tree, we can add constraints corresponding to the empirical risk to our formulation. We denote the empirical risk of a classifier h on a sample S by $\hat{R}(h \mid S) = \frac{1}{N} \sum_{(x,y,z) \in S} \mathbb{I}[y \neq h(x)]$. For a parameter $\varepsilon \geq 0$, we consider the constraint $\hat{R}(h^* \mid S) \leq (1 + \varepsilon) \cdot \hat{R}(h \mid S)$, which states that the increase in the empirical risk of an edited decision tree h^* must be within 100ε % of the initial decision tree h. This constraint can be expressed as follows:

$$
\begin{aligned}
&\rho_l \geq 0, && \forall l \in \mathcal{L} \\
&R^*_l - N \cdot \lambda^*_l \leq \rho_l \leq R^*_l + N \cdot \lambda^*_l, && \forall l \in \mathcal{L}, * \in \{+, -\}, \\
&R^+_l = \sum_{n=1}^{N} y^{(n)} \cdot \phi_{l,n}, && \forall l \in \mathcal{L}, \\
&R^-_l = \sum_{n=1}^{N} (1 - y^{(n)}) \cdot \phi_{l,n}, && \forall l \in \mathcal{L}, \\
&\frac{1}{N} \sum_{l \in \mathcal{L}} \rho_l \leq (1 + \varepsilon) \cdot R(h \mid S).
\end{aligned}
$$

These constraints are derived by replacing $h(x^{(n)})$ and γ_l in Constraints (6.18–6.21), which are the constraints for expressing PD Γ_{PD}, with $y^{(n)}$ and ρ_l, respectively. In our experiments, we added the above constraints into our overall formulation.

6.5 Experiments

In this section, we conducted experiments on real datasets to evaluate our proposed FADE method. All code was implemented in Python 3.7 with scikit-learn and IBM ILOG CPLEX v12.10, and all experiments were conducted on 64-bit macOS Catalina 10.15.6 with Intel Core i9 2.4 GHz CPU and 64 GB memory. In addition, a 5-hour time limit was imposed for obtaining the solution.

6.5.1 Experimental Setup

We consider out-of-sample deployment settings [335], where a classifier is trained on a biased dataset and is deployed on individuals who are not included in the training dataset. We investigate the behavior and effectiveness of FADE in such scenarios. To reproduce this setting and apply FADE to it, we performed the following procedures:

1. Divide a training sample S into S_0 and S_1, where $S_{\bar{z}} = \{(x, y, z) \in S \mid z = \bar{z}\}$ for $\bar{z} \in \{0, 1\}$.

2. Train a decision tree h on the biased sample S_0

3. Edit h on the entire training sample S by FADE.

In our preliminary experiments, we confirmed that procedures (1) and (2) often increased the DP values of the initial decision trees h. We trained the initial decision trees

h by CART [49] with a maximum depth of 4. We denote the method that optimizes Γ_{PD} (resp. Γ_{ED}) as the dissimilarity measure Γ by **FADE-PD** (resp. **FADE-ED**).

Datasets. We used the German ($N = 1000, D = 31$) [88] and COMPAS ($N = 6167, D = 17$) [24] datasets, which are standard benchmarking datasets in the context of fairness in machine learning [127]. The task of the German dataset is to predict whether individuals will default on their loan, while the task of the COMPAS dataset is to predict whether individuals will reoffend within 2 years. The sensitive attributes z represent whether the individuals are young adults (Age < 25) for the German dataset, and whether they are African-American for the COMPAS dataset.

Baseline Method. To evaluate the performance of the decision trees edited by our method, we compared them with decision trees trained by the *discrimination-aware decision tree (DADT)* algorithm [167] on the entire training sample S as the **Baseline** method. DADT is an in-processing method for learning fair decision trees. In its training process, branching features are selected based on the specific criterion that considers not only class labels y but also sensitive attributes z. In addition, the predictive labels are optimized under the constraint on the DP value of the trained decision tree for a given threshold θ.

6.5.2 Experimental Results

Sensitivity Analysis of Discrimination Threshold

To evaluate the behavior of our proposed framework, we show the sensitivity of the threshold θ for the discrimination score $\Delta_z(h^* \mid S)$ in our optimization problem. Figure 6.3 presents $\Gamma(h, h^*)$ between an initial decision tree h and its edited one h^* for

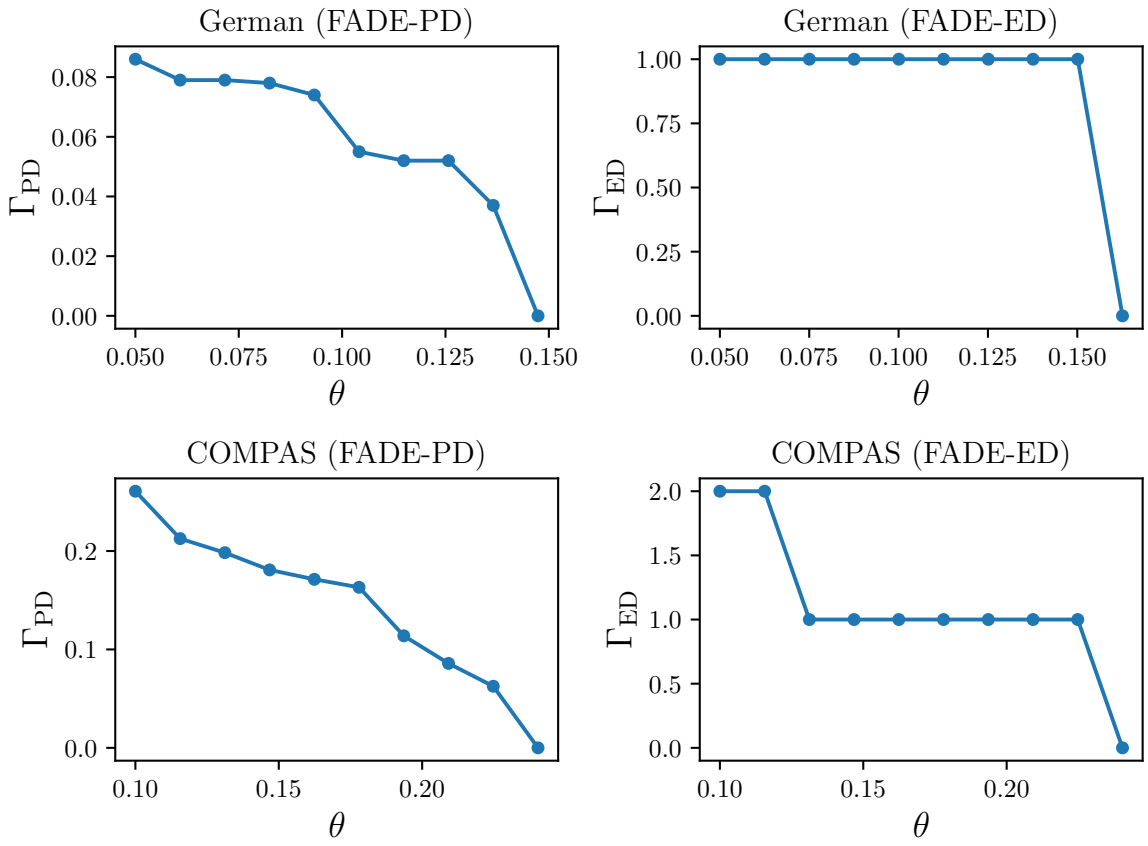


Figure 6.3: Sensitivity analysis of discrimination threshold θ in our optimization problem.

each θ. It can be seen that both PD and ED between h and h^* monotonically increased as θ decreased. This indicates that as the constraint corresponding to a discrimination score Δ_z became stronger, it became more difficult to edit a biased decision tree h into a fair decision tree h^* with fewer changes with respect to the dissimilarity $\Gamma(h, h^*)$.

Performance Comparison with Baseline

Next, we compared the performance of decision trees edited by FADE with Baseline. We set $\theta = 0.01, \varepsilon = 0.2$ for the German dataset and $\theta = 0.1, \varepsilon = 0.3$ for the COMPAS dataset. We measured the average empirical risk $\hat{R}$ and DP Δ_z. We also report the average values of PD Γ_{PD} and ED Γ_{ED} between the initial biased decision tree trained by CART and decision trees obtained by each method.

Table 6.1 presents the results of 10-fold cross validation. Although our FADE edited biased decision trees, their empirical risk and DP were only approximately 3% lower than those obtained by Baseline. In addition, we can also see that FADE achieved lower PD and ED than those obtained by Baseline. For example, in the German dataset, the test risk, test DP, PD, and ED given by FADE-ED were 0.298, 0.066, 0.172, and 1.0, respectively, which were comparable to or less than those obtained by Baseline. These results indicate that our proposed method FADE can convert biased decision trees into fair decision trees without significantly changing them.

Computational Complexity. Whereas FADE-ED often succeeded in finding an optimal solution within the predefined time limit, FADE-PD did not for either the German or COMPAS datasets. Actually, the average running time of FADE-ED was 187 seconds on the German dataset and 4786 seconds on the COMPAS dataset, while that of FADE-PD was 18000 seconds on both the German and COMPAS datasets since

Table 6.1: Results of 10-fold cross validation.

(a) German dataset

Method	Empirical Risk		DP		PD	ED
	Train	Test	Train	Test		
Baseline	0.269 ± 0.0082	0.295 ± 0.021	0.00599 ± 0.0041	0.0816 ± 0.049	0.251 ± 0.035	16.2 ± 2.1
FADE-PD	0.289 ± 0.012	0.318 ± 0.047	0.00449 ± 0.0031	0.0951 ± 0.09	0.0907 ± 0.018	4.9 ± 2.6
FADE-ED	0.288 ± 0.014	0.298 ± 0.029	0.00371 ± 0.0025	0.066 ± 0.05	0.172 ± 0.051	1.0 ± 0.0

(b) COMPAS dataset

Method	Empirical Risk		DP		PD	ED
	Train	Test	Train	Test		
Baseline	0.395 ± 0.014	0.399 ± 0.027	0.0679 ± 0.026	0.0786 ± 0.03	0.325 ± 0.041	15.1 ± 2.2
FADE-PD	0.402 ± 0.0071	0.401 ± 0.022	0.092 ± 0.0081	0.0953 ± 0.03	0.204 ± 0.045	8.2 ± 3.4
FADE-ED	0.414 ± 0.0088	0.413 ± 0.014	0.0757 ± 0.022	0.0718 ± 0.032	0.268 ± 0.038	1.7 ± 0.67

FADE-PD never succeeded in finding optimal solutions within 5-hours. These results seem counterintuitive because the total number of variables and constraints included in FADE-PD is less than that in FADE-ED as shown in Section 6.4.4. These results indicate that minimizing PD Γ_{PD} under the fairness constraint with respect to DP Δ_z is more computationally difficult than ED Γ_{ED}.

Edited Rules. Finally, we present examples of branching features edited by FADE in Table 6.2, which displays pairs of replaced branching rules to eliminate discrimination from the initial decision trees. These results can help decision-makers to identify the causes of unfairness in their deployed models, and suggest ways to eliminate them [4, 264].

6.6 Conclusion

In this study, we proposed a fairness-aware post-processing method for converting a trained decision tree into a fair decision tree without significantly changing it in terms of its prediction and interpretation. For this purpose, we introduced two dissimilarity measures between decision trees based on the prediction discrepancy and edit distance, and proposed a mixed-integer linear optimization approach for optimizing them under fairness constraints. Through experiments on real datasets, we confirmed the effectiveness of our method by comparing it with an existing in-processing method for learning fair decision trees.

In future work, we plan to develop a more designed dissimilarity measure between decision trees by considering the depth of nodes in the decision trees and the dissimilarity between the node labels. In addition, it would be interesting to extend our method to more general tasks of editing machine learning models [236, 294, 309]. For

Table 6.2: Examples of edited branching features by FADE.

(a) German dataset

Initial Rule	$\rightarrow$	**Edited Rule**
Job Skill ≤ 2	$\rightarrow$	Duration ≤ 10
Duration ≤ 10	$\rightarrow$	Credit Amount ≤ 3368
Duration ≤ 10	$\rightarrow$	Checking Account ≤ 2
Checking Account ≤ 1	$\rightarrow$	Housing = Rent

(b) COMPAS dataset

Initial Rule	$\rightarrow$	**Edited Rule**
Priors ≤ 1	$\rightarrow$	Charge Degree = Felonies
Priors ≤ 3	$\rightarrow$	Juvenile-Misdemeanors ≤ 0
Age ≤ 31	$\rightarrow$	Juvenile-Felonies ≤ 0
Juvenile-Felonies ≤ 0	$\rightarrow$	Age ≤ 27

example, recently, Sinitsin et al. proposed a model editing framework of DNNs; that is, modifying the parameters of a trained DNN so as to improve its prediction performance for a given test sample, while maintaining that for the training sample [309]. It is interesting for future work to incorporate the hearts of our framework, i.e., minimizing the dissimilarities in prediction and interpretation under fairness constraints, into their framework for DNNs. Extending our framework to other settings of machine learning, such as online learning [374] or transfer learning [323], is also an interesting future direction.

Chapter 7

Conclusions

7.1 Summary

The aim of this study was to develop new practical frameworks of explainable machine learning for algorithmic decision-making. In this thesis, we focused on Counterfactual Explanation (CE) and decision trees, which are popular approaches for explainable machine learning, and developed three methods for improving their practicality based on mixed-integer linear optimization (MILO). We summarize each technical contribution of this thesis as follows:

- In Chapter 4, we proposed a post-hoc local explanation method, named Distribution-Aware Counterfactual Explanation (DACE), that provides realistic actions for users. To suggest a realistic action, it is desirable to evaluate actions by taking into account the characteristics corresponding to the underlying data distribution. For that purpose, we introduced a new objective function of CE that evaluates the reality of actions based on feature-correlations and outlier risks by the Mahalanobis distance and local outlier factor. Then, we proposed an efficient MILO

formulation to approximately optimize the cost function. Experimental results demonstrated the effectiveness of our method compared to existing CE methods.

- In Chapter 5, we proposed a new framework of CE, named Ordered Counterfactual Explanation (OrdCE), that provides not only an action but also its order of changing features. To make actions easier for users to execute, it is desirable to show not only a perturbation but also an appropriate order of changing the features based on asymmetric interactions among features, such as causal effects. For that purpose, we introduced an "ordered action," which is a pair of a perturbation and an order of changing the features, and designed a new objective function of CE that evaluates the required effort of ordered actions based on feature interactions estimated in advance. Then, we proposed an MILO formulation for extracting an optimal ordered action that minimizes our cost function. By experiments on real datasets, we confirmed the effectiveness of our method.

- In Chapter 6, we proposed a post-processing framework, named Fairness-Aware Decision tree Editing (FADE), that modifies a given unfair decision tree into a fair one without significantly changing its prediction and interpretation. When interpretable models are retrained during their deployment, it is desirable to modify only those parts of their prediction and interpretation that are essentially necessary to satisfy given constraints. For that purpose, we introduced two dissimilarity measures between decision trees based on the prediction discrepancy and edit distance as objective functions. Then, we proposed an MILO formulation for obtaining an optimal decision tree that minimizes the dissimilarity with a given decision tree under fairness constraints. Experimental results demonstrated the effectiveness of our method compared to existing learning methods.

Overall, we made three technical contributions to explainable machine learning in this thesis. Through this study, we (i) elucidated three concrete desiderata of CE and decision trees by exploring their practicality issues, (ii) introduced new optimization problems for satisfying the identified desiderata by mathematically modeling them as objective functions and constraints, and (iii) proposed efficient and flexible optimization methods for the formulated problems based on MILO. We believe that the results in this thesis contribute to the first step of developing new practical frameworks of explainable machine learning for algorithmic decision-making in the real world.

7.2 Towards the Future

For future work, we would like to improve our proposed methods from the perspectives of their efficiency, extensibility, and usability. Important future work related to this work is listed below.

- *Efficient Optimization*: By experiments on real datasets, we often observed the computational difficulties of our proposed methods based on MILO. In general, the scalability of MILO-based methods mainly depends on its size of formulation, i.e., the total number of variables and constraints. Therefore, we plan to devise more efficient MILO formulations in the sense of their total number of variables and constraints. It is also interesting future work to reduce our formulated tasks to other mathematical optimization problems rather than MILO problems. For example, SP-LIME [280] and Anchor [281], which are popular post-hoc local explanation methods, propose efficient algorithms for their optimization tasks of extracting explanations by incorporating the existing techniques of submodular optimization and multi-armed bandit, respectively. Therefore, we will consider

developing efficient algorithms by incorporating other mathematical optimization techniques into our optimization tasks.

- *Model Extension*: In this thesis, we identified a few desiderata that the explanation provided by each existing method should satisfy, and proposed objective functions or constraints based on each desideratum. To develop a practical framework for explainable machine learning, we need to extend our mathematical frameworks to support a wider class of objective functions and constraints based on other desiderata. For that purpose, it is essential to systematically identify not only (1) what kinds of actual desiderata are expressed as concrete objective functions or constraints, but also (2) how they can be optimized [85, 99, 217, 261]. Therefore, it is important for future work (1) to systematically classify the desiderata of explanations depending on their properties, and (2) to develop methods for efficiently formulating them as objective functions and constraints of MILO problems. Moreover, creating new practical forms of explanation that can provide human users with more meaningful insights by extending and combining the existing methods, i.e., post-hoc local explanation and interpretable models, is also interesting for future work[1].

- *User Study*: In practice, explanations provided by our proposed methods are finally evaluated by human users. Therefore, it is important for future work to conduct user experiments to investigate whether our methods can actually help human users. In fact, several papers on explainable machine learning published recently often include user experiments to evaluate the usability of their

[1]To tell the truth, we have already developed a new framework of CE that globally summarizes actions assigned to instances over the entire input space with decision trees, which will be published in [173].

proposed explanation methods (e.g., [66, 108, 198, 279]). The number of papers comparing the effectiveness of existing popular methods by conducting massive user experiments is also increasing for the last few years [17, 137, 161, 163, 269]. Following these previous studies, we plan to conduct user experiments to qualitatively evaluate the actual usability of explanations provided by our methods, and to improve our methods based on the results. Furthermore, it is important to clarify new desiderata of explanations through user experiments, which is crucial for future research on explainable machine learning. In this study, we identified a few desiderata based on the issues discussed by previous studies or our empirical observations on benchmark datasets. In practice, however, the desiderata of explanations for human users often depend on each situation, e.g., type of the decision-making task or preference of the human users [85,217,356]. Therefore, an important future direction of this study is to identify hidden desiderata through user experiments, and to attempt to mathematically model them.

Bibliography

[1] Amina Adadi and Mohammed Berrada. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018.

[2] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 9525–9536, 2018.

[3] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 700–712, 2020.

[4] Philip Adler, Casey Falk, Sorelle A. Friedler, Tionney Nix, Gabriel Rybeck, Carlos Scheidegger, Brandon Smith, and Suresh Venkatasubramanian. Auditing black-box models for indirect influence. *Knowledge and Information Systems*, 54(1):95–122, 2018.

[5] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E. Hinton. Neural Additive Models: Interpretable Machine Learning with Neural Nets. In *Proceedings of the 35th International*

Conference on Neural Information Processing Systems, 2021. Available at `https://openreview.net/forum?id=wHkKTW2wrmm`.

[6] Sushant Agarwal, Shahin Jabbari, Chirag Agarwal, Sohini Upadhyay, Steven Wu, and Himabindu Lakkaraju. Towards the unification and robustness of perturbation and gradient based explanations. In *Proceedings of the 38th International Conference on Machine Learning*, pages 110–119, 2021.

[7] Charu C. Aggarwal. *Outlier Analysis*. Springer Publishing Company, Incorporated, 2nd edition, 2016.

[8] Sina Aghaei, Mohammad Javad Azizi, and Phebe Vayanos. Learning optimal and fair decision trees for non-discriminative decision-making. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 1418–1426, 2019.

[9] Sina Aghaei, Andres Gomez, and Phebe Vayanos. Learning optimal classification trees: Strong max-flow formulations. *arXiv,* arXiv:2002.09142, 2020.

[10] Gaël Aglin, Siegfried Nijssen, and Pierre 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.

[11] Alfred V. Aho, M. R. Garey, and Jeffrey D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computation*, 1(2):131–137, 1972.

[12] Ulrich Aïvodji, Hiromi Arai, Olivier Fortineau, Sébastien Gambs, Satoshi Hara, and Alain Tapp. Fairwashing: the risk of rationalization. In *Proceedings of the 36th International Conference on Machine Learning*, pages 161–170, 2019.

[13] Ulrich Aïvodji, Hiromi Arai, Sébastien Gambs, and Satoshi Hara. Characterizing the risk of fairwashing. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=9PnKduzf-FT`.

[14] Ulrich Aïvodji, Alexandre Bolot, and Sébastien Gambs. Model extraction from counterfactual explanations. *arXiv,* arXiv:2009.01884, 2020.

[15] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[16] André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10):1340–1347, 2010.

[17] Yasmeen Alufaisan, Laura R. Marusich, Jonathan Z. Bakdash, Yan Zhou, and Murat Kantarcioglu. Does explainable artificial intelligence improve human decision-making? In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 6618–6626, 2021.

[18] David Alvarez-Melis and Tommi S. Jaakkola. On the robustness of interpretability methods. In *Proceedings of the 2018 ICML Workshop on Human Interpretability in Machine Learning*, pages 66–71, 2018.

[19] David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7786–7795, 2018.

[20] David Alvarez Melis, Harmanpreet Kaur, Hal Daumé III, Hanna Wallach, and Jennifer Wortman Vaughan. From human explanation to model interpretabil-

ity: A framework based on weight of evidence. In *Proceedings of the 9th AAAI Conference on Human Computation and Crowdsourcing*, pages 35–47, 2021.

[21] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. Guidelines for Human-AI Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 3:1–13, 2019.

[22] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *Proceedings of the 6th International Conference on Learning Representations*, 2018. Available at `https://openreview.net/forum?id=Sy21R9JAW`.

[23] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 35–44, 2017.

[24] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine Bias — ProPublica. URL: `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`, 2016. Accessed Dec. 20th, 2021.

[25] Sercan Ö. Arik and Tomas Pfister. TabNet: Attentive interpretable tabular learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 6679–6687, 2021.

[26] Florent Avellaneda. Efficient inference of optimal decision trees. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 3195–3202, 2020.

[27] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):1–46, 2015.

[28] Vincent Ballet, Xavier Renard, Jonathan Aigrain, Thibault Laugel, Pascal Frossard, and Marcin Detyniecki. Imperceptible adversarial attacks on tabular data. In *Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services: Data, Fairness, Explainability, Trustworthiness and Privacy*, 2019. Available at `https://arxiv.org/abs/1911.03274`.

[29] Solon Barocas and Andrew D. Selbst. Big data's disparate impact. *California Law Review*, 104(3):671–732, 2016.

[30] Solon Barocas, Andrew D. Selbst, and Manish 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.

[31] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020.

[32] Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. RelatIF: Identifying explanatory training samples via relative influence. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 1899–1909, 2020.

[33] Samyadeep Basu, Phil Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *Proceedings of the 9th International Conference on Learning Representations*, 2021. Available at `https://openreview.net/forum?id=xHKVVHGDOEk`.

[34] Andrew L. Beam and Isaac S. Kohane. Big Data and Machine Learning in Health Care. *Journal of the American Medical Association*, 319(13):1317–1318, 2018.

[35] Clément Bénard, Gérard Biau, Sébastien da Veiga, and Erwan Scornet. Interpretable random forests via rule extraction. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, pages 937–945, 2021.

[36] Clément Bénard, Gérard Biau, Sébastien da Veiga, and Erwan Scornet. SIRUS: Stable and Interpretable RUle Set for classification. *Electronic Journal of Statistics*, 15(1):427–505, 2021.

[37] Dimitris Bertsimas, Arthur Delarue, Patrick Jaillet, and Sebastien Martin. Optimal explanations of linear models. *arXiv,* arXiv:1907.04669, 2019.

[38] Dimitris Bertsimas and Jack Dunn. Optimal classification trees. *Machine Learning*, 106(7):1039–1082, 2017.

[39] Dimitris Bertsimas, Jack Dunn, and Nishanth Mundru. Optimal prescriptive trees. *INFORMS Journal on Optimization*, 1(2):164–183, 2019.

[40] Umang Bhatt, McKane Andrus, Adrian Weller, and Alice Xiang. Machine Learning Explainability for External Stakeholders. In *Proceedings of the 2020 ICML Workshop on Human Interpretability in Machine Learning*, pages 25–33, 2020.

[41] Umang Bhatt, Adrian Weller, and José M. F. Moura. Evaluating and aggregating feature-based model explanations. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 3016–3022, 2020.

[42] Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, José M. F. Moura, and Peter Eckersley. Explainable machine learning in deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 648–657, 2020.

[43] Przemyslaw Biecek and Tomasz Burzykowski. *Explanatory Model Analysis*. Chapman and Hall/CRC, 2021.

[44] Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, 5(4):2403–2424, 2011.

[45] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

[46] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[47] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[48] Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199–231, 2001.

[49] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[50] Leo Breiman and Nong Shang. Born again trees. Technical report, University of California Berkeley, 1996.

[51] Tim Brennan, William Dieterich, and Beate Ehret. Evaluating the predictive validity of the compas risk and needs assessment system. *Criminal Justice and Behavior*, 36(1):21–40, 2009.

[52] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying density-based local outliers. *ACM SIGMOD Record*, 29(2):93–104, 2000.

[53] Ozan İrsoy, Olcay Taner Yıldız, and Ethem Alpaydın. Soft decision trees. In *Proceedings of the 21st International Conference on Pattern Recognition*, pages 1819–1822, 2012.

[54] Cabinet Secretariat of Japan. Social Principles of Human-Centric AI. URL: `https://www.cas.go.jp/jp/seisaku/jinkouchinou/pdf/humancentricai.pdf`, 2019. Accessed Dec. 20th, 2021.

[55] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *Proceedings of the 2009 IEEE International Conference on Data Mining Workshops*, pages 13–18, 2009.

[56] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, 21(2):277–292, 2010.

[57] Flavio P. Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R. Varshney. Optimized pre-processing for discrimination prevention. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 3995–4004, 2017.

[58] Miguel Á. Carreira-Perpiñán and Pooya Tavallali. Alternating optimization of decision trees, with application to learning sparse oblique trees. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 1219–1229, 2018.

[59] Emilio Carrizosa, Cristina Molero-Río, and Dolores Romero Morales. Mathematical optimization in classification and regression trees. *TOP*, 29(1):5–33, 2021.

[60] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730, 2015.

[61] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):1–34, 2019.

[62] Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How Interpretable and Trustworthy Are GAMs? In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 95–105, 2021.

[63] Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. This looks like that: Deep learning for interpretable image recognition. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 8930–8941, 2019.

[64] Chaofan Chen, Kangcheng Lin, Cynthia Rudin, Yaron Shaposhnik, Sijia Wang, and Tong Wang. A holistic approach to interpretability in financial lending:

Models, visualizations, and summary-explanations. *Decision Support Systems*, 152:113647, 2021.

[65] Chaofan Chen and Cynthia Rudin. An optimization approach to learning falling rule lists. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, pages 604–612, 2018.

[66] Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. 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.

[67] Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. Jordan. L-Shapley and C-Shapley: Efficient Model Interpretation for Structured Data. In *Proceedings of the 7th International Conference on Learning Representations*, 2019. Available at `https://openreview.net/forum?id=S1E3Ko09F7`.

[68] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[69] Danielle Keats Citron and Frank A. Pasquale. The scored society: Due process for automated predictions. *Washington Law Review*, 89:1–33, 2014.

[70] Michele Conforti, Gerard Cornuejols, and Giacomo Zambelli. *Integer Programming*. Springer Publishing Company, Incorporated, 2014.

[71] Amanda Coston, Ashesh Rambachan, and Alexandra Chouldechova. Characterizing fairness over the set of good models under selective labels. In *Proceedings of the 38th International Conference on Machine Learning*, pages 2144–2155, 2021.

[72] Ian Covert, Scott M. Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 17212–17223, 2020.

[73] Nelson Cowan. The magical mystery four: How is working memory capacity limited, and why? *Current Directions in Psychological Science*, 19(1):51–57, 2010.

[74] Mark Craven and Jude Shavlik. Extracting tree-structured representations of trained networks. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, pages 24–30, 1996.

[75] Mark Craven and Jude W. Shavlik. Using sampling and queries to extract rules from trained neural networks. In *Proceedings of the 11th International Conference on International Conference on Machine Learning*, pages 37–45, 1994.

[76] Zhicheng Cui, Wenlin Chen, Yujie He, and Yixin 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.

[77] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6970–6979, 2017.

[78] Sanjeeb Dash, Oktay Günlük, and Dennis Wei. Boolean decision rules via column generation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 4660–4670, 2018.

[79] Christian Davenport. The Rashomon Effect, Observation, and Data Generation. In *Media Bias, Perspective, and State Repression: The Black Panther Party*, Cambridge Studies in Contentious Politics, pages 52–73. Cambridge University Press, 2009.

[80] Emir Demirović and Peter J. Stuckey. Optimal decision trees for nonlinear metrics. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 3733–3741, 2021.

[81] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 590–601, 2018.

[82] Botty Dimanov, Umang Bhatt, Mateja Jamnik, and Adrian Weller. You shouldn't trust me: Learning models which conceal unfairness from multiple explanation methods. In *Proceedings of the 24th European Conference on Artificial Intelligence*, pages 2473–2480, 2020.

[83] Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan 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.

[84] Jiayun Dong and Cynthia Rudin. Exploring the cloud of variable importance for the set of all good models. *Nature Machine Intelligence*, 2:810–824, 2019.

[85] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv,* arXiv:1702.08608, 2017.

[86] Finale Doshi-Velez, Mason Kortz, Ryan Budish, Chris Bavitz, Sam Gershman, David O'Brien, Kate Scott, Stuart Schieber, James Waldo, David Weinberger, et al. Accountability of AI under the law: The role of explanation. *arXiv,* arXiv:1711.01134, 2017.

[87] Mengnan Du, Ninghao Liu, and Xia Hu. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77, 2019.

[88] Dheeru Dua and Casey Graff. UCI machine learning repository. URL: `http://archive.ics.uci.edu/ml`, 2017. Accessed Dec. 20th, 2021.

[89] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence*, 2017.

[90] Bradley Efron. Prediction, estimation, and attribution. *Journal of the American Statistical Association*, 115(530):636–655, 2020.

[91] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

[92] Adam Elmachtoub, Jason Cheuk Nam Liang, and Ryan 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.

[93] Bradley J. Erickson, Panagiotis Korfiatis, Zeynettin Akkus, and Timothy L. Kline. Machine learning for medical imaging. *RadioGraphics*, 37(2):505–515, 2017.

[94] Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. *Nature Medicine*, 25(1):24–29, 2019.

[95] Christian Etmann, Sebastian Lunz, Peter Maass, and Carola Schoenlieb. On the connection between adversarial robustness and saliency map interpretability. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1823–1832, 2019.

[96] Riki Eto, Ryohei Fujimaki, Satoshi Morinaga, and Hiroshi Tamano. Fully-Automatic Bayesian Piecewise Sparse Linear Models. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, pages 238–246, 2014.

[97] FICO, Google, Imperial College London, MIT, University of Oxford, UC Irvine, and UC Berkeley. Explainable Machine Learning Challenge. URL: `https://community.fico.com/s/explainable-machine-learning-challenge`, 2018. Accessed Dec. 20th, 2021.

[98] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 20(177):1–81, 2019.

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

[100] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–21, 2010.

[101] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[102] Jerome H. Friedman and Bogdan E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3):916–954, 2008.

[103] Nicholas Frosst and Geoffrey E. Hinton. Distilling a neural network into a soft decision tree. *arXiv,* arXiv:1711.09784, 2017.

[104] Ryohei Fujimaki and Satoshi Morinaga. Factorized Asymptotic Bayesian Inference for Mixture Modeling. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pages 400–408, 2012.

[105] Kazuto Fukuchi, Satoshi Hara, and Takanori Maehara. Faking fairness via stealthily biased sampling. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 412–419, 2020.

[106] Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1607–1616, 2018.

[107] Johannes Fürnkranz, Tomáš Kliegr, and Heiko Paulheim. On cognitive preferences and the plausibility of rule-based models. *Machine Learning*, 109(4):853–898, 2020.

[108] Jingyue Gao, Xiting Wang, Yasha Wang, Yulan Yan, and Xing Xie. Learning groupwise explanations for black-box models. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pages 2396–2402, 2021.

[109] Damien Garreau and Dina Mardaoui. What does LIME really see in images? In *Proceedings of the 38th International Conference on Machine Learning*, pages 3620–3629, 2021.

[110] Damien Garreau and Ulrike von Luxburg. Explaining the Explainer: A First Theoretical Analysis of LIME. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 1287–1296, 2020.

[111] Damien Garreau and Ulrike von Luxburg. Looking Deeper into Tabular LIME. *arXiv,* arXiv:2008.11092, 2020.

[112] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[113] Amirata Ghorbani, Abubakar Abid, and James Y. Zou. Interpretation of neural networks is fragile. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 3681–3688, 2019.

[114] Amirata Ghorbani and James Zou. Data Shapley: Equitable Valuation of Data for Machine Learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2242–2251, 2019.

[115] Leilani H. Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *Proceedings of the 2018 IEEE 5th International Conference on Data Science and Advanced Analytics*, pages 80–89, 2018.

[116] Jaime Roquero Gimenez, Amirata Ghorbani, and James Zou. Knockoffs for the mass: New feature importance statistics with false discovery guarantees. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, pages 2125–2133, 2019.

[117] Jaime Roquero Gimenez and James Zou. Discovering conditionally salient features with statistical guarantees. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2290–2298, 2019.

[118] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

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

[120] David Gries, Alain J. Martin, Jan L. A. van de Snepscheut, and Jan Tijmen Udding. An algorithm for transitive reduction of an acyclic graph. *Science of Computer Programming*, 12(2):151–155, 1989.

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

[122] Riccardo Guidotti and Salvatore Ruggieri. On the stability of interpretable models. In *Proceedings of 2019 International Joint Conference on Neural Networks*, pages 1–8, 2019.

[123] Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. FastIF: Scalable influence functions for efficient model interpretation and debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10333–10350, 2021.

[124] Wenbo Guo, Sui Huang, Yunzhe Tao, Xinyu Xing, and Lin Lin. Explaining Deep Learning Models – a Bayesian Non-Parametric Approach. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 4519–4529, 2018.

[125] Gurobi. Gurobi - The Fastest Solver. URL: `http://www.gurobi.com/`, 2018. Accessed Dec. 20th, 2021.

[126] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[127] Sara Hajian, Francesco Bonchi, and Carlos Castillo. Algorithmic bias: From discrimination discovery to fairness-aware data mining. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2125–2126, 2016.

[128] Kazuaki Hanawa, Sho Yokoi, Satoshi Hara, and Kentaro Inui. Evaluation of similarity-based explanations. In *Proceedings of the 9th International Conference on Learning Representations*, 2021. Available at `https://openreview.net/forum?id=9uvhpyQwzM_`.

[129] Leif Hancox-Li. Robustness in machine learning explanations: Does it matter? In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 640–647, 2020.

[130] Satoshi Hara and Kohei Hayashi. Making Tree Ensembles Interpretable: A Bayesian Model Selection Approach. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, pages 77–85, 2018.

[131] Satoshi Hara, Kouichi Ikeno, Tasuku Soma, and Takanori Maehara. Maximally invariant data perturbation as explanation. In *Proceedings of the 2018 ICML Workshop on Human Interpretability in Machine Learning*, pages 54–58, 2018.

[132] Satoshi Hara and Masakazu Ishihata. Approximate and exact enumeration of rule models. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 3157–3164, 2018.

[133] Satoshi Hara and Takanori Maehara. Enumerate Lasso Solutions for Feature Selection. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1985–1991, 2017.

[134] Satoshi Hara and Takanori Maehara. Convex Hull Approximation of Nearly Optimal Lasso Solutions. In *Proceedings of the 16th Pacific Rim International Conference on Artificial Intelligence*, pages 350–363, 2019.

[135] Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. Data Cleansing for Models Trained with SGD. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 4213–4222, 2019.

[136] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3323–3331, 2016.

[137] Peter Hase and Mohit Bansal. Evaluating explainable AI: Which algorithmic explanations help users predict model behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5540–5552, 2020.

[138] Peter Hase, Harry Xie, and Mohit Bansal. The Out-of-Distribution Problem in Explainability and Search Methods for Feature Importance Explanations. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=HCrp4pdk2i`.

[139] Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer Series in Statistics. Springer Publishing Company, Incorporated, 2nd edition, 2009.

[140] Johannes Haug, Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Leveraging model inherent variable importance for stable online feature selection. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1478–1502, 2020.

[141] James B. Heaton, Nick G. Polson, and Jan Witte. Deep learning in finance. *arXiv,* arXiv:1602.06561, 2016.

[142] James B. Heaton, Nick G. Polson, and Jan Witte. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12, 2017.

[143] High-Level Expert Group on AI of European Commission. Ethics guidelines for trustworthy AI. URL: `https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai`, 2019. Accessed Dec. 20th, 2021.

[144] Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the 6th Annual Conference on Computational Learning Theory*, pages 5–13, 1993.

[145] Geoffrey E. Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv,* arXiv:1503.02531, 2015.

[146] Sepp Hochreiter and Jürgen Schmidhuber. Flat Minima. *Neural Computation*, 9(1):1–42, 1997.

[147] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 9737–9748, 2019.

[148] Cheng-Yu Hsieh, Chih-Kuan Yeh, Xuanqing Liu, Pradeep Kumar Ravikumar, Seungyeon Kim, Sanjiv Kumar, and Cho-Jui Hsieh. Evaluations and methods for explanation through robustness analysis. In *Proceedings of the 9th International Conference on Learning Representations*, 2021. Available at `https://openreview.net/forum?id=4dXmpCDGNp7`.

[149] Xiyang Hu, Cynthia Rudin, and Margo Seltzer. Optimal sparse decision trees. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 7265–7273, 2019.

[150] Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*, 51(1):141–154, 2011.

[151] Laurent Hyafil and Ronald L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1):15–17, 1976.

[152] Aapo Hyvärinen and Stephen M. Smith. Pairwise likelihood ratios for estimation of non-gaussian structural equation models. *Journal of Machine Learning Research*, 14(1):111–152, 2013.

[153] IBM. CPLEX Optimizer — IBM. URL: `https://www.ibm.com/analytics/cplex-optimizer`, 2021. Accessed Dec. 20th, 2021.

[154] Yasutoshi Ida, Yasuhiro Fujiwara, and Hisashi Kashima. Fast Sparse Group Lasso. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1700–1708, 2019.

[155] Alexey Ignatiev, Edward Lam, Peter J. Stuckey, and Joao Marques-Silva. A scalable two stage approach to computing optimal decision sets. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 3806–3814, 2021.

[156] Alexey Ignatiev, Joao Marques-Silva, Nina Narodytska, and Peter J. Stuckey. Reasoning-based learning of interpretable ml models. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pages 4458–4465, 2021.

[157] Maia Jacobs, Jeffrey He, Melanie F. Pradier, Barbara Lam, Andrew C. Ahn, Thomas H. McCoy, Roy H. Perlis, Finale Doshi-Velez, and Krzysztof Z. Gajos.

Designing AI for Trust and Collaboration in Time-Constrained Medical Decisions: A Sociotechnical Lens. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 659:1–14, 2021.

[158] Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, 2020.

[159] Dominik Janzing, David Balduzzi, Moritz Grosse-Wentrup, and Bernhard Schölkopf. Quantifying causal influences. *Annals of Statistics*, 41(5):2324–2358, 2013.

[160] Dominik Janzing, Lenon Minorics, and Patrick Bloebaum. Feature relevance quantification in explainable AI: A causal problem. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 2907–2916, 2020.

[161] Sérgio Jesus, Catarina Belém, Vladimir Balayan, João Bento, Pedro Saleiro, Pedro Bizarro, and João Gama. How Can I Choose an Explainer? An Application-Grounded Evaluation of Post-Hoc Explanations. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 805–815, 2021.

[162] Neil Jethani, Mukund Sudarshan, Yindalon Aphinyanaphongs, and Rajesh 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.

[163] Jeya Vikranth Jeyakumar, Joseph Noor, Yu-Hsi Cheng, Luis Garcia, and Mani Srivastava. How Can I Explain This to You? An Empirical Study of Deep Neural Network Explanation Methods. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 4211–4222, 2020.

[164] Jongbin Jung, Connor Concannon, Ravi Shroff, Sharad Goel, and Daniel G. Goldstein. Simple rules to guide expert classifications. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 183(3):771–800, 2020.

[165] Alexandros Kalousis, Julien Prados, and Melanie Hilario. Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and Information Systems*, 12(1):95–116, 2007.

[166] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1):1–33, 2012.

[167] Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. Discrimination aware decision tree learning. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, pages 869–874, 2010.

[168] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Proceedings of the 2012 Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50, 2012.

[169] Kentaro Kanamori and Hiroki Arimura. Fairness-aware decision tree editing based on mixed-integer linear optimization. *Transactions of the Japanese Society for Artificial Intelligence*, 36(4):B–L13_1–10, 2021.

[170] Kentaro Kanamori, Satoshi Hara, Masakazu Ishihata, and Hiroki Arimura. Enumeration of distinct support vectors for interactive decision making. In *Proceedings of the 2019 ICML Workshop on Human in the Loop Learning*, 2019. Available at `https://arxiv.org/abs/1906.01876`.

[171] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Hiroki 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.

[172] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Hiroki Arimura. Distribution-aware counterfactual explanation by mixed-integer linear optimization. *Transactions of the Japanese Society for Artificial Intelligence*, 36(6):C–L44_1–12, 2021.

[173] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Yuichi Ike. Counterfactual Explanation Trees: Transparent and Consistent Actionable Recourse with Decision Trees. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (to appear)*, 2022.

[174] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, Yuichi Ike, Kento Uemura, and Hiroki Arimura. Ordered counterfactual explanation by mixed-integer linear optimization. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 11564–11574, 2021.

[175] Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. Model-agnostic counterfactual explanations for consequential decisions. In *Proceedings*

of the 23rd International Conference on Artificial Intelligence and Statistics, pages 895–905, 2020.

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

[177] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse: From counterfactual explanations to interventions. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 353–362, 2021.

[178] Amir-Hossein Karimi, Julius von Kügelgen, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 265–277, 2020.

[179] Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 92:1–14, 2020.

[180] Jalil Kazemitabar, Arash A. Amini, Adam Bloniarz, and Ameet Talwalkar. Variable importance using decision trees. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 425–434, 2017.

[181] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan 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.

[182] Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. Interpreting Black Box Predictions using Fisher Kernels. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390, 2019.

[183] Been Kim. *Interactive and Interpretable Machine Learning Models for Human Machine Collaboration*. PhD thesis, Massachusetts Institute of Technology, 2015.

[184] Been Kim and Finale Doshi-Velez. Machine learning techniques for accountability. *AI Magazine*, 42(1):47–52, 2021.

[185] Been Kim, Rajiv Khanna, and Oluwasanmi Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 2288–2296, 2016.

[186] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *Proceedings of the 35th International Conference on Machine Learning*, pages 2668–2677, 2018.

[187] Pieter-Jan Kindermans, Kristof T. Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: Patternnet and patternattribution. In *Proceedings of the 6th International Conference on Learning Representations*, 2018. Available at `https://openreview.net/forum?id=Hkn7CBaTW`.

[188] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015. Available at `http://arxiv.org/abs/1412.6980`.

[189] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1885–1894, 2017.

[190] Seiichi Kondo, Keisuke Otaki, Madori Ikeda, and Akihiro Yamamoto. Fast computation of the tree edit distance between unordered trees using IP solvers. In *Proceedings of the 17th International Conference on Discovery Science*, pages 156–167, 2014.

[191] Brian Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[192] I. Elizabeth Kumar, Carlos Scheidegger, Suresh Venkatasubramanian, and Sorelle Friedler. Shapley Residuals: Quantifying the limits of the Shapley value for explanations. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=0XJDcC07tQs`.

[193] I. Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with Shapley-value-based explanations as feature importance measures. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5491–5500, 2020.

[194] Isaac Lage, Emily Chen, Jeffrey He, Menaka Narayanan, Been Kim, Samuel J. Gershman, and Finale Doshi-Velez. Human evaluation of models built for inter-

pretability. In *Proceedings of the 7th AAAI Conference on Human Computation and Crowdsourcing*, pages 59–67, 2019.

[195] Isaac Lage, Andrew Slavin Ross, Been Kim, Samuel J. Gershman, and Finale Doshi-Velez. Human-in-the-loop interpretability prior. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10180–10189, 2018.

[196] Vivian Lai and Chenhao Tan. On human predictions with explanations and predictions of machine learning models: A case study on deception detection. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, pages 29–38, 2019.

[197] Himabindu Lakkaraju, Nino Arsov, and Osbert Bastani. Robust and stable black box explanations. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5628–5638, 2020.

[198] Himabindu Lakkaraju, Stephen H. Bach, and Jure Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1675–1684, 2016.

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

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

[201] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, and Marcin Detyniecki. Issues with post-hoc counterfactual explanations: a discussion. In *2019 ICML Workshop on Human in the Loop Learning*, 2019. Available at `https://arxiv.org/abs/1906.04774`.

[202] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. The dangers of post-hoc interpretability: Unjustified counterfactual explanations. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2801–2807, 2019.

[203] Thibault Laugel, Xavier Renard, Marie-Jeanne Lesot, Christophe Marsala, and Marcin Detyniecki. Defining locality for surrogates in post-hoc interpretablity. In *Proceedings of the 2018 ICML Workshop on Human Interpretability in Machine Learning*, pages 47–53, 2018.

[204] Guang-He Lee, David Alvarez-Melis, and Tommi S. Jaakkola. Towards robust, locally linear deep networks. In *Proceedings of the 7th International Conference on Learning Representations*, 2019. Available at `https://openreview.net/forum?id=SylCrnCcFX`.

[205] Guang-He Lee and Tommi S. Jaakkola. Oblique decision trees from derivatives of relu networks. In *Proceedings of the 8th International Conference on Learning Representations*, 2020. Available at `https://openreview.net/forum?id=Bke8UR4FPB`.

[206] Kyubin Lee, Akshay Sood, and Mark Craven. Understanding learned models by identifying important features at the right resolution. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 4155–4163, 2019.

[207] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Ng. Efficient L1 regularized logistic regression. In *Proceedings of the 21st National Conference on Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference*, pages 401–408, 2006.

[208] Bruno Lepri, Nuria Oliver, Emmanuel Letouzé, Alex Pentland, and Patrick Vinck. Fair, transparent, and accountable algorithmic decision-making processes. *Philosophy & Technology*, 31(4):611–627, 2018.

[209] Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[210] David Lewis. *Counterfactuals*. John Wiley & Sons, Inc., 2013.

[211] Jiuyong Li, Hong Shen, and Rodney Topor. Mining the optimal class association rule set. *Knowledge-Based Systems*, 15(7):399–405, 2002.

[212] Qiaomei Li, Rachel Cummings, and Yonatan Mintz. Optimal local explainer aggregation for interpretable prediction. *arXiv,* arXiv:2003.09466, 2021.

[213] Xiao Li, Yu Wang, Sumanta Basu, Karl Kumbier, and Bin Yu. A debiased mdi feature importance measure for random forests. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 8047–8057, 2019.

[214] Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, and Margo Seltzer. Generalized and scalable optimal sparse decision trees. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6150–6160, 2020.

[215] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy*, 23(1), 2021.

[216] Peter Lipton. Contrastive explanation. *Royal Institute of Philosophy Supplement*, 27:247–266, 1990.

[217] Zachary C. Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[218] Tongliang Liu, Gábor Lugosi, Gergely Neu, and Dacheng Tao. Algorithmic stability and hypothesis complexity. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2159–2167, 2017.

[219] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 623–631, 2013.

[220] Gilles Louppe, Louis Wehenkel, Antonio Sutera, and Pierre Geurts. Understanding variable importances in forests of randomized trees. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pages 431–439, 2013.

[221] Yang Young Lu, Yingying Fan, Jinchi Lv, and William Stafford Noble. Deeppink: Reproducible feature selection in deep neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 8690–8700, 2018.

[222] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2:56–67, 2020.

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

[224] Scott M. Lundberg, Bala Nair, Monica S Vavilala, Mayumi Horibe, Michael J. Eisses, Trevor Adams, David E. Liston, Daniel King-Wai Low, Shu-Fang Newman, Jerry Kim, and Su-In Lee. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature Biomedical Engineering*, 2(10):749–760, 2018.

[225] Roy De Maesschalck, Delphine Jouan-Rimbaud, and Desire L. Massart. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18, 2000.

[226] Divyat Mahajan, Chenhao Tan, and Amit Sharma. Preserving causal constraints in counterfactual explanations for machine learning classifiers. In *Proceedings of the 2019 NeurIPS Workshop on Do the right thing: Machine learning and Causal Inference for improved decision making*, 2019. Available at `https://arxiv.org/abs/1912.03277`.

[227] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *National Institute of Science of India*, 2:49–55, 1936.

[228] Dina Mardaoui and Damien Garreau. An Analysis of LIME for Text Data. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, pages 3493–3501, 2021.

[229] Charles Marx, Flavio Calmon, and Berk Ustun. Predictive multiplicity in classification. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6765–6774, 2020.

[230] Hayden McTavish, Chudi Zhong, Reto Achermann, Ilias Karimalis, Jacques Chen, Cynthia Rudin, and Margo I. Seltzer. Fast sparse decision tree optimization via reference ensembles. *arXiv,* arXiv:2112.00798, 2021.

[231] Nicolai Meinshausen. Node harvest. *Annals of Applied Statistics*, 4(4):2049–2072, 2010.

[232] George 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.

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

[234] Smitha Milli, Ludwig Schmidt, Anca D. Dragan, and Moritz Hardt. Model reconstruction from model explanations. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, pages 1–9, 2019.

[235] Graziano Mita, Paolo Papotti, Maurizio Filippone, and Pietro Michiardi. Libre: Learning interpretable boolean rule ensembles. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 245–255, 2020.

[236] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. *arXiv,* arXiv:2110.11309, 2021.

[237] Brent Mittelstadt, Chris Russell, and Sandra Wachter. Explaining explanations in ai. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, pages 279–288, 2019.

[238] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012.

[239] Christoph Molnar. *Interpretable Machine Learning - A Guide for Making Black Box Models Explainable*. Lulu.com, 2020.

[240] Johanna Moore and William Swartout. Explanation in expert systems: A survey. Technical report, University of Southern California, 1989.

[241] Jonathan Moore, Nils Hammerla, and Chris Watkins. Explaining deep learning models with constrained adversarial examples. In *Proceedings of the 16th Pacific Rim International Conference on Artificial Intelligence*, pages 43–56, 2019.

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

[243] Ian Munro. Efficient determination of the transitive closure of a directed graph. *Information Processing Letters*, 1(2):56–58, 1971.

[244] Feng Nan, Joseph Wang, and Venkatesh Saligrama. Pruning random forests for prediction on a budget. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 2342–2350, 2016.

[245] Nina Narodytska, Alexey Ignatiev, Filipe Pereira, and Joao Marques-Silva. Learning Optimal Decision Trees with SAT. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 1362–1368, 2018.

[246] Dana Nau, Malik Ghallab, and Paolo Traverso. *Automated Planning: Theory & Practice*. Morgan Kaufmann Publishers Inc., 2004.

[247] Weili Nie, Yang Zhang, and Ankit Patel. A theoretical explanation for perplexing behaviors of backpropagation-based visualizations. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3809–3818, 2018.

[248] Sarah Nogueira, Konstantinos Sechidis, and Gavin Brown. On the stability of feature selection algorithms. *Journal of Machine Learning Research*, 18(174):1–54, 2018.

[249] Harsha Nori, Rich Caruana, Zhiqi Bu, Judy Hanwen Shen, and Janardhan Kulkarni. Accuracy, interpretability, and differential privacy via explainable boosting. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8227–8237, 2021.

[250] Mohammad Norouzi, Maxwell D. Collins, Matthew Johnson, David J. Fleet, and Pushmeet Kohli. Efficient non-greedy optimization of decision trees. In *Proceedings of the 29th International Conference on Neural Information Processing Systems*, pages 1729–1737, 2015.

[251] Mahsan Nourani, Samia Kabir, Sina Mohseni, and Eric D. Ragan. The effects of meaningful and meaningless explanations on trust and perceived system accuracy in intelligent systems. In *Proceedings of the 7th AAAI Conference on Human Computation and Crowdsourcing*, pages 97–105, 2019.

[252] Hidekazu Oiwa and Ryohei Fujimaki. Partition-wise linear models. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 3527–3535, 2014.

[253] Yuzuru Okajima and Kunihiko Sadamasa. Deep neural networks constrained by decision rules. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 2496–2505, 2019.

[254] Hao Yi Ong, Dennis Wang, and Xiao Song Mu. Diabetes prediction with incomplete patient data. Technical report, 2014.

[255] Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer. Deep learning for financial applications : A survey. *Applied Soft Computing*, 93:106384, 2020.

[256] Danqing Pan, Tong Wang, and Satoshi Hara. Interpretable companions for black-box models. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pages 2444–2454, 2020.

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

[258] Samir Passi and Solon Barocas. Problem formulation and fairness. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, pages 39–48, 2019.

[259] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pages 3126–3132, 2020.

[260] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. On counterfactual explanations under predictive multiplicity. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, pages 809–818, 2020.

[261] Michael J. Pazzani. Knowledge discovery from data? *IEEE Intelligent Systems and their Applications*, 15(2):10–12, 2000.

[262] Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2nd edition, 2009.

[263] Dino Pedreschi, Fosca Giannotti, Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, and Franco Turini. Meaningful Explanations of Black Box AI Decision Systems. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 9780–9784, 2019.

[264] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 560–568, 2008.

[265] Rok Piltaver, Mitja Luštrek, Matjaž Gams, and Sanda Martinčić-Ipšić. What makes classification trees comprehensible? *Expert Systems with Applications*, 62:333–346, 2016.

[266] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, Microsoft, 1998.

[267] Gregory Plumb, Maruan Al-Shedivat, Ángel Alexander Cabrera, Adam Perer, Eric Xing, and Ameet Talwalkar. Regularizing black-box models for improved interpretability. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 10526–10536, 2020.

[268] Gregory Plumb, Denali Molitor, and Ameet Talwalkar. Model agnostic supervised local explanations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2520–2529, 2018.

[269] Forough Poursabzi-Sangdeh, Daniel G. Goldstein, Jake M. Hofman, Jennifer Wortman Wortman Vaughan, and Hanna Wallach. Manipulating and measuring model interpretability. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 237:1–52, 2021.

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

[271] Alun Preece, Dan Harborne, Dave Braines, Richard Tomsett, and Supriyo Chakraborty. Stakeholders in Explainable AI. *arXiv,* arXiv:1810.00184.

[272] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: Unbiased Boosting with Categorical Features. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6639–6649, 2018.

[273] Garima Pruthi, Frederick Liu, Mukund Sundararajan, and Satyen Kale. Estimating training data influence by tracing gradient descent. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 19920–19930, 2020.

[274] Litao Qiao, Weijia Wang, and Bill Lin. Learning accurate and interpretable decision rule sets from neural networks. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 4303–4311, 2021.

[275] John Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.

[276] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.

[277] Goutham Ramakrishnan, Yun Chan Lee, and Aws Albarghouthi. Synthesizing action sequences for modifying model decisions. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 5462–5469, 2020.

[278] Karthikeyan Natesan Ramamurthy, Bhanukiran Vinzamuri, Yunfeng Zhang, and Amit Dhurandhar. Model agnostic multilevel explanations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 5968–5979, 2020.

[279] Kaivalya Rawal and Himabindu 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.

[280] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.

[281] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1527–1535, 2018.

[282] Laura Rieger, Chandan Singh, William Murdoch, and Bin Yu. Interpretations are useful: Penalizing explanations to align neural networks with prior knowledge. In *Proceedings of the 37th International Conference on Machine Learning*, pages 8116–8126, 2020.

[283] Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.

[284] Andrew Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1660–1669, 2018.

[285] Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2662–2670, 2017.

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

[287] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16:1–85, 2022.

[288] Cynthia Rudin and Şeyda Ertekin. Learning customized and optimized lists of rules with mathematical programming. *Mathematical Programming Computation*, 10(4):659–702, 2018.

[289] Cynthia Rudin and Yaron 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.

[290] Salvatore Ruggieri. Enumerating distinct decision trees. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2960–2968, 2017.

[291] Francesco Rundo, Francesca Trenta, Agatino Luigi di Stallo, and Sebastiano Battiato. Machine learning for quantitative finance applications: A survey. *Applied Sciences*, 9(24):5574, 2019.

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

[293] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[294] Shibani Santurkar, Dimitris Tsipras, Mahalaxmi Elango, David Bau, Antonio Torralba, and Aleksander Madry. Editing a classifier by rewriting its prediction

rules. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=aM7UsuOAzB3`.

[295] Andre Schidler and Stefan Szeider. SAT-based Decision Tree Learning for Large Data Sets. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 3904–3912, 2021.

[296] Patrick Schwab and Walter Karlen. CXPlain: Causal Explanations for Model Interpretation under Uncertainty. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 10220–10230, 2019.

[297] Andrew D. Selbst and Solon Barocas. The intuitive appeal of explainable machines. *Fordham Law Review*, 87(3):1085–1139, 2018.

[298] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *Proceedings of the 2017 IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[299] Lesia Semenova, Cynthia Rudin, and Ronald Parr. A study in rashomon curves and volumes: A new perspective on generalization and model simplicity in machine learning. *arXiv,* arXiv:1908.01755, 2019.

[300] Thiago Serra, Christian Tjandraatmadja, and Srikumar Ramalingam. Bounding and counting linear regions of deep neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4558–4566, 2018.

[301] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[302] Lloyd S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, II:307–317.

[303] Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvärinen, Yoshinobu Kawahara, Takashi Washio, Patrik O. Hoyer, and Kenneth Bollen. DirectLiNGAM: A Direct Method for Learning a Linear Non-Gaussian Structural Equation Model. *Journal of Machine Learning Research*, 12:1225–1248, 2011.

[304] Reza Shokri, Martin Strobel, and Yair Zick. On the privacy risks of model explanations. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 231–241, 2021.

[305] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3145–3153, 2017.

[306] Andrew Silva, Matthew Gombolay, Taylor Killian, Ivan Jimenez, and Sung-Hyun 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.

[307] Noah Simon, Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A Sparse-Group Lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

[308] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In

Proceedings of the Workshop at 1st International Conference on Learning Representations, 2014. Available at `http://arxiv.org/abs/1312.6034`.

[309] Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. Editable neural networks. In *Proceedings of the 8th International Conference on Learning Representations*, 2020. Available at `https://openreview.net/forum?id=HJedXaEtvS`.

[310] Leon Sixt, Maximilian Granz, and Tim Landgraf. When explanations lie: Why many modified BP attributions fail. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9046–9057, 2020.

[311] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial Attacks on Post Hoc Explanation Methods. In *Proceedings of the 2020 AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.

[312] Dylan Slack, Sophie Hilgard, Himabindu Lakkaraju, and Sameer Singh. Counterfactual Explanations Can Be Manipulated. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=iaO_IH7CnGJ`.

[313] Dylan Slack, Sophie Hilgard, Sameer Singh, and Himabindu Lakkaraju. Reliable Post hoc Explanations: Modeling Uncertainty in Explainability. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=rqfq0CYIekd`.

[314] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. *arXiv,* arXiv:1706.03825, 2017.

[315] Gavin Smith, Roberto Mansilla, and James Goulding. Model class reliance for random forests. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 22305–22315, 2020.

[316] Nataliya Sokolovska, Yann Chevaleyre, and Jean-Daniel Zucker. A provable algorithm for learning interpretable scoring systems. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, pages 566–574, 2018.

[317] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. In *Proceedings of the Workshop at 2nd International Conference on Learning Representations*, 2015. Available at `http://arxiv.org/abs/1412.6806`.

[318] Ilia Stepin, Jose M. Alonso, Alejandro Catala, and Martín Pereira-Fariña. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access*, 9:11974–12001, 2021.

[319] Mukund Sundararajan and Amir Najmi. The Many Shapley Values for Model Explanation. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9269–9278, 2020.

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

[321] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *Proceedings of the 2nd International Conference on Learning Representations*, 2014. Available at `http://arxiv.org/abs/1312.6199`.

[322] Kuo-Chung Tai. The tree-to-tree correction problem. *Journal of the ACM*, 26(3):422–433, 1979.

[323] Masaaki Takada and Hironori Fujisawa. Transfer learning via ℓ_1 regularization. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 14266–14277, 2020.

[324] The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems. Ethically Aligned Design: A Vision for Prioritizing Human Well-being with Autonomous and Intelligent Systems. URL: `http://standards.ieee.org/develop/indconn/ec/autonomous_systems.html`, 2019. Accessed Dec. 20th, 2021.

[325] Robert Tibshirani. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58:267–288, 1994.

[326] Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

[327] Nikolaj Tollenaar and Peter G. M. van der Heijden. Which method predicts recidivism best?: a comparison of statistical, machine learning and data mining

predictive models. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 176(2):565–584, 2013.

[328] Gabriele Tolomei, Fabrizio Silvestri, Andrew Haines, and Mounia Lalmas. 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*, pages 465–474, 2017.

[329] Richard Tomsett, Dave Braines, Dan Harborne, Alun Preece, and Supriyo Chakraborty. Interpretable to whom? a role-based model for analyzing interpretable machine learning systems. In *Proceedings of the 2018 ICML Workshop on Human Interpretability in Machine Learning*, pages 8–14, 2018.

[330] Sohini Upadhyay, Shalmali Joshi, and Himabindu Lakkaraju. Towards Robust and Reliable Algorithmic Recourse. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=AuVKs6JmBtY`.

[331] Berk Ustun. *Simple Linear Classifiers via Discrete Optimization*. PhD thesis, Massachusetts Institute of Technology, 2017.

[332] Berk Ustun, Yang Liu, and David Parkes. Fairness without harm: Decoupled classifiers with preference guarantees. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6373–6382, 2019.

[333] Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2015.

[334] Berk Ustun and Cynthia Rudin. Learning optimized risk scores. *Journal of Machine Learning Research*, 20(150):1–75, 2019.

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

[336] Berk Ustun, Michael Brandon Westover, Cynthia Rudin, and Matt T. Bianchi. Clinical prediction models for sleep apnea: The importance of medical history over symptoms. *Journal of Clinical Sleep Medicine*, 12(2):161–168, 2016.

[337] Michael van Lent, William Fisher, and Michael Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the 16th Conference on Innovative Applications of Artifical Intelligence*, pages 900–907, 2004.

[338] Kush R. Varshney. Trustworthy machine learning and artificial intelligence. *XRDS*, 25(3):26–29, 2019.

[339] Suresh Venkatasubramanian and Mark Alfano. The philosophical basis of algorithmic recourse. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 284–293, 2020.

[340] Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *arXiv,* arXiv:2010.10596, 2020.

[341] Sicco Verwer and Yingqian Zhang. Learning optimal classification trees using a binary linear program formulation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 1625–1632, 2019.

[342] Thibaut Vidal and Maximilian Schiffer. Born-again tree ensembles. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9743–9753, 2020.

[343] Daniël Vos and Sicco Verwer. Robust optimal classification trees against adversarial examples. *arXiv,* arXiv:2109.03857, 2021.

[344] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law & Technology*, 31:841–887, 2018.

[345] Alvin Wan, Lisa Dunlap, Daniel Ho, Jihan Yin, Scott Lee, Suzanne Petryk, Sarah Adel Bargal, and Joseph E. Gonzalez. NBDT: Neural-backed decision tree. In *Proceedings of the 9th International Conference on Learning Representations*, 2021. Available at `https://openreview.net/forum?id=mCLVeEpplNE`.

[346] Danding Wang, Qian Yang, Ashraf Abdul, and Brian Y. Lim. Designing Theory-Driven User-Centric Explainable AI. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 601:1–15, 2019.

[347] Fulton Wang and Cynthia Rudin. Falling Rule Lists. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, pages 1013–1022, 2015.

[348] Hao Wang, Berk Ustun, and Flavio Calmon. Repairing without retraining: Avoiding disparate impact with counterfactual distributions. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6618–6627, 2019.

[349] Jialei Wang, Ryohei Fujimaki, and Yosuke Motohashi. Trading interpretability for accuracy: Oblique treed sparse additive models. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1245–1254, 2015.

[350] Jiaxuan Wang, Jenna Wiens, and Scott Lundberg. Shapley Flow: A Graph-based Approach to Interpreting Model Predictions. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, pages 721–729, 2021.

[351] Rui Wang, Xiaoqian Wang, and David I. Inouye. Shapley explanation networks. In *Proceedings of the 9th International Conference on Learning Representations*, 2021. Available at `https://openreview.net/forum?id=vsU0efpivw`.

[352] Tong 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.

[353] Tong Wang, Cynthia Rudin, Finale Doshi-Velez, Yimin Liu, Erica Klampfl, and Perry MacNeille. A Bayesian Framework for Learning Rule Sets for Interpretable Classification. *Journal of Machine Learning Research*, 18(70):1–37, 2017.

[354] Zhuo Wang, Wei Zhang, Ning Liu, and Jianyong Wang. Scalable Rule-Based Representation Learning for Interpretable Classification. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=q_fMLfwTAJY`.

[355] Dennis Wei, Sanjeeb Dash, Tian Gao, and Oktay Gunluk. Generalized linear rule models. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6687–6696, 2019.

[356] Daniel S. Weld and Gagan Bansal. The challenge of crafting intelligible intelligence. *Communications of the ACM*, 62(6):70–79, 2019.

[357] Adrian Weller. Transparency: Motivations and challenges. In *Proceedings of the 2017 ICML Workshop on Human Interpretability in Machine Learning*, pages 55–62, 2018.

[358] Stephen F. Weng, Jenna Reps, Joe Kai, Jonathan M. Garibaldi, and Nadeem Qureshi. Can machine-learning improve cardiovascular risk prediction using routine clinical data? *PLOS ONE*, 12(4):1–14, 2017.

[359] Hilary Paul Williams. *Model Building in Mathematical Programming*. John Wiley & Sons, Inc., 5th edition, 2013.

[360] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable Bayesian Rule Lists. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3921–3930, 2017.

[361] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I. Inouye, and Pradeep K. Ravikumar. On the (in)fidelity and sensitivity of explanations. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 10965–10976, 2019.

[362] Chih-Kuan Yeh, Joon Sik Kim, Ian E. H. Yen, and Pradeep Ravikumar. Representer point selection for explaining deep neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 9311–9321, 2018.

[363] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. INVASE: Instance-wise variable selection using neural networks. In *Proceedings of the 7th International Conference on Learning Representations*, 2019. Available at `https://openreview.net/forum?id=BJg_roAcK7`.

[364] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[365] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P. Gummadi. Fairness constraints: A flexible approach for fair classification. *Journal of Machine Learning Research*, 20(75):1–42, 2019.

[366] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, Krishna P. Gummadi, and Adrian Weller. From parity to preference-based notions of fairness in classification. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 228–238, 2017.

[367] Valentina Zantedeschi, Matt Kusner, and Vlad Niculae. Learning binary decision trees by argmin differentiation. In *Proceedings of the 38th International Conference on Machine Learning*, pages 12298–12309, 2021.

[368] Hans Zantema and Hans L. Bodlaender. Finding small equivalent decision trees is hard. *International Journal of Foundations of Computer Science*, 11(02):343–354, 2000.

[369] Aleš Završnik. Algorithmic justice: Algorithms and big data in criminal justice settings. *European Journal of Criminology*, 18(5):623–642, 2021.

[370] John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLOS Medicine*, 15(11):1–17, 2018.

[371] Haoran Zhang, Quaid Morris, Berk Ustun, and Marzyeh Ghassemi. Learning Optimal Predictive Checklists. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021. Available at `https://openreview.net/forum?id=bDHBNVtB9XA`.

[372] Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262, 1989.

[373] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting CNNs via Decision Trees. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2019*, pages 6261–6270, 2019.

[374] Wenbin Zhang and Eirini Ntoutsi. FAHT: An adaptive fairness-aware decision tree classifier. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 1480–1486, 2019.

[375] Zhengze Zhou, Giles Hooker, and Fei Wang. S-LIME: Stabilized-LIME for model explanation. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2429–2438, 2021.

[376] Haoran Zhu, Pavankumar Murali, Dzung Phan, Lam Nguyen, and Jayant Kalagnanam. A Scalable MIP-based Method for Learning Optimal Multivariate Decision Trees. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 1771–1781, 2020.

[377] Hui Zou and Trevor Hastie. Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2):301–320, 2005.