



**CONTRASTIVE EXPLANATION
FOR MACHINE LEARNING**

MARCEL ROBER



Master Thesis Business Informatics

Contrastive Explanation for Machine Learning

Marcel Jurriaan Robeer

Department of Information and Computing Sciences

July 2018

Supervisors

dr. M.J.S. Brinkhuis	Utrecht University
dr. ir. J.M.E.M. van der Werf	Utrecht University
J.S. van der Waa, MSc	TNO, Delft University of Technology
Prof. dr. M.A. Neerincx	TNO, Delft University of Technology



Universiteit Utrecht

TNO innovation
for life

ABSTRACT

Introduction. Recent advances in Interpretable Machine Learning (iML) and Explainable Artificial Intelligence (XAI) have shown promising approaches that are able to provide human-understandable explanations. However, these approaches have also been criticized for disregarding human behavior in explanation. When humans ask for an explanation they generally *contrast* the given output against an output of interest. We propose to use this human tendency to ask questions like ‘Why this output (the fact) instead of the other (the foil)?’ as a natural way of limiting an explanation to its key causes.

Method. In this study we present an end-to-end approach for extracting contrastive explanations for machine learning (ML). First, we define how to apply contrastive explanation to ML. Next, we extensively study 84 iML methods in a systematic literature review to overview approaches for enhancing interpretability in machine learning, and finding method parts most suitable for contrastive explanations. We develop *Foil Trees*: a model-agnostic approach to extracting explanations for finding the set of rules that causes the explanation to be predicted the actual outcome (fact) instead of the other (foil). Moreover, we quantitatively validate Foil Trees, and empirically evaluate contrastive explanations in a user experiment.

Results. Quantitative validation showed that Foil Trees are able to accurately mimic the decision boundaries of the model it aims to explain (94% *fidelity*), generalizes well on unseen data (88% *accuracy*), provides 78% shorter explanations than their non-contrastive counterparts (*mean length* of 1.19 over 5.37) and does this all in real-time (60ms on average per explanation). Moreover, we conducted a user experiment on 121 participants to establish how contrastive and non-contrastive explanations are perceived in terms of general preference, transparency and trust. We found that contrastive explanations are preferred over non-contrastive explanations in terms of understandability, informativeness of contents and alignment with their own decision-making. This preference lead to an increased general preference and willingness to act upon the decision.

Discussion. These results suggest that it is feasible to extract generalizable, objectively transparent contrastive explanations for ML, and that contrastive explanations provide an intuitive means to create informative minimal-length human-understandable explanations that are preferable and more persuasive.

Keywords. Interpretable Machine Learning (iML) · Explainable Artificial Intelligence (XAI) · Contrastive Explanation · Decision Trees · Model-Agnostic · Human Interpretable Machine Learning · Foil

ACKNOWLEDGEMENTS

In December 2017, I ventured on a journey to tackle defining and applying contrastive explanations to machine learning. Over the months, this project challenged me by requiring me to be a *generalist* as well as a *specialist*. While still in an early stage, explainable artificial intelligence (XAI) and interpretable machine learning (iML) have shown many promising approaches to tackle a pivotal issue in our current and future society, and uncovered many challenges for the years to come. I am glad to have learned from the difficulties and opportunities in this subject area, and that I was able to contribute to tackling the issue even if ever so slightly.

This study would not have been possible without a number of people pivotal throughout my research. First, Matthieu Brinkhuis for the insightful discussions we had, your enthusiasm for my research and the subject area, and your valuable insights regarding psychology, experimentation and psychometrics. Second, Jan Martijn van der Werf, for your never-ending guidance and support—challenging me even beyond my information science bachelor years. Third, I would like to thank TNO for providing me with the opportunity and means for conducting my master thesis in an exciting and ambitious subject area. Specifically, I would like to thank Jasper van der Waa—your relentless ideas, support and enthusiasm that went into this project. Mark Neerinx, your extensive knowledge and eagerness for new ideas and exploring beyond my own discipline. My colleagues Ajaya, Arthur, Jurriaan, Martin and Riccardo in Applied AI, who provided me with valuable discussions and reviews. Finally, a special mention goes to my fellow graduate interns in Soesterberg, who taught me a good coffee and fun distractions during the day are pivotal to a good end result.

Last but not least, I would like to thank my girlfriend, friends and family, who encouraged me to excel this year. Without them, none of this would have been possible.

Marcel Robeer

CONTENTS

Abstract	i
Acknowledgements	ii
Abbreviations	vii
1 Introduction	1
1.1 Context	2
1.2 Research Approach: Design Science	2
1.3 Thesis Outline	5
2 Background: Interpretable Machine Learning	7
2.1 Artificial Intelligence and Machine Learning	7
2.2 Key Terminology	10
2.3 Why Explain?	12
2.4 What Makes an Explanation Interpretable?	15
2.5 Making Machine Learning Interpretable	18
2.6 Practical and Ethical Considerations	22
Summary	23
Literature Study	25
3 Contrastive Explanation	27
3.1 Why P rather than Q?	27
3.2 Contrasts, Facts, Foils and Counterfactuals	28
3.3 Contrastive Explanation in Machine Learning	30
3.4 Conclusion	33
4 Interpretable Machine Learning Methods	35
4.1 Literature Review Method	35
4.2 Taxonomy of Interpretable Machine Learning Methods	38
4.3 Explanatory Representations	41
4.4 Methods	45
4.5 Discussion & Conclusion	60

Foil Trees	65
5 Foil Trees: Contrastive Explanations for Machine Learning	67
5.1 Contrastive Explanation as Binary Classification	67
5.2 Implementation	71
6 Quantitative Validation	75
6.1 Performance Metrics	75
6.2 Setup	76
6.3 Results	79
6.4 Discussion & Conclusion	80
Empirical Evaluation	83
7 Experiment	85
7.1 Glucose Level Prediction Data	85
7.2 Experimental Design	87
7.3 Results	93
7.4 Discussion	97
7.5 Conclusion	99
Conclusion and Outlook	101
8 Conclusion and Outlook	103
8.1 Conclusion	103
8.2 Future Directions	105
Bibliography	107
Appendix	119
A Machine Learning Model Evaluation	119
B Machine Learning Techniques	123
C Validation Data	133
D Experiment Materials	135
Publications	157

ABBREVIATIONS

AI	Artificial Intelligence
CNN	Convolutional Neural Network
DNN	Deep Neural Network
DT	Decision Tree
DR	Decision Rule
FI	Feature Importance
HCI	Human-Computer Interaction
iML	Interpretable Machine Learning
ML	Machine Learning
NN	Neural Network
PAL	Personal Assistant for a healthy Lifestyle
PDP	Partial Dependence Plot
RL	Reinforcement Learning
RNN	Recurrent Neural Network
T1DM	Type 1 Diabetes Mellitus
XAI	Explainable Artificial Intelligence

1 INTRODUCTION

Machine Learning (ML) systems can learn analytical models from data without being explicitly programmed. In recent years, these data-driven algorithms have proven to be very successful in various tasks—sometimes even outperforming humans—, and are therefore commonly used to guide important decisions [43]. Even though these decisions affect humans, to optimize task performance ML models often become too complex to be intelligible to humans [43, 46]. Already two decades ago, this lack of interpretability was acknowledged as a main obstacle for real-world applications [159]. Even today, enabled by an increase in model complexity, a lack of intelligible ML models is a prevalent barrier to adoption in areas such as healthcare [24] and fully autonomous cars [93]. *Interpretable ML* (iML)¹ is the area of research tackling these issues by creating interpretable ML models.

Problem statement

With recent successes of black-box ML models (e.g., [125, 167]), the field of iML has seen renewed attention with the *Explainable Artificial Intelligence* (XAI) program of DARPA [65]. XAI intends to create a new suite of ML techniques that produce more interpretable ML models while maintaining a high level of performance [65]. While the recent resurgence of XAI models has shown promising approaches (see e.g., [10, 72, 148]), the field has been criticized for disregarding (i) *what constitutes a good explanation* and (ii) *the requirements of the intended users* explanations are made for [43, 124]. Insights from the social sciences can be employed to improve explanations, thereby boosting understanding and trust [110, 123, 124].

One key notion is that explanation-seeking behavior is generally *contrastive* [109, 124]: when asking why a model predicted an output humans (implicitly) ask for a contrast against an expected output. For example, in the context of a self-driving car an explanation for ‘Why did you hit the tree?’ might require a different explanation for ‘Why did you hit the tree rather than the person?’ than for ‘Why did you hit the tree rather than staying on the road?’ A good explanation should therefore consider the *fact* (output) as well as the *foil* (expected output) [109].

We hypothesize that explanations in iML can benefit from explicitly considering the fact and foil. Not only can we present a concise explanation that addresses the explanation expected by the end-user, but the expected outcome for a given situation can be used to improve the model. To test our hypothesis, we aim to define and implement *contrastive explanations* for ML.

¹ To avoid confusion with *Interactive ML* (IML), we use the abbreviation iML for interpretable ML.

Contributions

This research project has scientific as well as societal contributions. The *scientific contributions* are (i) the definition of a framework for contrastive explanations in ML, (ii) a systematic review of iML methods, (iii) applying contrastive explanation to ML in a general-purpose manner to create accurate yet convincing explanations, and (iv) empirical insights into user preferences and perceptions regarding non-contrastive versus contrastive explanations. In addition, the *societal contribution* is the improvement of intelligibility of ML models in various contexts, with a specific focus on healthcare.

1.1 Context

Type 1 Diabetes Mellitus (T1DM) is a chronic condition where insufficient insulin is produced by the body, affecting blood glucose levels that may lead to complications that threaten health and survival [200]. The majority of T1DM occurs in children and adolescents, with an estimated 130,000 cases of T1DM in children aged 0-14 in Europe [178, 200]. While the cause for T1DM is currently unknown, daily administration and management of blood glucose levels are known to reduce the risk of negative health effects [40, 200].

PAL project

The *Personal Assistant for a healthy Lifestyle* (PAL) project develops a system for children aged 8-14, their parents and health care professionals that aims to advance the self-management of T1DM for when children reach adolescence [131].^{2,3} The PAL system comprises an Embodied Conversational Agent (ECA) robot and (mobile) avatar, a set of mobile health (mHealth) applications (e.g., diabetes diary, educational quizzes), and dashboards for the caregivers (i.e., health care professionals and parents). All parts are interconnected with a shared knowledge-base and reasoning mechanism [55].

PAL wants to develop an action-recommendation ML algorithm that suggests children with the most appropriate action (e.g., food item or insulin injection) to administer the blood glucose levels throughout the day. In addition, they want this action recommendation accompanied with a convincing explanation using iML in order to enhance trust and thereby adherence to the action-recommendation algorithm.

1.2 Research Approach: Design Science

The research project is best characterized as *Design Science* [198]. Design science problems are improvement problems, that aim to design an *artifact* (object of study) to improve a problem *context*. We iteratively investigate the problem context (*investigation*), and design artifacts (*design*) using existing problem-solving knowledge and newly gained knowledge from our investigation.

² <http://www.pal4u.eu/>

³ The PAL project is funded by European Union Horizon2020 grant nr. 643783-RIA.

In this section, we define the overall aims and objectives of this research project. Next, we overview the research questions to address the overall aims. Subsequently, we detail the corresponding research methods. Lastly, we overview the six project phases that we use to structure the project.

Aims and objectives

We make explicit the artifact and problem context by defining the design problem using the template of Wieringa [198]:

This research aims to improve explanations for machine learning models by devising and implementing a contrastive explanation technique for the selection of food items that is interpretable for caregivers in order to provide convincing explanations that enhance trust and understanding of the PAL system.

Research questions

To address our main objective, the research is structured with a main research question (RQ):

RQ *How can contrastive explanation improve the explanatory capability of machine learning methods?*

The main research question is answered by four subquestions (SQs). The first SQ revolves around surveying existing literature on methods for interpretable machine learning. SQ2 considers how to automatically acquire explanations most suitable for creating contrastive explanations. SQ3 expands on determining the foil (expected output) when an explanation is desired. For SQ4, we use insights from SQ1 and SQ2 to generate explanations that are human-understandable in the problem context. Lastly, we evaluate the quality of our output:

SQ1 *What are current approaches to automatically acquire explanations from machine learning systems?*

We survey existing methods for explanation of machine learning models to catalog the research that has been conducted for benefit of researchers in this area, and create a thorough understanding of their strengths and weaknesses. We use their insights for the development of our own approach.

1.1 *How do iML methods enhance interpretability in machine learning?*

1.2 *How can these methods be used for contrastive explanation in machine learning?*

SQ2 *How to automatically acquire decision rule explanations from machine learning systems?*

Based on the findings in SQ1, we develop a method for generating explanations for an arbitrary outcome (i.e., *fact* or *foil*) that holds at least locally. This method forms the foundation for our approach for contrastive machine learning explanation.

SQ3 *How can we determine the foil for a required explanation?*

First, we aim to distinguish which foils to consider for a machine learning task (3.1), and consequently develop a method for automatically acquiring these foils (3.2) when an explanation is requested:

- 3.1 *Which foils can we distinguish for machine learning?*
- 3.2 *How to automatically acquire these foils for a required explanation?*

SQ4 *How can we generate human-understandable contrastive explanations?*

Using the approaches developed in SQ2 and SQ3.2, we develop an end-to-end method for contrastive explanation (4.1) and transform the final explanation to a human-understandable format (4.2):

- 4.1 *How to generate contrastive explanations, given an output to explain and the foil?*
- 4.2 *How to present the contrastive explanation in a human-understandable way?*

SQ5 *What is the quality of the contrastive explanations?*

We assess the quality of the approach through a quantitative functional performance assessment (5.1) as well as human evaluation of interpretability (5.2) by assessing the perceptions of the intended users of our approach in an experiment:

- 5.1 *How well do the contrastive explanations perform in terms of explanation length, generalizability, truthfulness to the model it aims to explain, and speed?*
- 5.2 *How are contrastive explanations versus non-contrastive explanations perceived in terms of general preference, transparency and trust?*

Research methods

To answer our research questions, we apply various research methods. Table 1.1 relates the research methods to the research questions to which they apply. SQ1, SQ2, SQ3.2, SQ4, and SQ5 are all part of the *design*, while SQ3.1 focuses on *investigation*. Through a literature research, we aim to overview scientific insights and use these to design and develop a method. The method is implemented in a proof-of-concept tool. We systematically review iML methods, combining methods found in partial overviews of subareas and snowballing these articles to find new methods. Moreover, we perform a functional evaluation of the truthfulness of the explanation to the model using a controlled experiment. Lastly, whether the method is deemed interpretable is evaluated through expert evaluation in a user experiment.

Table 1.1 Research methods used to answer the subquestions (SQs)

Research method	SQ1		SQ2	SQ3		SQ4		SQ5	
	1.1	1.2		3.1	3.2	4.1	4.2	5.1	5.2
<i>Systematic review</i>	✓	✓							
<i>Literature research</i>			✓	✓	✓	✓	✓		
<i>Controlled experiment</i>								✓	
<i>User experiment</i>									✓

Project phases

We structure the research around six main phases, based on the design science research process model [140]. Each of the first five phases addresses one or more subquestions (SQs), while during communication (phase 6) we answer the main research question (RQ). We define the six phases as follows:

1. Background	Define and overview iML and contrastive explanation for ML	SQ1-3
2. Method review	Literature review of existing iML methods	SQ1
3. Design	Design and development of an end-to-end automated approach for contrastive ML explanation	SQ2-4
4. Implementation	Implementation of the approach in a proof-of-concept tool	SQ4
5. Evaluation	Functional and human-grounded evaluation of the approach, applied to the use-case in form of the proof-of-concept tool	SQ5
6. Communication	Writing final version of thesis and papers based on project	RQ

1.3 Thesis Outline

The remainder of this thesis is structured as follows. First, as a theoretical baseline Chapter 2 expands on *interpretability and explanation*, both from a philosophical and psychological perspective, as well as from its application in computer science through iML. The remainder of this thesis is subdivided into four parts, as depicted in Figure 1.1.

Part I reports the *literature review*. Chapter 3 defines contrastive explanation concepts and applies them to machine learning (ML). Chapter 4 provides a systematic review of existing interpretable machine learning (iML) methods and provides insights for operationalizing contrastive explanation in machine learning.

Part II describes and validates our *Foil Trees*: our approach for automated contrastive ML explanations. Chapter 5 details the approach and its implementation. Chapter 6 then quantitatively validates our approach using a controlled experiment.

Part III presents a *human experiment* that assesses perceptions and preferences for contrastive versus non-contrastive explanations, and discusses the results and their implications.

Part IV concludes the thesis and present future directions for research.



Figure 1.1 Thesis overview

2 BACKGROUND: INTERPRETABLE MACHINE LEARNING

Interpretability is seen as an important criterion for the adoption of machine learning (ML) models in real-world tasks [43, 157]. Even though it is such a pivotal concept, the desired quality of interpretability is ill-defined: ‘What is interpretability?’, ‘Why is it important?’, and ‘How can we measure interpretability?’

In this chapter, we use the scientific understanding of interpretability and explanation as a baseline to answer these questions in scope of this work. We overview scientific literature from the fields of artificial intelligence (AI), philosophy of science, social sciences (in particular psychology) and human-computer interaction (HCI). First, Section 2.1 introduces the main ML concepts and four types of ML. Section 2.2 introduces key iML terminology as a baseline. Section 2.3 expands on reasons for explanation. Section 2.4 examines measuring interpretability and key factors to consider for making ML explanations interpretable. Section 2.5 overviews approaches to make ML interpretable. Lastly, Section 2.6 discusses practical and ethical considerations fundamental to interpretable ML.

2.1 Artificial Intelligence and Machine Learning

Artificial Intelligence (AI) is the study of the synthesis and analysis of computational agents that intelligently solve tasks given information and a set of actions or decisions [141, 158]. *Machine Learning* (ML) is a subfield of AI that focuses on computational approaches of learning analytical models from data to optimize task performance [127]. In this thesis, we focus on ML specifically unless mentioned otherwise. In this section, we first define general ML concepts, and then introduce the types of ML tasks.¹

Machine learning concepts

In order to learn from data, an ML algorithm takes *inputs* and produces a *model* $\hat{f}(\cdot)$ so that it can transform inputs into *outputs*. It is defined as

¹ For a more detailed overview of ML model evaluation and ML algorithms, we refer the reader to Appendix A and Appendix B, respectively.

Definition (*Machine learning model*).

A machine learning model is a mapping $\hat{f} : \mathcal{X} \rightarrow \mathcal{Y}$, where

- $x_1, x_2, \dots, x_N \in \mathcal{X}$ are the model *inputs* (data points), and;
- $y_1, y_2, \dots, y_N \in \mathcal{Y}$ are the model *outputs*.

We refer to \mathcal{X} as the *input space*, and \mathcal{Y} as the *target space*. We write $\hat{f}(X) = Y$ as a shorthand for $\{\hat{f}(x) \mid x \in X\} = Y$.

The input x_i consists of a set of m *feature-value* pairs (a_i, v_i) , where a_i is a *feature* (also referred to as attribute or variable) and v_i is the value from the domain of a_i . A feature describes the domain of a single element in the input—e.g., a number in the input data may represent a color, the speed a car was driving at, or the presence of a disease. The values of these feature are either restricted in the number of values they can take on (*discrete*) or they can take on an infinite number of values (*continuous*).

Let Λ be the universe of all feature labels, and \mathcal{V} be the universe of all feature values.

Definition (*Feature set*).

A feature set is a tuple (\mathcal{A}, L, R) where

- Feature space \mathcal{A} is the Cartesian product of m features $\mathcal{A} = a_1 \times a_2 \times \dots \times a_m$.
- $L : \mathcal{A} \rightarrow \Lambda$ is a function describing the (potentially empty) label for each feature.
- $R : \mathcal{A} \rightarrow P(\mathcal{V})$ is a mapping of a feature to a range of potential feature values for that feature. A feature $a \in \mathcal{A}$ is
 - *discrete* if $R(a)$ can take on a countable set of feature values, or;
 - *continuous* otherwise.

We denote the value assigned to an input x_i for attribute $a_i \in \mathcal{A}$ as $v_i \in \mathcal{V}$.

Types of machine learning

We distinguish four types of tasks in ML [158]: *supervised learning*, *unsupervised learning*, *semi-supervised learning*, and *reinforcement learning*.

Supervised learning. Supervised tasks are given a *training set* of N input-output pairs $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ where each y was generated by an unknown function $y = f(x)$. In supervised learning, the outputs corresponding to inputs are often referred to as *labels*. The goal is to discover a function $\hat{y} = \hat{f}(x)$ that approximates the true function $f(\cdot)$ with respect to a performance or loss function. We can then assess the performance of the model by giving it previously unseen data (the *test set*) and comparing the predicted labels to the actual labels. The test set is not used during training and contains labeled instances taken from the full data set, typically between 20% and 40% of the total data set.

Supervised tasks are subdivided into two subtasks based on the type of output. When y can take on a finite set of values (classes, such as *spam* or not *spam*), the task is called *classification*.

Problems with two classes are referred to as *binary classification*, while problems with more classes are denoted *multi-class classification*. If y is a number (e.g., the expected temperature), the problem is referred to as *regression analysis*.

Unsupervised learning. Unsupervised tasks aim to find structure in the data without receiving any feedback. These tasks do not have labeled inputs, i.e. it is trained on N inputs x_1, x_2, \dots, x_N . Examples of tasks are *clustering*—discerning clusters within the data where inputs are similar—, *association rule mining*—discovering interesting relationships between data—, and *anomaly detection*—finding instances that deviate from the normal pattern in the data, such as fraudulent bank transactions.

Semi-supervised learning. Semi-supervised tasks have only part of their input data labelled. Typically, the number of unlabeled examples greatly outnumbers the number of labeled examples. The annotation with ground truth labels is very expensive (or even infeasible) for some tasks. However, it might be possible to create a more accurate supervised model if we can also leverage the provided unlabeled data. In semi-supervised learning the initial goal is to identify labels for all inputs. After all inputs have been labeled, the problem can be treated as a supervised learning task.

Reinforcement learning. Unlike the other types of learning, in reinforcement learning (RL) an *agent* can affect the *environment*. In RL, an *agent* learns from the *environment* by performing actions, getting new information and a reinforcement (reward or punishment) from the environment [176]. The algorithm aims to maximize the cumulative reward, by finding state-action pairs obtained from the environment that were most responsible for acquiring the reward. To find the optimal *policy*—the action-selection model—, the agent has to balance between *exploitation* (maximize reward you know about) and *exploration* (choose to acquire more information about the environment).

Formally, RL can be defined as a *Markov-Decision Process* (MDP). An MDP is a 4-tuple $\langle S, A, T, R \rangle$, with a set of possible states S and possible actions A . At each discrete time step t , the agent observes a state $s_t \in S$ and selects an action $a_t \in A_s$, where $A_s \subseteq A$ is the set of possible actions in s . Next, the agent immediately receives a scalar reward r_t and changes its state to s_{t+1} . $T : S \times A \times S \rightarrow \mathbb{R}_{\geq 0}$ is the probability of ending up in one state from another state. The goal is to find an optimal *policy* $\pi : S \rightarrow A$ (a mapping of probabilities for selecting the next action) that maximizes the expected cumulative reward. The immediate reward is retrieved using a *reward function* $R : S \times A \rightarrow \mathbb{R}$ [158, 176].

Take for example a scenario where we want let a robot solve a maze by navigating a grid. The states (S) are all the spots in the maze without a wall; its actions (A) may be *forward*, *backward*, *turn left*, or *turn right*. In a given spot we may restrict the potential actions, as the robot cannot move into a spot with a wall. To solve the maze, the robot should reach the finish line, where we give a reward of 100 points. In addition, some states result in a reward of another 10 points. When performing RL, we can give it the goal of maximizing the total number of points. If instead

we would rather find the shortest path to the finish line, we might only consider a reward at the finish line and optimize for this reward.

2.2 Key Terminology

Interpretable Machine Learning (iML) and Explainable Artificial Intelligence (XAI) revolve around several key concepts. In this section we define these concepts as they are used in this study.

Defining interpretability

Despite its importance there is no consensus in scientific literature regarding the *definition* of interpretability of ML [43, 110]. Interpretability in ML suffers from a severe *underspecification* of the problem [110]. It is viewed as a comparative measure, where one technique or model may be more interpretable than another [123]. In its contents, interpretability conveys some sense of how the model works [110], governing how easy it is to understand a model in a particular domain [189].

In the field of iML, three works attempt to define interpretability. Doshi-Velez and Kim [43, p. 2] define interpretability as the “ability to explain or to present in understandable terms to a human.” Miller [123, p. 12] describe interpretability as “the degree to which an observer can understand the cause of a decision.” Ross et al. [155, p. 1] claim that “[a] model is interpretable if it provides explanations for its predictions in a form humans can understand”. We strengthen our definition of interpretability by employing two key insights prevalent throughout the aforementioned definitions:

1. A model is made interpretable by providing *explanations* for its decisions [43, 123, 155], and;
2. These explanations should be provided in *understandable terms to a human* [43, 155].

Here, we view a *decision* as a generic output of AI systems. By combining these insights, in context of this work we define interpretability as follows:

Definition (*Interpretability*).

The ability of a machine learning system to explain its decisions in understandable terms to a human.

Similar to Miller [123], we use the terms *interpretability* and *explainability* interchangeably. In addition, we equate *comprehensibility* with interpretability.

Model interpretability. In interpretability, it is important to distinguish between the interpretability of the *model* itself, and the interpretability of the *learning algorithm* [157]. Note that in our definitions, we specifically refer to the interpretability of the model. While the learning algorithm may be interpretable, this does not imply that the resulting model will be interpretable as well. For example, the simple functions in each neuron in a neural network may be easily explained to an end-user, but this does not elucidate how the full network captured intricate patterns in the

input data to arrive at a decision. Still, we argue that when a model is deployed factors such as the chosen algorithm, training metadata and descriptive statistics of the input data set may provide a general sense of a sound approach to solve the problem at hand—and therefore are significant information to convey [46].

Explanation

An essential part of interpretability is the ability to provide an explanation. Like interpretability, the term explanation also suffers from a multitude of definitions. Therefore, in the following paragraphs we consider the definitions of explanations used in scientific literature and use these to define explanation in scope of this work.

Explanation: a process and a product. Explanation is both a *process* and a *product* [123]. It encompasses both the explanatory act, as well as the thing to be explained (*explanandum*) and the explanatory response (*explanans*) [156]. However, this distinction is not always clear. This is further complicated because explanation captures a plurality of concepts, even in human psychology itself [28]. Most scientific literature on explanation emphasizes the product of explanation (*explanandum*), instead of the explanatory act. Accordingly, our characterization of explanation also revolves around the explanatory product.

Scientific and everyday explanation. While a lot of work in the philosophy of science focuses on *scientific explanation*, there is more to explanation than just scientific explanation [156]. Scientific explanation focuses on creating theoretical and empirical statements that describe phenomena in the world around us [71]. Explaining general phenomena requires a rigid process of hypothesis formulation and experimental or formal justification [71]. In contrast, everyday explanations are often between individuals and attempt to convey some sort of understanding of why particular facts (e.g., events or decisions) occurred [86, 123]. They are *singular*, in that they describe one particular event instead of general laws [156]. Between individuals, these everyday explanations are a currency to exchange beliefs [112]. In this thesis, we focus on everyday explanation. While this distinction may not always be crisp, this means that we strive to formulate everyday explanations that convey understanding between two agents—rather than explanations that hold generally in the form of laws.²

Defining everyday explanation. In its simplest form, an *everyday explanation* (also referred to as *ordinary explanation*) is defined as an answer to a ‘why’-question [e.g., 108, 112, 123, 156, 191]. However, this definition disregards what constitutes an explanation itself.

In general, the scientific consensus is that the main content of an explanation is a causal relationship between events and the outcome. Mandel [117] argues that ‘why’-questions are causal questions. Thus, an explanation should encompass a rationale that provides the information needed to establish causation [68]. Miller et al. [124] also emphasizes the importance of causation

² Please note that this does not imply that non-scientific explanations are not generalizable to more cases, but rather that we are not seeking explanations that hold for all cases of a phenomenon.

in explanation, stating that the attribution of causes to events is necessary to provide explanations. While we do acknowledge that some authors argue that not all explanations are about causal relationships [e.g., 86], in the context of ML models—where outputs are caused by the given inputs—we restrict our definition of explanation to causal explanation. Even when some explanations contain non-causal elements, causal elements seem to be preferred for explanation [86].

While these definitions of explanation are drawn from the philosophy of science, there is no single agreed-upon definition of explanation for machine learning (ML). Within ML, Miller [123] interprets explanation as *post-hoc interpretability*—i.e., interpretability of the decision by the machine learning model after it was made. Even though at first glance post-hoc interpretability seems an ML equivalent of asking a ‘why’-question to explain a decision, using this definition poses the issue that we would define explanation in terms of explanation, and vice versa. Instead, we view interpretability as a concept that is defined in terms of explanation. Our definition of explanation merely considers the causal contents of the everyday explanation, and thus we define it as follows:

Definition (*Explanation*).

A causal description of why a decision occurred.

Explanatory agents. This definition of the explanation leaves out one key aspect of everyday explanation: that the explanation is conveyed between two agents. These two agents are distinguished using two roles:

- ▶ *Explainer*: the agent giving the explanation, and;
- ▶ *Explainee*: the agent receiving the explanation, with the aim to understand it.

Please note that given the aforementioned definition of interpretability, the explainee is a human.

2.3 Why Explain?

To understand a decision people ask for an explanation of *why* it happened, *how* it happened, and even how it could have been *prevented* [117]. Analogously, if we are to use automated decisions by ML models in real-world applications, human-understandable explanations are an intuitive means to establish why these decisions were made in non-interpretable models.

Already two decades ago, a lack of interpretability has been acknowledged as a main obstacle for real-world applications [159]. Even today, a lack of intelligible ML models is a prevalent barrier to adoption in areas such as healthcare [24] and fully autonomous cars [93] due to the increase in model complexity. With a lack of interpretability, explanations provide a basis to verify whether the system adheres to quality properties other than its decision performance.

For real-world applications, it is important to consider *why* interpretability is necessary. When designing for interpretability, the motivating goal it aims to address should be taken into account. In this section, we overview the five main motivators for why interpretability is required in ML.

They can be categorized as *transparency, trust, model debugging, learning from the model* and *ethics & regulation*.

Transparency

The inability for most end-users to understand ML decisions originates from *opaque* models [21, 74, 110]. Transparency is the opposite of opacity [110]. It can be used as a means to counter opacity, and is a motivator for interpretability in itself. Burrell [21] distinguishes three root causes of opacity prevalent in ML models:

- opacity as a result of intentional hiding of the decision-making process by corporations and institutions as proprietary protection;
- opacity stemming from the technical illiteracy of everyday users—who cannot understand the underlying mechanisms—, and;
- opacity as a result of the mathematical paradigm to handle data diverging from the demands of human-scale reasoning and comprehension.

These three root causes can be tackled by providing transparency regarding the decision-making process, and explaining the inner workings of the model in terms and cognitive chunks understandable for the explainee.

In addition to being a motivator for interpretability, transparency facilitates other desiderata of interpretability. For example, a more transparent model may increase trust on behalf of users [107], and allows for the examination of ethical and legal concerns [42].

Trust

In order for people to apply ML models in real-world tasks, they need to *trust* that it will perform well and understand beforehand when it will not perform well [43, 74, 110, 123]. Verification that the model uses sensible inferences for decision making can ensure reliability, robustness and safety when it is applied on a real-world task [43, 160]. In particular, we can uncover the cases where the model fails in reality [46], observe whether it still performs even when greatly varying inputs [43, 160], and is able to withstand attempts of users gaming the system [110].

Trust is strongly tied to the *generalizability* (or *transferability*) of a model: whether a model can be applied in new contexts, and generalizes well beyond its training data. The difficulty of generalizability lies in that learning problems suffer from the *bias-variance trade-off*: they have to simultaneously minimize bias (how consistently correct the model performs) and variance (the variation, or spread, of outputs), to prevent overfitting or underfitting on the input data [118]. If it follows patterns of the input data too well the model might not hold in other contexts, while if the model is too simple it might not capture the complex relationships in real-world tasks [118]. This is further complicated by systematic bias in input data, where prejudices of humans in the data selection process can misrepresent the relations it aims to capture [52]. For example, while we may not want to discriminate against a minority—the fact that they are a minority will result in a systematic underrepresentation of them in the input data. With the general objective

of maximizing performance, an easy method to differentiate a large group of people from the rest is to (unknowingly) use the fact that they are a minority as a distinguishing characteristic—potentially discriminating against a group as a consequence of distributions in input data.

Lastly, ML tasks might optimize for *mismatched objectives* [43]. In these cases, their goal is to maximize performance on a proxy objective, as the actual objective cannot be captured or measured easily. Though the ML model may perform well on the proxy, this does not mean that it is optimal performance for the actual objective per se. For example, while the overall goal may be to improve patient health, we may only try to optimize for a measurable property such as blood sugar level or body temperature. With the risk of mismatched objectives, iML can assist in uncovering whether the decision is also optimal for the actual objective or solely the proxy objective.

Model debugging

When we uncover the weaknesses of our model, we can use this knowledge to improve the model itself. Especially when there is high-dimensional or large-scale data, debugging or improving the model becomes a futile endeavor [21]. Complex inputs that each subtly change the model output may require intricate understanding for model improvement [21]. If we uncover what our model is doing and explain *why*, it becomes easier to improve it [160]. Interpretability can facilitate model improvement in three ways. Firstly, explanations are used for model debugging: in cases where the model failed we might ask why it failed [86]. Secondly, model interpretability can help in detecting and preventing human bias in the model and data [160]. Thirdly, making the model interpretable allows for comparing different models and architectures [160].

Learn from model

Machine learning revolves around learning an analytical model from data. Automatically learning from vast amounts of data may lead to a model discovering intricate patterns that may not be apparent to human observers. Instead of merely deploying such a model in practice, the rationale used by the ML system has been used to inform decision makers and further understanding of the problem domain [88, 160]. A model may be informative to end-users in other ways than merely its outputs [110]. For example, if we want to determine whether a patient has a disease based on their DNA, we may be just as interested in knowing the parts of DNA responsible for said disease than just the fact they are likely to obtain said disease.

From a scientific point of view, by asking explanations we can acquire new knowledge for scientific understanding [43, 110]. Especially when the goal is to make strong claims about phenomena, such as *causality*, a more comprehensive understanding of the problem is required [110]. Note that our account of explanation may prove to be insufficient to justify explanations for scientific phenomena, as we focus more on everyday explanation.

Ethics & legislation

Nowadays, decisions made by ML systems are applied in settings where they affect humans. They may decide on your job application, whether you can be granted a loan, or the media you

are presented with on a website. This has raised questions about the *fairness* and *ethics* of ML decision-making [43, 46, 74, 110, 194]. ML systems may exhibit undesirable *discrimination*, as past data may contain correlations we do not want to capture in our system [35, 46]. Decisions based on direct characteristics, such as gender, race, religion and sexuality are not only unwanted, but oftentimes even illegal [46]. Even if these are not directly present, they may be unknowingly uncovered by indirect data, such as occupation data or postal codes [46]. With discrimination as a concern, *privacy*-preserving ML systems may address potential issues through transparency of the individual’s input data being used for automated decision-making [35]. With no direct access to input data, as often this data is the key resource for an information-driven organization, interpretability of individual decisions can help address privacy concerns [35].

Recently, this societal consideration of ethics in ML has been backed up by compliance to legislation: the European Union’s *General Data Protection Regulation* (GDPR) ‘right to explanation’ [137]. The GDPR requires all algorithms automatically processing EU citizen’s data to explain their decisions as a safeguard for their rights and freedom [63]. Legally, according to the GDPR explanation should provide us with ‘meaningful information about the logic of processing’ [46, 63]. Though it is not the main aim of this work, it is also important to note that the ML model should be *contestable* [63, 110]: allowing for decisions to be altered when their propositions are falsified.

2.4 What Makes an Explanation Interpretable?

Miller [123] argues that some techniques or models may be more interpretable than others. With the goal of increasing interpretability, we require a form of benchmarking by measuring interpretability. The difficulty of interpretability lies in that it is *subjective* [11, 157]: what one person might call an interpretable explanation may not be interpretable to another person at all. First, we overview methods for measuring the interpretability of ML models. However, measurements can usually only be performed after a method for interpretability was already created. Therefore, in the subsequent subsection we examine other factors that may improve interpretability in specific contexts or in general.

Measuring interpretability

Doshi-Velez and Kim [43] remark that there are currently two predominant approaches to evaluating interpretability: (i) *human evaluation* and (ii) *evaluation through a quantifiable proxy*.

Human evaluation. The first approach, human evaluation, revolves around evaluating the interpretable ML system in the context of an application [43]. If the system is deemed useful by people in some practical context—or a simplified version of it—, then it must also be useful in a more general context. Interpretability must take into account the explainee’s limitations and perceptions [43, 148]. While more expensive to test than quantifiable proxies, human-grounded evaluation allows to comparatively evaluate the interpretability of one model to another, regardless of model class (e.g., decision tree or classification rules) [157]. When testing on a real task, performing experiments or questioning domain experts poses the benefit of gaining actual insights of whether

the approach will work in the envisioned application domain [43].

One caveat is that it may be difficult to separate interpretability from the impression of *validity of the model* [157]. Personal beliefs are a major determinant in whether a model is perceived to be interpretable, and these may hinder the evaluation of the general method. These prejudices may also reach beyond the domain the model describe, as an explainee may have bias for or against ML systems based on their past experiences with them [42]. As noted earlier, another downside of human evaluation methods is that they are more *expensive*: they require time and effort to perform, and demand high standards of evaluation design [43].

Quantifiable proxy. In the second approach interpretability is measured by using a formal complexity measure [157]. Examples of such a complexity measure are the depth of a decision tree, the minimum description length of a decision tree, the number of rules or rule depth in a classification rule list, the number of features in regression, or the Akaike Information Criterion. Generally, measures are based on a claim that a (class of) model(s) are inherently interpretable [43]. Next, they may achieve simplicity by constraining the model size in some manner.

While formal complexity measures may be easy to measure—and therefore pose a practical advantage—, one might argue that they are a too simplistic representation for an imprecise concept as interpretability [157]. The downside of formal complexity measures is that (i) they may only be applicable to one class of models (such as decision trees or classification rules), or (ii) they only provide a very coarse measure of complexity—completely disregarding the semantics (i.e., contents) of the model [51, 157].

Formal complexity measures are based on the heuristic that simpler explanations are preferred [136, 157]. By reducing the number of features, parameters, or elements in an explanation it becomes easier to understand. Authors employing formal complexity measures base their assumption on an *Occam's razor*-type (*parsimonious*) argument: when multiple explanations hold, we should pick the simplest explanation that is consistent with the data [136, 158]. This notion of the human favor of simplicity is also backed up by empirical evidence. Huysmans et al. [79] found that when people were asked to perform judgments using models overall the comprehensibility, answer accuracy and answer for decision tables, decision trees and classification rules tended to increase when models became smaller.

However, Freitas [51] cautions for extreme simplicity in the explanation of complex problems (which may invoke an averse reaction of the explainee), and that the acceptance of an explanation may depend even more on the degree of alignment with the explainee's background knowledge rather than rule size. In causal explanation, Pacer and Lombrozo [136] found that people prefer minimizing the number of unexplained causes (*root simplicity*) rather than the minimizing the total number of causes (*node simplicity*), even more so when the causal effect of the root nodes (the topmost nodes in a causal chain) is strong.

When using a quantifiable proxy to measure interpretability, the central question remains: 'How simple should the explanation be?' As a starting point, one might consider the number of cognitive entities (or facts) in an explanation. Commonly, considerations are based on well-known

Table 2.1 Evaluation approach advantages and disadvantages

Approach	Advantages	Disadvantages
Human evaluation	Independent of model class	Expensive to conduct, influenced by human preferences
Quantifiable proxy	Cheap to obtain	Model class dependent, disregards model contents

heuristics such as the finding of Miller [122] that humans are capable of processing 7 ± 2 cognitive entities, and Jennings et al. [81] who found that humans are limited in estimating the relatedness of more than two variables. Beyond such generic heuristics, the key consideration is the *interpretability-accuracy trade-off* [148, 165]: in order for a model to be more sparse (interpretable), it usually has to be less faithful to the input data.

The evaluation trade-off. Both methods of evaluation have pros and cons [43, 157]. Table 2.1 overviews the pros and cons for each approach. On the one hand, human evaluations may show the interpretability and adoption in the target domain regardless of model class, but are more expensive to conduct and be influenced by human preferences. On the other hand, quantifiable proxies pose the benefit of being easy to calculate, but are typically model class related and only give a coarse indicator—disregarding model semantics.

Consequently, evaluation of interpretable ML should strike a balance between the two approaches—combining the advantages of each approach. Because they give a rough indicator of the approximate length we should strive for in an explanation, quantifiable proxies are a good basis for deciding on an approximation for explanation length. Still, claims of the explanatory capability of a method in practice should also be backed up by human-grounded empirical evidence.

Factors for interpretability

While the preceding methods of evaluation may give insights into interpretability after devising an interpretable ML approach, there are other factors that can be taken into account to create an optimal explanation.

Doshi-Velez and Kim [43] hypothesize that the type of explanation sought may depend on whether it aims to explain the whole model (*global*) or a single decision (*local*), the problem area, the severity of the problem it aims to solve, time constraints of the explainee, and the explainee’s expertise. These considerations are backed up by Edwards and Veale [46], who also argue that ML explanations are restricted by the type of explanation (*global* or *local*), the multi-dimensionality (complexity) of the problem domain, and the type of explainee.

A second factor is that when considering that people who are outliers given the data may most commonly encounter an unexpected decision, for exactly these people providing an explanation may be the most difficult [46]. This is an obvious drawback of statistical generalization. However, it poses the benefit of uncovering exactly that: the reason the decision deviated from expectations

might be because the underlying model is too simplistic and overgeneralizes—providing a basis for contesting the decision and/or model improvement.

Lastly, another factor that is regularly taken into consideration as improving the understandability and acceptance of relationships are *monotonicity constraints* [51]. Simply put, a relationship is monotonic if it either always increases or always decreases. Monotonicity constraints prevent having to explain cases where first, for instance, the likelihood of an event increases with the occurrence of another event, but suddenly decreases after a given threshold. In addition to being less demanding to comprehend, monotonic relationships can also pose the benefit of preventing effects of complex correlations to affect the explanation of a single relationship [35].

2.5 Making Machine Learning Interpretable

Now we have a sense and understanding of what interpretability is, why it is important and how it can be measured, the practical issue of conceiving methods for interpretable ML (iML) remains. In this section, we overview considerations and methods for making ML interpretable. First, we enumerate evaluation criteria for comparing iML methods. Second, we present two general strategies to make ML interpretable. Third, we draw from iML and human-computer interaction (HCI) literature to gain insights in ways to present explanations in human-intelligible ways. Next, we discuss how explanation depends on the informativeness of inputs and outputs. Finally, we argue how interactivity can enhance explanations.

Evaluation criteria

While interpretability is imperative in iML, it is only one dimension of essential criteria that iML methods should consider. In black-box rule extraction literature, Craven and Shavlik [33] list six criteria to evaluate algorithms. These criteria state that the significance of iML methods depend on the interpretability of the explanation, how accurate they represent what they explain, the scalability of the method, whether the method can be applied in general, and its direct availability for usage. We adapt these criteria, and extend them to hold more for iML methods in general:

- ▶ **Interpretability** (*comprehensibility*): the extent to which representations are human understandable, evaluated through human evaluation and/or quantifiable proxies (see Section 2.4).
- ▶ **Fidelity**: the extent to which representations accurately describe the underlying model, evaluated by comparing the iML model output to the underlying model output using traditional performance measures (as detailed in Appendix A).³
- ▶ **Accuracy**: how accurately extracted representations can predict unseen examples—also measured using traditional performance measures.⁴
- ▶ **Scalability**: the ability of the iML method to scale well with large numbers of input features, large data sets and complex underlying models. Scalability is measured as the running

³ Please note that when there is no underlying model—i.e. the iML method is transparent/interpretable by design—fidelity cannot be measured.

⁴ As we will see later, not all iML methods (e.g., ones showing feature importance, or which neurons are activated for a given instance) can be used to predict new instances.

time (run-time complexity or empirical evaluation), but also considers how well the interpretability with size.

- ▶ **Generality:** the degree of dependence of the iML method on special training regimes, specific underlying models or model architectures, types of data, and type of ML tasks. To become widely accepted, iML methods must exhibit a high level of generality.
- ▶ **Software availability:** the extent to which researchers make their methods available to potential users—e.g., as open source software. Readily available methods can easily be adopted by end-users, algorithm variants can be examined, and new functionality can be developed conveniently.

Even though scalability, generality and software availability hold for an iML method in general, interpretability, fidelity and accuracy depend on the specific case the iML method is applied to. Therefore, they are to be assessed on a case-by-case basis. Moreover, learning iML representation is challenging because interpretability is generally a competing objective with accuracy and fidelity [100]. For optimal accuracy and fidelity, the representation should take into account the nuances and exception in predicting an output. Instead, interpretability favors simplicity—a more comprehensible explanation it typically a simple one. When using a quantifiable proxy for interpretability, these trade-offs can be made automatically using *multi-objective optimization*.

Explanatory strategies

To reach interpretability in ML through explanation, there are two explanatory strategies that can be exercised [74]: *descriptive* and *persuasive*.

- ▶ **Descriptive explanations** focus on creating a truthful explanation of the model (maximum *fidelity*). They attempt to accurately convey the inner workings of the model, but as a repercussion may have to give in to their interpretability. To measure faithfulness, quantifiable proxies are the principal means.
- ▶ **Persuasive explanations** aim to be as convincing to the explainee as possible. In order to do so, they may employ explainee characteristics, preferences and knowledge. With the goal of explainee persuasion, they often have to give into their model fidelity, in order to maximize interpretability. To measure how convincing an explanation is, the sole means is through human evaluation.

When making an ML model interpretable, one picks the strategy based on the goal of the explanation. Descriptive explanations are most appropriate when the aim is reducing opacity, model debugging, learning from the model or ethical considerations. Especially when trust or assurance of transparency are a concern, persuasive strategies should be applied. In some considerations, e.g. full transparency or legal issues, both strategies might be required to reach the desired level of interpretability.

Presenting explanations

Explanations may take on a multitude of forms. Here, we overview four representations pervasive in human-computer interaction (HCI) and iML literature. Note that these representations are not mutually exclusive: often textual and graphical methods are combined. While in some cases one representation can pose benefits over another, the design challenge is to match the right representation to the explanatory task at hand [150]. To illustrate the types of explanations, Figure 2.1 shows example explanations of each type for why a model classified an image as a 'cat'.

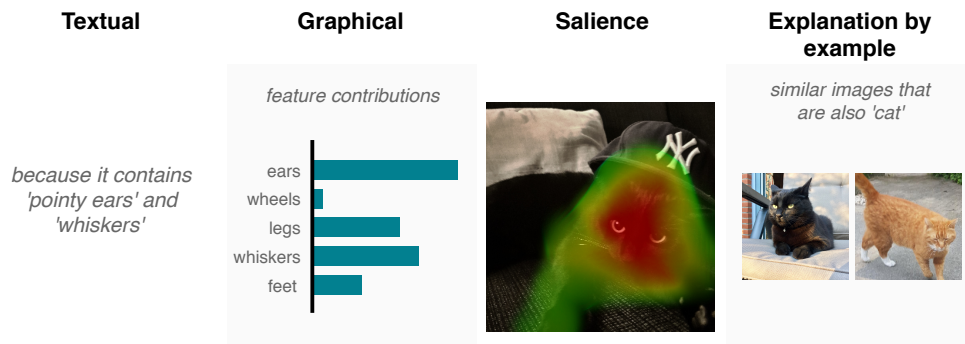


Figure 2.1 Types of explanations for an image classified as 'cat'

Textual. Humans often justify decisions verbally [110]. As such, explanation through text is the most common approach for providing explanations in decision-support systems and of ML models [e.g., 99, 184]. Textual representations range from very restricted (formulas of code) to natural language. The preference for textual representations seems omnipresent, as even when there is no textual understanding in an underlying model approaches are specifically made such that they provide natural language explanations. For instance, Hendricks et al. [72] use class definitions of the objects classified in a image classifier to give a descriptive explanation of why an image is classified as such. Ehsan et al. [47] adds rationalizations to actions taken in the training process as if a human had performed these actions—giving the ML model the ability to generate natural language explanations for all actions it performs.

Graphical. Textual representations have difficulty in explaining complex networks of data or to accurately convey relationships. Visualization is a very popular approach for global explanation (i.e., explaining the entire model at once) or to explain individual decisions with charts (often used in combination with textual explanations). Liu et al. [111] overviews *visual analytics* techniques for understanding, diagnosing and refining ML models. Note that these are frequently interactive visualizations—allowing the explainee to interact with the explanation. Goldstein et al. [62] use chart-based visualizations to explain classifications while showing the underlying patterns of data. Patel et al. [138] support ML model developers with a tool that employs graphical visualizations for model debugging. In decision-support systems *charts* are often used in conjunction with text for explaining decisions [99, 184]. Krause et al. [95] use charts to visualize statistics

to enhance textual rule explanations. Kulesza et al. [98] employ charts to visualize numerical, time and musical key data with confidence intervals as explanations for song recommendation predictions.

Saliency. Saliency maps show an explanations as the relative importance of individual elements compared to their neighbor elements. They are commonly presented as an overlay on their original domain—e.g., the input image or input e-mail. Layer-wise Relevant Propagation [7] obtains the relevant input features of a (deep) neural network. In images, this can create a heatmap showing the relevant pixels for a decision. Saliency maps are often used to show the *visual attention* of an ML model—i.e., where the model is looking at. For images and text, Ribeiro et al. [148] show the relevant features of a decision as parts of an image or highlighted text, respectively. Zahavy et al. [205] combine saliency maps for all decisions, and cluster similar saliency maps to generalize the decisions of a reinforcement learning agent—explaining the entire model as a simplified Markov Decision Process (MDP).

Explanation by example. Explanation by example is a computer equivalent of human-style justification through analogy. They are often used in combination with other explanatory representations, but in unsupervised learning this is the most recurrent approach. While it may not be a full explanation, showing similar cases to a decision equips an explainee with insights to build their own mental model of the ML system [202]. Kim et al. [88] use similar examples to extract features (e.g., colors in images or ingredients in recipes) common in subspaces in a cluster. Yang and Shafto [202] explain individual decisions through *Bayesian teaching*: showing similar cases in the input data to let the explainee learn about the model’s decision-making process.

Explanation informativeness

Explanation in ML depends on the informativeness of inputs and outputs. As long as these inputs and outputs refer to some non-opaque concept (e.g., have clear feature names and labels)—a clear explanation can be constructed. When citing causes for explanation, this can be equated to opacity resulting in explanations to fail. To take an example from Lipton [108, p. 55]: “[...] suppose that the decayed insulation in the high-voltage lines running between the walls caused the fire in the department and is the event mentioned on page 17 of the accident report. If someone asks why the fire occurred, it is unhelpful to say ‘Because of the event reported on page 17 of the accident report.’” Likewise, in ML it may be very unhelpful to explain a decision as ‘because variable X was 0’—when variable X refers to gender and 0 means that the gender is male.

If either is not informative in itself, there are still presentation methods to reach a useful explanation. For example, in clustering (where the output are similar cases) *explanation by example* can provide an explanation that is more informative than that a case belongs to ‘cluster X ’. As another example, image inputs the inputs are integers representing pixel colors on a screen—not that informative individually. Showing the input image and visualizing a saliency map can explain what an ML model is looking for in an image. However, arguably a more verbose explanation is one where a textual understanding of the concepts on the image is conveyed.

Interactive explanations

Most current explanations in iML are *static* [1]. Static explanations merely provide a single explanation, that conveys a single message as a ground-truth explanation. However, when automating the iML process *interactive explanations* allow a user to explore the ML systems' behavior freely. Through interactive visualization and dialogue systems, interaction can be a powerful means to enable people to iteratively explore and gather insight from large amounts of complex information [182]. In order to get a sense of an entire model or part of the model, explainees can, for example, drill down from high-level explanation to very detailed ones [12]. Examples of current HCI-focused approaches that leverage this principle are explanatory debugging [99]—showing explanation of text predictions with evidence (e.g., important words and folder size) and the prediction probability—, Ravelo [177]—that offers instance-level explanations by interactively exploring features and data items for text data—, and Prospector [94]—a visual analytics tool showing partial dependence diagnostics and localized inspections of why data points are predicted as they are.

2.6 Practical and Ethical Considerations

In our quest to make the field of ML more interpretable, some authors caution for disregarding the general trust, opacity and ethical concerns while we tackle these desiderata (e.g., [14, 74, 110]). In a desire to optimize for human understandability, there may be a trade-off between creating *optimal* and *desirable* explanations. While not the main aim of this study, we deem it important before continuing our work to briefly touch upon potential ethical and practical issues in the research area of iML.⁵

Firstly, as Lipton [110] notes, in taking human explanation methods as a golden standard for iML research we might replicate human shortcoming in conveying and assessing interpretations. The best explanation usually does not include the true cause [91]. With a preference of humans to align explanations with their own beliefs, they may come up with a rationale of why that explanation may be correct to them [92]. This preference highlights the potential decoupling between explanation validity and perceived validity. In addition, explanation is often applied as a tool for social comparison, where the content of the explanation can suffer under the attempt of an explainer to create strategic misunderstanding in order to retain the social disparity between explainer and explainee [147].

Secondly, as an ethical consideration Binns et al. [14], Herman [74] and Lipton [110] warn for optimizing for convincing explanations. These *persuasive explanations* (see Section 2.4) have the goal of convincing a human that the underlying model is correct, and may use misleading but plausible explanations to do so. While persuasive explanations are an appealing approach to gain acceptance of ML methods in practice, when decisions affect humans in critical ways we may want to be reserved in applying persuasive explanation methods. Rather than actually tackling explanatory goals such as trust and ethics & legislation, these methods would only attempt to convincingly trick humans into thinking they are up to the expected standards—'nudging' explainees into ac-

⁵ FAT-ML is the field of ML specifically focused on *fairness, accountability and transparency*.

cepting a justification. Even when it is not the direct goal to persuade, benchmarking based on human perception of explanation quality can have severe consequences. For instance, in clinical decision support systems (CDSSs) Bussone et al. [22] found that comprehensive explanations lead to over-reliance on the system. Merely including a high number of cognitive entities in an explanation lead the explainees to believe that the system better understood the problem, and therefore accepted the explanation even though the outcome was incorrect.

Lastly, making methods interpretable may be at odds with the broader objective of AI to surpass human ability to solve complex tasks [110]. This is the same misalignment of ML methods' computational approach to handle vast amounts of data and human-style thinking that causes opacity [21]. In attempting to make very complex decisions, we may have to give into interpretability (i.e., make an accuracy-interpretability trade-off) in order decipher the complex task. iML methods need to be evaluated and used with caution, as ensuring their robustness is an often unaddressed and currently unsolved problem [59]. Still, in practical considerations notions of ethics and legal concerns should be examined on a case-to-case basis. As Burrell [21] and Hildebrandt [75] argue: in some cases perhaps the only method for ensuring trust in a decision-making process is not to use any ML methods at all.

Summary

- *Interpretability* is the ability of an ML system to explain its decisions in understandable terms to a human. An *explanation* is a causal description of why a decision occurred. Explanations are conveyed from an *explainer* to an explainee *explainee*.
- Explanations in ML have the goal of transparency, trust that the model will work well in practice, model debugging, learning from the model, and ethical & legal considerations for the decision-making process it is part of.
- Criteria for iML evaluation are *interpretability*, *fidelity*, *accuracy*, *scalability*, *generality* and *software availability*. Interpretability can be measured using either *human evaluation* or a quantifiable proxy—but often require a trade-off between the two.
- Explainability can be reached through *descriptive* (maximizing fidelity) and *persuasive* (with as main goal explainee persuasion, often giving into fidelity) strategies. Persuasive strategies are most appropriate for trust or transparency. Descriptive strategies should be used for reducing opacity, model debugging, learning from the model or ethical considerations.
- iML explanations come in the form of text, images, salience overlays on the input data, or showing groups of example instances that result in the same decision. Explanations are typically presented using the model inputs and outputs, and therefore they depend on their informativeness. With an automated explanation method, interactive visualizations and dialogue systems can enhance explainability by exploration of the ML systems' behavior.

LITERATURE STUDY

We define contrastive explanation, facts, foils and counterfactuals for machine learning. Furthermore, we systematically review 84 interpretable machine learning methods published between 1994 and April 2018.

3 CONTRASTIVE EXPLANATION

Perhaps one of the most essential questions for interpretability in machine learning is: ‘What constitutes a *good* explanation?’ A predominant finding from research in the philosophy of science and social sciences is that explanation-seeking behavior is generally *contrastive* [109, 123]: when asking why a model predicted an output humans (implicitly) ask for a contrast against an expected output. A full causal explanation for an event is undesired and may even be infeasible. In practice, humans always provide partial explanations instead of full ones [156]. Instead of a full account, we expect an explanation for the key factors that caused the given output instead of another.

This section introduces the main concepts of contrastive explanation, and defines how their meaning in the philosophical theories and psychology can be transferred to meaningful definitions in machine learning (ML). Section 3.1 introduces the general notion of contrastive explanation. Section 3.2 defines the main concepts in contrastive explanation. Lastly, Section 3.3 applies these concepts to ML to define a framework for using contrastive explanation in ML.

3.1 Why P rather than Q?

Authors in contrastive explanation argue that all ‘why’-questions are of the form ‘Why *P* rather than *Q*?’ [9, 108, 109, 117]. Here, *P* is the actual decision and *Q* is the expected decision that did not occur. Even if *Q* is left implicit—i.e., one might ask ‘Why *P*?’—the expected response is contrastive. The question ‘Why did John hit the tree?’ may be asked with the intention of knowing (i) ‘Why did John rather than Alice hit the tree?’, (ii) ‘Why did John hit the tree rather than avoiding it?’ or (iii) ‘Why did John hit the tree rather than hitting a car?’ All these variations of expected events require different explanations. Therefore, to create a good explanation one should take into account the event that occurred as well as the counterfactual event. To avoid confusion, throughout the remainder of this thesis we will adopt the terminology used by Lipton [109] and refer to actual event *P* as the *fact*, and the expected (class of) event(s) *Q* as *foil*.

Humans are well-equipped in detecting a foil from the language and tone in a conversation, and the context surrounding the explanation [123]. However, to automatically obtain the foil from an explainee requesting explanation may be difficult or even infeasible [123]. Nevertheless, for an explanation to be deemed useful in practice it is important that the explainer and explainee understand the expected output—e.g., given the question ‘Why was this animal classified as a cat?’ the explanation ‘Because it has four legs’ may not be a good explanation if the foil is ‘rather than a dog’ instead of ‘rather than an insect’. Even if the level of abstraction in ML can produce causal chains with very few elements, as people are known to process contrastive explanations and apply counterfactual thinking in day-to-day life (see psychological evidence by e.g., [180]), a contrastive explanation will be more intuitive and therefore arguably more valuable [123]. This

is especially true when considering that humans tend not to ask for explanations as long as the output is as expected. Rather, people seek explanation in case of an abnormality [154], such as model failure or suspected unfair decision making.

An additional argument for contrastive explanations is that they pose the benefit that they only require the difference between the fact P and foil Q as a sufficient explanation. Instead of giving a full causal attribution (i.e., all causes C for event E), contrastive explanations may be easier to derive without needing to know all of the potential causes for a given event [109, 123]. This is not only a benefit for human explanation, but it may be leveraged by automatic explanation to reduce the required computational power and time for an automated explanation [123].

3.2 Contrasts, Facts, Foils and Counterfactuals

Contrastive explanation is very closely tied to counterfactual thinking [121]. Despite their close ties, contrastive explanation and counterfactual thinking use somewhat different terminology. Therefore, in this section we overview counterfactual thinking and contrastive explanation, and clarify some terminology as we apply it throughout the remainder of this work in order to avoid confusion.

Counterfactual thinking and contrastive explanation

In counterfactual thinking, one might hypothetically alter a past event and see how it changed the outcome, i.e. ‘What if I had done X instead of Y , would the outcome still be Z ?’ [121]. Even though the process to arrive at the explanation may initiate with a different negation, the elements changed for explanation turn out to be the same. In contrastive explanation, one first decides on a *counterfactual outcome event* (foil; counterfactual to E) and then determines the *counterfactual causes* (counterfactual to C) that best explain the change of said outcome event. Lipton [108, p. 691] refers to the counterfactual causes as the *difference condition*: “we explain why event P occurred rather than event Q by giving information about the causal history of P that would not have applied to the history of Q if Q had occurred”. Conversely, in counterfactual explanation the alteration (i.e., an alternative action or the absence of the action) is a *counterfactual cause* C such that the outcome is changed to a different outcome E . However, note that a single counterfactual alteration of the outcome may require multiple alterations of causes. As a result, we draw from insights in counterfactual thinking, but view them from a contrastive explanation perspective.

Contrast class, fact and foil

With emphasis on the outcome, we first consider the class of all possible outcomes in a given context: the *contrast class*. Van Fraassen [191] argues that in ‘Why P rather than Q ?’ the contrast class is a class of propositions X comprising of P together with all alternative events to P (including Q). Barnes [9] claims that we might view the contrast class as *state spaces*, i.e. the possibilities of the system.

We define the *contrast class*, *fact* and *foil* for a decision (event) \mathcal{E} that was caused by a set of causes

C:

Definition (*Contrast class, Fact, Foil*).

A *contrast class* \mathcal{F} are all possible alternatives to a decision given the context (i.e., the range of values for a decision E). The *fact* is the actual decision $f \in \mathcal{F}$, while the *foil* is any other member of the contrast class that is not f —i.e., $g \in \mathcal{F} \setminus \{f\}$.

Counterfactuals and semifactuals

To explain specific events that happened, *actual causation* is important [67]: we must distinguish the causes that are actually responsible for the outcome. Identifying these causes can be done through the *backgrounding* (also known as *discounting*), where we view a cause as non-relevant because it also holds for the foil [86, 123]. *Counterfactuals* are any change in causes that results in a different outcome event. In specific, because there might be many counterfactuals we adopt the *closest possible world* view [194], where we look for the minimal change required in causes such that the outcome will change from fact to foil. In addition, in constructing a counterfactual explainers look for the most *most import features* to best describe the difference between fact and foil [123].

The defining feature of counterfactuals is that the corresponding outcome changes, and are therefore not the same as *semifactuals*—where even though the fact is altered the outcome does not change [153]. This is related to the discussion of *compatible* versus *incompatible contrasts*—whether the foil happening means that the fact does not [61]. Based on the generally adopted accounts of counterfactual thinking and contrastive explanation, we exclusively focus our work on *incompatible* contrasts—i.e., we assume that the model output may not be in the fact state and foil state simultaneously.

Considering ML, we define a counterfactual as a change in the input data resulting in the decision to be changed from fact to not-fact [194]. In contrastive explanation, we are specifically interested in changing the fact to the foil, and therefore define it as follows:

Definition (*Counterfactual*).

A *counterfactual* is a change in the input data resulting in the decision to be changed from fact to foil.

Note that multiple counterfactuals may hold for a particular change from fact to foil. Any member of this set of counterfactuals is a *targeted contrastive explanation*:

Definition (*Contrastive Explanation*).

A *contrastive explanation* is any member of the set of *counterfactuals* resulting in the decision being changed from fact to foil.

Example

Consider an example of deciding on a patient health status. Figure 3.1 shows this case as a causal graph. This directed acyclic graph (DAG) consists of a set of events (E) depicted by boxes connected by arrows, that indicate conditional dependencies. Rounded boxes indicate causes C , while non-rounded boxes is the health outcome \mathcal{E} . Under consideration is the contrast class \mathcal{F} consisting of four potential outcomes: Diabetes, Heart disease, Overweight or Healthy. Red boxes with a double border indicate the causes C that hold for *fact* Diabetes, while hatched boxes indicate what holds for *foil* Heart disease. The contrastive explanation is what counterfactually holds for the foil, but not for the fact—i.e., Heart condition.

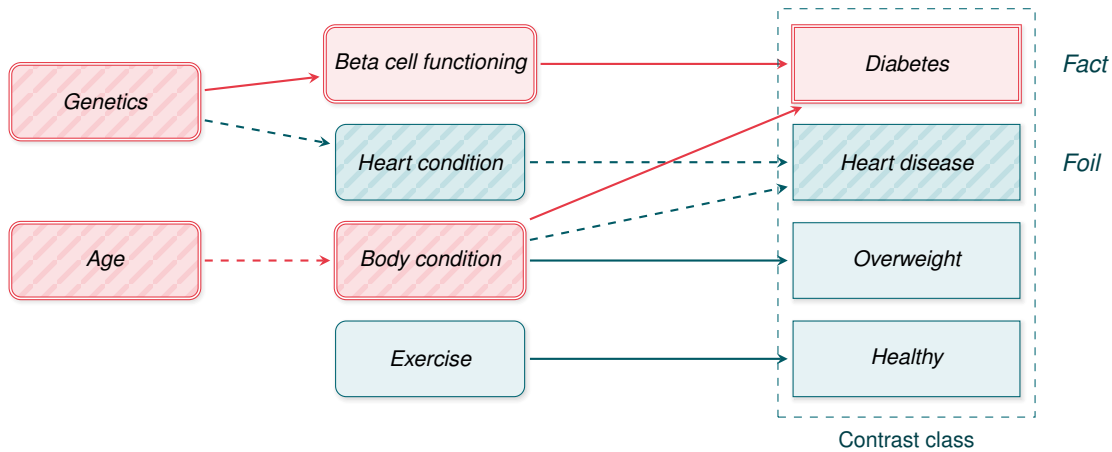


Figure 3.1 Example causal graph for explaining ‘Why diabetes (fact) instead of heart disease (foil)?’

3.3 Contrastive Explanation in Machine Learning

Though contrastive explanation is a grounded process in human psychology [153], to the best of our knowledge no automated contrastive explanation methods are available for ML. In this section, we address causal models in ML and define foils for ML.

Causal models

Conveying an explanation using causes is the de-facto standard in philosophical theories and social sciences. Nevertheless, automatically creating an accurate causal model is difficult in practice [194]. Ensuring that relationships in ML are causal requires an underlying theory or experimental manipulations to support that variable A causes variable B , instead of for instance an unobserved confounder C being responsible for changes in levels of both A and B [169].

However, the assumption we can make for all ML models is that their *outputs* are caused by their *inputs* (see Section 2.2). In unsupervised, semi-supervised and supervised ML we may take an independence assumption of all variables—similar to *naive Bayes* in classifiers. In reinforcement learning, when we are able to affect the environment we may perturb the inputs and observe

the actual outputs—thereby uncovering a causal chain. If the explanation is post-hoc (i.e., after the decision was made), our information may be limited to only historical observations—thus we may not be able to make causal claims. By viewing the inputs of ML as causes for their outputs, ML explanations can be communicated in terms of their inputs and outputs.

Foils

There is a large body of work on types of counterfactuals and selecting counterfactuals for explanation¹ that is not applied to ML in specific. Some work does focus on counterfactuals in ML: counterfactuals have been adopted as a means for deciding on causal structures for explaining data (see e.g., [84]) and to generate explanations of the minimal changes required to change a classification outcome [130].

What these counterfactual approaches have in common is that they do not take into account the actual foil. Their emphasis is on changing input data, so that the outcome changes from the actual output to any other output. In other words, they are not *targeted contrastive explanations*. Based on literature, in this section we define our account of potential foils in ML. For a more comprehensive overview, we do not merely explain the potential foils but rather the *contrast class* of outcomes—i.e., describe the class for the fact *and* the foil. Based on the rendition of Barnes [9] that a contrast class describes a state space of outcomes (i.e., the value range for the target space), we describe the state space of outcomes a contrast class may encompass. Table 3.1 summarizes the contrast classes per type of ML.

Table 3.1 Contrast classes per type of machine learning

Type	Contrast class(es)
Supervised learning	
<i>Classification</i>	▶ Class labels
<i>Regression analysis</i>	▶ All values (compare two outcome values) ▶ Binned values ▶ Statistical norm
Unsupervised learning	▶ Clusters ▶ Anomaly, not-anomaly
Reinforcement learning	▶ Reachable states ▶ Actions ▶ Time

Generally, a foil describes the presence or absence of an event, or a deviation from the norm [117, 156]. In the ML paradigm, the outputs of an ML model depend on the type of ML that was applied to the problem. Thus, we list potential contrast classes for each of the types of ML:

- ▶ In *(semi-)supervised learning* the output is either a *class* (classification) or a *number* (regression analysis):
 - For classification the contrast class consists of all output classes (class labels). Note that

¹ We refer the reader to Roese and Olson [154] or Miller [123] for a concise overview.

in binary classification (i.e., classifying two classes) the foil is by definition the other class. If the output is not expected to be class 1, the output can only be class 2.

- In regression analysis the output is not discrete. Thus, the contrast class are all other possible values that may be output by the regression analysis—infinately many values. As such, one may compare it to a relative class of values (i.e., *higher* or *lower*) or the deviation from some (statistical) norm—such as a measure of central tendency (e.g., mean, median or mode).
- For *unsupervised learning* outputs commonly do not have descriptive outputs. When *clustering*, the contrast class may be all clusters—i.e., one might ask ‘Why is this case put into cluster 1 instead of cluster 2?’ Unsupervised *anomaly detection* distinguishes two classes: anomaly or not anomaly. Like binary classification, the foil given an output is the other class.
- The output of *reinforcement learning* (RL) are actions resulting in states. However, simply designating the contrast class as alternative actions or states disregards two properties of RL: (i) not all actions may be possible in a given state, and (ii) RL also has a time dimension. Depending on the informativeness, the foil may be made explicit by either the action, the state or both. For instance, in the robot example one might ask ‘Why did the robot go forward instead of turning left and crossing the finish?’. Here, turning left is an action, while crossing the finish is a state. Regarding the time dimension, in his work on contrastive explanation Lipton [108, p. 694] noted as an example “suppose we explain why a bomb went off at noon rather than in the evening”—a clear case of when a time dimension may be included as a foil.

Foil level of detail. In the aforementioned foils we already briefly touched upon the possibility that foils can take on an abstracted form. Foils may have an inherent hierarchy that is explicitly known or one that can be constructed.

For *continuous contrast classes*—e.g., the output of regression analysis or the time dimension in RL—the contrast may refer to a specific value (such as a different value or a norm), or we may group values by discretizing them into groups (such as ‘more than’, ‘less than’, ‘earlier’, ‘first quantile’ or ‘21-30 year olds’).

For *discrete contrast classes* there may be a known grouping that groups the individual members of the contrast class into categories. Take for instance an action selection mechanism for children to learn to self-manage type I diabetes [131], where actions may be playing a particular quiz question, starting a new dialogue or filling in their diary. Here, when for example the selected action is filling in a diary, the foil of playing a particular quiz question may be too fine. Rather, a more coarse foil—playing a quiz at all—may be more appropriate. Another example is hierarchical clustering, where may compare the actual cluster the observation was put into to any level of cluster, as long as this cluster does not include the actual decision itself.

3.4 Conclusion

We have studied literature from the philosophy of science and psychology, to define contrastive explanation concepts for machine learning (ML). For ML, a *contrastive explanation* is a change in input data resulting in the decision to be changed from *fact* (the actual outcome) to *foil* (the outcome of interest). Any of these changes in input data is referred to as a *counterfactual*. The set of potential facts and foils under consideration is referred to as the contrast class. For ML, this contrast class depends on the type of ML—e.g., it can be class labels for classification tasks, and reachable states, actions and different times for reinforcement learning. Moreover, multiple members of the contrast class can be grouped to form a composite foil.

4 INTERPRETABLE MACHINE LEARNING METHODS

In defining our contrastive explanation approach, we draw from existing iML approaches in scientific literature. We systematically review iML methods, with the aim of finding the one(s) most optimal to form the foundation of our contrastive explanation method. The need for this systematic review is twofold. First, during the writing of this thesis, no comprehensive peer-reviewed overview of iML methods exists that considers all four types of ML. Second, recent iML methods generally do not reflect on previous methods that could be suitable for current use-cases [1]. Therefore, by systematically reviewing iML methods we can find techniques that form a general basis for all types of ML, while considering the full range of iML methods.

Section 4.1 outlines the review process taken for our systematic literature review. Section 4.2 introduces a general taxonomy to subdivide the iML methods found in this review. Section 4.3 overviews the ML-specific representations used to convey the explanations to end-users. Finally, Section 4.4 provides an in-depth overview of the studied iML methods. The findings of the literature review are discussed in Section 4.5.

4.1 Literature Review Method

A systematic literature review (SLR) in software engineering consists of three phases [90]: (1) *planning* the review, (2) *conducting* the review, and (3) *reporting* the review. First, this section introduces the *research questions* and the *review protocol*. Second, we overview the results of the identification, selection and quality assessment of research that is included in the iML method overview.

Research questions

As introduced in Section 1.2, with the SLR we aim to address research question “*What are current approaches to automatically acquire explanations from machine learning systems?*”, addressed by two subquestions that aim to (i) obtain generic insights regarding how approaches reach interpretability in ML, and (ii) transfer these insights into the domain of contrastive explanation for machine learning:

- 1.1 *How do iML methods enhance interpretability in machine learning?*
- 1.2 *How can these methods be used for contrastive explanation in machine learning?*

Search process

Literature studies that summarize topic evidence in software engineering are characterized by their unstructured nature [20]. To avoid confirmation bias and to ensure all relevant tools are considered, we objectively review iML tools.

Typical systematic reviews start off from a from search strings containing keywords that may reveal a number of methods relevant to our study. However, as became apparent in Chapter 2 the field of iML suffers from imprecise terminology. Methods that may not be intended directly for interpretability are used as such, and may therefore be difficult to distinguish based on keywords online. Instead, we opt for identification through various articles that have previously overviewed part of iML literature in their literature study. These articles were extracted by consulting major software engineering search databases—i.e., *SpringerLink*, *ACM*, *IEEEExplore* and *Google Scholar*.

Selection criteria. In our literature review, we use to following criteria to select iML methods:

- ▶ **Machine learning tool.** It is an *automated* technique that focuses on explanations for ML decisions—rather than other fields of computer science.
- ▶ **Developed for interpretability.** It explicitly mentions that the development of the method has a goal of improving *interpretability* (e.g., comprehensibility, providing explanations to end-users, transparency)—or is used as such by subsequent studies.
- ▶ **Not general-purpose ML.** It is not a general-purpose ML technique. Using this criteria we exclude methods such as decision trees, or Lasso regression.
- ▶ **Describes method.** Describes actual explanation extraction method, instead of using an already existing method for explanation (e.g., explanatory debugging [99] and Ravelo [177]).
- ▶ **Novelty.** The work provides a new method. Multiple works may continue on this, extending the method or showcasing the method on new use-cases. However, we aggregate them into a single method where we describe the most up-to-date version.
- ▶ **Scientific.** While we acknowledge that tools exist beyond the scientific body of work (e.g., eli5, Skater and tree-reg), these tools (i) typically implement scientific methods, or (ii) provide novel improvements without argumentation or documentation.

Note that we included articles that have the main purpose of the iML method development, as well as articles where the iML method was only one element of the article.

Included and excluded studies. Our overview resulted in 84 iML methods. iML methods published before April 2018 were included. We report our review according to PRISMA [126]. Section 4.4 expands on each method in detail.

Figure 4.1 shows how the methods in the overview in Section 4.4 were selected. Our initial *identification* consisted of the *general* (non-peer reviewed) iML method overview of Guidotti et al. [64], Chakraborty et al. [25]’s overview of deep neural networks (DNNs) iML methods, and a study of *saliency* map extraction from by Kindermans et al. [89] and Montavon et al. [129]. The

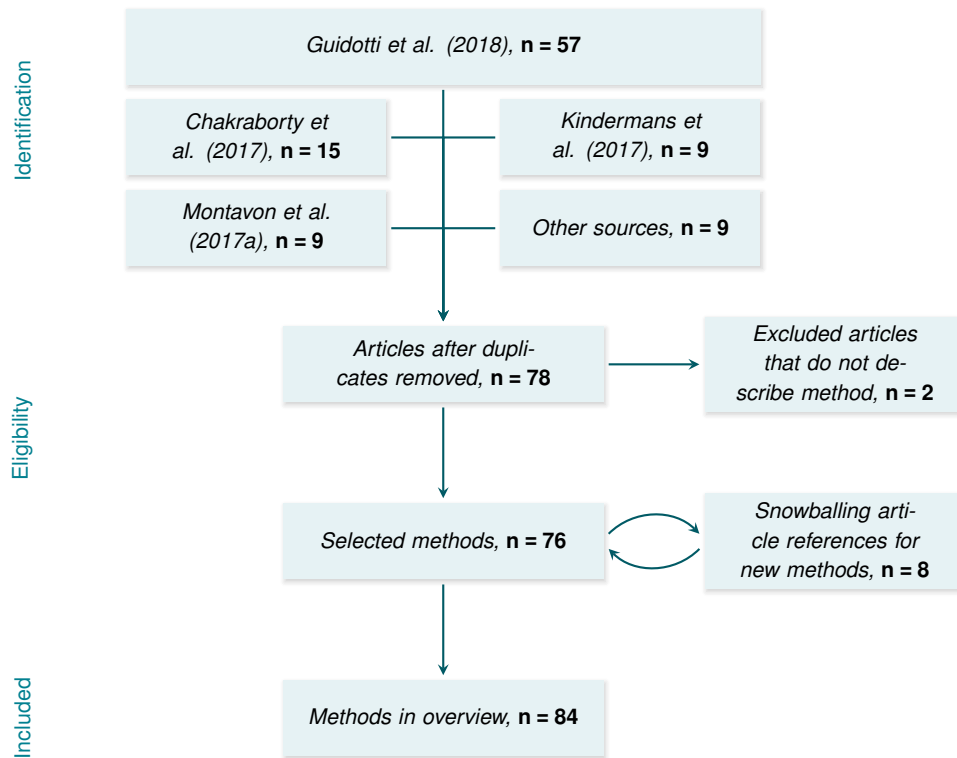


Figure 4.1 PRISMA flow chart describing the literature review of interpretable machine learning (iML) methods

identification phase resulted in 78 methods. From these methods, we excluded Gibbons et al. [60] (single tree extraction without description of method details) and Radford et al. [145] (does not describe method). From the remaining 76 methods we performed snowballing and reverse snowballing—i.e., considering the articles that a method references, and the articles that directly reference the method—to identify another 8 methods after testing for eligibility. Finally, this resulted in the 8 methods in the final overview.

Data collection. The data extracted from each study were as follows:

- ▶ The method name, and source and full reference.
- ▶ Classification of the type of representation used for explanation.
- ▶ The type of ML model and type of data (images, text, and/or tabular data) the method works on.
- ▶ How it can best be categorized according to the taxonomy in Section 4.2: the scope of the explanation (i.e., ‘Does it explain the whole model or only part of it?’), and its approach taken.
- ▶ The type(s) of ML it is able to provide explanations for.

- ▶ How the method was evaluated, and whether this included empirical evaluation with human subjects.
- ▶ Available source code and implementation language.
- ▶ Summary of the study, including a high-level description of the method.

4.2 Taxonomy of Interpretable Machine Learning Methods

Methods for tackling iML can broadly be categorized in two ways [46, 64, 110]. First, whether the original model is designed as *transparent*, is first learned and then *decomposed* into an iML model, or treated as a *black-box* after learning (and queried *post-hoc*). Second, whether the explanation is for a single instance (*subject-centric* or *local*) or for the entire model (*model-centric* or *global*). Our taxonomy adopts subdivisions made by Edwards and Veale [46] and Lipton [110], and categorizes iML approaches along two dimensions: the *approach* taken and the *scope* of the explanation. Figure 4.2 shows the taxonomy of ML methods along these dimensions, with the number of methods for each approach-scope pair. Note that five methods (one decompositional, four pedagogical) are able to provide local as well as global explanations. Below, for each of the dimensions we provide a brief consideration of the type of iML methods covered by this dimension.

		Approach		
		Transparent	Decompositional	Pedagogical
Scope	Global	n = 14	n = 16	n = 33
	Local	n = 2	n = 14	n = 10

Figure 4.2 Taxonomy of methods for interpretable machine learning (iML)

Explanation scope

The explanation scope is either *global* or *local*.

Global. Global approaches aim to explain the complete model. Strictly speaking, they require an explanation in which the explainee is able to comprehend an aspect of the entire model at once [110]. Edwards and Veale [46] argue that these aspects may include the *setup information* (e.g., type of model used, its goal and parameter settings), *training metadata* (e.g., summary statistics on the data, description of the training process), *performance metrics* it acquired in training, and a *simplified explanation of the model* (simplified human-understandable accounts of how inputs are turned into outputs). The latter is the sort of explanation we focus on in transparent design, and decompositional or pedagogical explanation.

Global models also include *surrogate* (or *mimic*) models [e.g., 10, 19]. Surrogate models are smaller,

simpler—and thereby arguably more interpretable—versions of a complex model. They explain the behavior of the underlying model, while also focusing on increased comprehensibility. Even if it is possible to condense a complex model into a more simple one, these models have as a downside that they commonly have to give into predictive performance to be more intelligible (interpretability-accuracy trade-off) and there might be a gap in functionality with the model they are attempting to mimic [19, 87].

Local. Unlike global explanations, local explanations only seek to explain a single decision. They show how the model arrived at the decision made. Local explanations only require an explanation of the neighborhood around the data point it predicted, and can therefore sometimes disregard large parts of the model in their explanation [46, 148]. Edwards and Veale [46, p. 27] note that even though these explanations are bound to a specific decision made regarding a subject, this does not restrict them to only being performed after a decision was made: “they are theoretically possible to give before or after a “decision” [...] if access to the model is provided.” Even though they may only give an explanation for a single decision, by combining local explanations explainees can build their own *mental model* of the ML model—an internal representations of how the system performs its decisions [46, 123].

In local explanation, explanation evaluation may also be more feasible—i.e., deciding whether the decision made was made using reasonable assumptions. Explainees are known to be able to explain single examples really well, even when using implicit knowledge [157]. Moreover, it aligns more with human-scale reasoning and semantic interpretation—and can therefore circumvent the issues stemming from the mathematical complexity and multi-dimensionality of the ML model [21].

Approach

We distinguish three approaches to make create interpretable ML explanations: *transparent*, *decompositional* and *pedagogical*. Figure 4.3 illustrates the difference between the three approaches: transparent iML methods create an interpretable model by design, decompositional methods access the underlying model mechanics for an explanation, while pedagogical methods treat the underlying model like a black-box. Below, we expand on each approach and state their respective advantages.

Transparent. In the strictest sense, we might call a model transparent if a person can contemplate the entire model at once [110]. This concept—referred to as *simulatability* [110]—is heavily related to quantifiable proxies in evaluating interpretability as mentioned earlier (see Section 2.4). By picking inherently more intelligible model classes (such as decision tables, decision trees, rule-based systems or linear models), and limiting their size in some way, an explanation can be conveyed to a user. Even when the model class is generally not perceived as intelligible—such as neural networks—, restricting models by picking a small model size can result in an explanation that is comprehensible for humans [110]. Transparent design is sometimes also referred to as *white-box* ML.

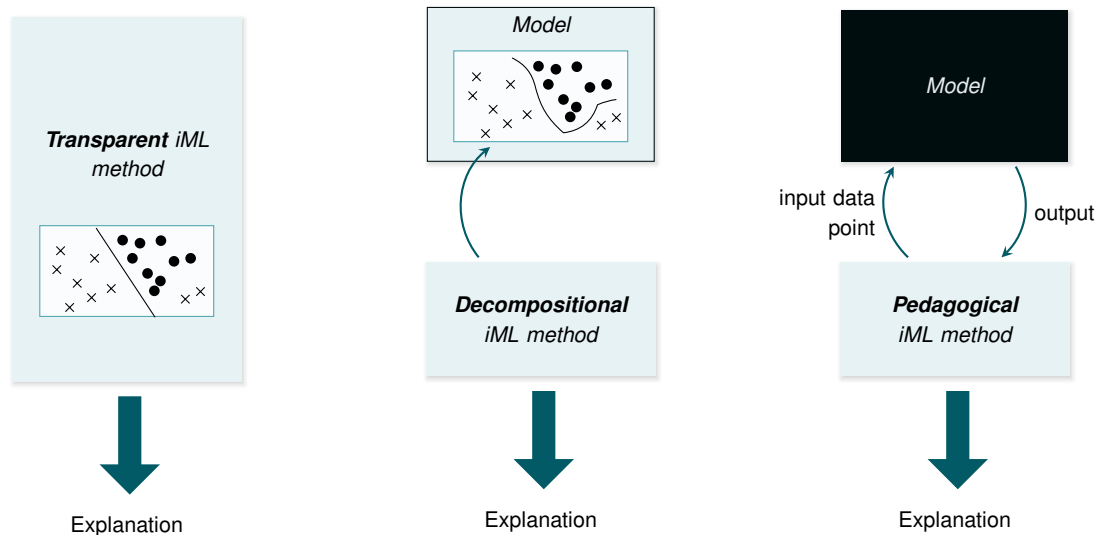


Figure 4.3 Transparent, decompositional and pedagogical interpretable ML (iML) methods

While such designs provide interpretability through transparency, they usually have to give into accuracy as they are unable to capture the intricate patterns of the input data [185]. This is also demonstrated in Figure 4.3, where the simply linear model classifier of the transparent iML method cannot fully capture all data points perfectly. Therefore, part of their design goal is to optimize model performance to similar levels to that of state-of-the-art ML techniques in that problem area.

Observe that in transparent design one directly creates a model with the aim of interpretability from data, and therefore does not have an underlying black-box model it aims to explain.

Decompositional. Transparency is achieved through *decomposability* [110]: each input, parameter, output and calculation might have an intuitive explanation. By using these explanations, a full explanation can be formed for an explainee. This does not only hold for already transparent models, but these partial explanations can often be extracted from underlying models as well. Some ML model classes are already decomposable by design (e.g., linear models), while other models may be made decomposable (e.g., the variable importance in Random Forests), or are attempted to be decomposed (e.g., deep neural networks) [46]. The downside of decompositional methods is that they are *model-dependent*—i.e., for each (implementation of a) class of models they require a specific approach. However, with access to the model, decomposable iML methods have the advantage of (i) being able to re-use intelligible parts of the model for the explanation, and (ii) having a high fidelity to the underlying model.

Pedagogical. Rather than decomposing the underlying model, pedagogical methods query the model like an oracle—providing insights based on the model inputs and corresponding outputs [5, 46]. Typically, they use the underlying model to generate examples, from which an interpretable representation is extracted. They treat the model like a *black-box* and decouple the

explanations from the decision-making process.

Pedagogical methods present a number of benefits. First, these methods are *model-agnostic*, in that they do not rely on creating specific interpretability methods for each (sub)class of model [46]. Instead, they can be applied to any ML problem within the same type of ML (e.g., classification or reinforcement learning). Hence, as long as they fall within the same ML type methods allow for the comparison of underlying models, and will even work if in the future new algorithms for creating models are introduced. They are commonly employed after automated decision has taken place (then referred to as *post-hoc* explanations), allowing explainees to build their own model of the complex underlying model [194]. Second, legally they pose the benefit that they *do not require for company models to rely on providing insights into trade secrets or intellectual property* [46]. Accordingly, these methods may manifest in widespread adoption due to their potential to mitigate the issue of interpretability without requiring to tackle opacity stemming from intentional corporate or state secrecy [see 21]. Third, by decoupling explanation from decision-making, these methods *do not sacrifice the predictive performance* of models [110]; one of the main goals of the XAI programme [65].

Pedagogical model interpretability usually requires multiple queries of the model, and may therefore be an relatively expensive means to acquire explanations. As acquiring explanations is not free, their utility must be balanced against the cost of generating them [44]. However, similar to explanation in daily life not all decisions require an explanation. Explanations are usually not required to work in real-time [86]. Especially in contrastive explanation, where explanations are sought when the outcome deviates from the expected outcome, post-hoc interpretability is arguably the most suitable paradigm similar to human-style explanation.

4.3 Explanatory Representations

What all these methods have in common is that they use the same set of representations to provide the explanation to the explainee. Section 2.5 introduced the four main representations to convey automated explanations: *textual*, *graphical*, *saliency* and *explanation by example*. In this section, we detail ML-specific methods that provide explanations through (a combination of) these explanations. These representations are based on *recognized models* for interpretation [51, 79], such as decision rules, decision trees and finite-state automata.¹ Note that while simply choosing such a representation does not consequently make the explanation interpretable (interpretability also depends on the semantics of the models, i.e. the contents). However, evaluating representations before selecting the iML method has the particular advantage of being able to consider the interpretability requirements of the intended explainees before picking a specific method [11].

Decision trees and rule-based models

Decision trees and rule-based models are natural representations of prediction problems for humans [80, 158]. They allow humans to perform a sequence of tests, in order to determine the

¹ Appendix B expands on the various basic machine learning (ML) techniques in more detail, including *decision trees*, *decision rules*, *regression models* and *neural networks*.

corresponding output for a given input. They are used for *why* (not) explanations—i.e., why a decision was made—, as well as *counterfactual* (what if) explanations [107]. We subdivide them into four categories:

- ▶ **Decision trees (DT)** subdivide the input feature space into mutually exclusive decisions regions in their leaf nodes. Explainees can decide on an outcome by performing a sequence of decisions from the root node to a leaf. Their interpretability is facilitated three factors [51]. Firstly, its graphical representation allows for an easy overview of larger models. Secondly, features considered more important in a decision are shown further to the top of the tree—showing the relative importance of features in a decision. Thirdly, they only show a subset of relevant features important in distinguishing between decisions. However, even though a feature may occur higher up in the tree, this does not necessarily mean it is used to distinguish between as many instances as features further down the decision path. Moreover, decisions can be replicated multiple times, introducing irrelevant attributes that do not affect the final decision.
- ▶ **Decision Rules (DR)** have a less restrictive nature over decision trees. This poses the benefit that they do not enforce mutually exclusive rules [79]—allowing simpler and shorter rules that end up in the same decision. Decision rules take on a multitude of forms, such as *decision sets*, *decision lists* and *M-of-N rules*. Figure 4.4 illustrates the difference between decision sets, and M-of-N rules.
 - **Decision Lists (DL)** (*if-then-else* rules) are a sequence of boolean literals, where if the literal evaluates to true the corresponding consequent is followed, otherwise the *else* path is followed [151]. They are easy to interpret locally, and allow for non-mutually exclusive and non-exhaustive decisions [51]. However, they do not give a direct clue about feature importance and are typically difficult to interpret as a whole, due to their typical length complexity as a whole [51]. Graphically, decision lists may also be represented as a *decision table* or *decision tree* [79].
 - **Decision Sets (DS)** (*if-then* rules) tackle the difficulty in interpretation as a whole of decision lists by simplifying rules to *if-then* statements [100, 152]. If all antecedents in the statement hold, the consequent is true. For example, in classification each class only has to be represented by a single rule.
 - **M-of-N rules** mitigate the length issue of decision lists, by using a set of simpler antecedents where M of the N antecedents must hold to result in a given consequent [186]. Due to their simpler nature, they typically do not require any conjunctions (AND) or disjunctions (OR) in their antecedents.
- ▶ **Scoring Systems (SS)** are similar to M-of-N rules in the sense that an explainee needs to evaluate multiple statements to arrive at a decision. Instead of simply counting the number of true instances, a point weight is attached to each statement—demonstrating relative feature importance—, where if the summed weight is above a threshold a certain decision holds. They are a popular classification system used for predictions in healthcare [190]. Figure 4.4 also shows an example scoring system.

Decision Set	M-of-N Rule	Scoring System										
<p>if BMI \geq 25 and sex = male and diabetes or hypertension then <i>Stroke</i></p> <p>if cholesterol < 194 and exercise/wk \geq 2 then <i>No</i> <i>Stroke</i></p>	<p><i>Stroke</i> if at least 2 of 4 rules below are true:</p> <ul style="list-style-type: none"> · sex = male and diabetes · BMI \geq 30 · hypertension · cholesterol \geq 240 	<p><i>Stroke</i> if total points \geq 5</p> <table border="0"> <tr> <td>age \geq 65</td> <td>+4 points</td> </tr> <tr> <td>BMI \geq 27</td> <td>+3 points</td> </tr> <tr> <td>cholesterol \geq 205</td> <td>+1 points</td> </tr> <tr> <td>exercise/wk < 2</td> <td>+1 points</td> </tr> <tr> <td>sex = female</td> <td>-3 points</td> </tr> </table>	age \geq 65	+4 points	BMI \geq 27	+3 points	cholesterol \geq 205	+1 points	exercise/wk < 2	+1 points	sex = female	-3 points
age \geq 65	+4 points											
BMI \geq 27	+3 points											
cholesterol \geq 205	+1 points											
exercise/wk < 2	+1 points											
sex = female	-3 points											

Figure 4.4 Example rules for decision sets, M-of-N rules and scoring systems for stroke risk prediction

Regression models

Regression models apply a function to each coefficient (input feature) to determine an outcome. Their interpretable counterparts are ones that reduce to a small number of features, or ones that use simple functions (e.g., a single weight) to each coefficient.

- ▶ **(Log)linear Models (LI)** help explainees understanding the influence of input features with respect to others by comparing their coefficients [189]. They attach a single weight to each coefficient. When normalized—i.e. made independent of unit—, they can show the relative feature importance (especially for linear and logistic models) by looking at the magnitude and sign of coefficients. Note that various regression techniques (e.g., *Lasso* and *Ridge*)² allow for increased interpretability by selecting a smaller relevant subset of features.
- ▶ **Partial Dependence Plots (PDP)** highlight the average partial relationship between a set of features (typically 1-3) and the outcome [53]. For each feature, it shows graphically how varying that feature influences the outcome, while averaging out the effect of other features.

Feature importance and saliency maps

These representations provide insights into a model by showing the relative contribution of features to an individual decision or model decision (e.g., class). These representations can broadly be categorized in two types:

- ▶ **Feature Importance (FI)** aims to acquire a ranked ordering of features' contributions to an individual outcome or global model decisions. Each feature has a positive or negative contribution to an outcome, where typically a subset of the highest contributions (e.g., top ten) are shown. To convey a ranked list of feature contributions, they require a meaningful label (e.g., name of feature, or word in text).

² Discussed in Appendix B

► **Saliency** methods (*attention maps*) show the importance of *individual outcomes* as an *overlay* on the original input. They are a popular means of visualizing relative importance in domains with a large number of features—such as *images* or *text*. While all methods have the same purpose, they vary in how they acquire the saliency of an input. These methods can acquire saliency in a *bottom-up* (i.e., changing inputs and observing changes in outputs) or *top-down* (i.e., changing the outputs and discovering how this affects the inputs) manner. We distinguish three types of saliency methods, each with their respective benefits and downsides [89]:

- **Sensitivity Analysis (SA)** (*gradients*) are a *bottom-up* saliency method that shows how small changes in inputs result in a different model output. These methods pose several benefits. Firstly, these methods only require only changes in inputs and observing outputs and are therefore *model-agnostic* (pedagogical) and *implementation invariant* (i.e., as long as the behavior of model $\hat{f}(\cdot)$ is the same they result in the same saliency regardless of architecture). In addition, they satisfy *input invariance* (a shift in inputs—e.g., making all pixels 10 points darker—results in the same saliency map). The downside of these methods is that you can only see the saliency of the input (typically coarse-grained), instead of introspecting the model’s individual layers. As a result, they might be less useful for model debugging.
- **Signal Methods (SM)** backpropagate a signal *top-down* through a (deep) NN to isolate input patterns responsible for neuron activations in the final layers of a NN. This enables a user to observe the behavior of individual layers. However, it has to give into implementation invariance in order to do so. SMs benefit from being input invariant.
- **Attribution Methods (AM)** decompose the feature relevances (determined by their respective weights) into the relevance of areas in the input layer. Their downside is that they are *not input invariant*, because they typically use a baseline reference point for determining saliency. The choice of this baseline is a hyperparameter set by the user, and affects the resulting saliency map—thereby being able to manipulate the resulting saliency map. Nevertheless, like SMs these methods allow for the introspection of individual NN layers.

Prototypes

Prototypes are example (partial) inputs that best summarize the data set or a model area (e.g., a class). They can either be *selected* from a set of (provided or generated) examples, or they are directly *reconstructed* from a model.

- **Prototype Selection (PS)** selects a small set of examples from the data set that best summarize that data set [13]. These methods are often aimed at describing neighborhoods of data, such as a particular class (classification) or cluster (unsupervised learning). In particular, they look for a minimal number of prototypes that best cover all data points in their neighborhood (in terms of inputs and outputs).



Figure 4.5 Example prototype reconstruction for images, adapted from Vedaldi [193]

- Prototype Reconstruction (PR)** seeks to find the most representative example of a class in a *decompositional* manner. Typically, this method is applied to DNNs for image classification tasks in a process called *activation maximization*. Activation maximization synthesizes the preferred stimuli for a neuron in the NN by starting from a random image and iteratively calculating through backpropagation how the input pixels should be changed to increase the activation of that neuron [132]. Neuron activations can be combined to introspect the models' prototypes at a layer level. Some authors use *natural image priors* (e.g., example images used in training), because activation maximization methods typically result in an optimization that does not resemble a natural image. Figure 4.5 illustrates prototype reconstruction for the 'dog' class for an image classifier, and illustrates how this explanation is improved using a natural image prior. The downside of these models is that even for moderately complex models, a good global approximation cannot generally be found [15].

Annotations

Humans oftentimes provide explanations and justification verbally or through text. To mimic this behavior, recent approaches have proposed methods for creating text explanations for decisions made by models.

- Annotation (AN)** provide textual explanations for a black-box model regardless of the type of input (i.e., images, text or tabular). To do so, they incorporate explanation generation in the training process of the model, resulting in an *explanatory model* in addition to the prediction model. When applying the model to a task, these methods can then annotate the outcomes with text explanations generated by the explanatory part of the model.

4.4 Methods

In our systematic review, we found 84 distinct iML methods. In this section, for each method we briefly detail the main method idea and contributions. Table 4.2 shows the iML methods considered, the year they were published, the explanation scope and approach, the explanatory representation used and to which types of ML the method applies. In addition, we included whether human evaluation was performed, and whether and in which language the source code

is available. Table 4.1 expands on the column meanings and the used abbreviations in columns.

Table 4.1 Legend of columns for Table 4.2

Column	Description	Abbreviations
Name	Method name	
Year	Year published	
Scope	Explanation scope	Global, Local
Appr.	Explanation approach	Decompositional, Pedagogical, Transparent
Repr.	Representation	Decision Rules, Decision Tree, Feature Importance, Prototype Selection, Prototype Reconstruction, Saliency (Sensitivity Analysis, Signal Methods, Attribution Methods), LInear model, ANnotation, * other
ML Type	Type of ML	Unsupervised, Supervised (Classification, Regression), Reinforcement Learning
H. Eval.	Human evaluation	
Code	Source code availability in this language	

Table 4.2 Overview of the 84 iML methods studied in the literature review

Name	Ref.	Year	Scope	Appr.	Rep.	ML Type	H.Eval.	Code
SVM+P	[133]	2002	G	D	DR	C		
ExtractRules	[56]	2005	G	D	DR	C		
GRG	[134]	2008	G	D	DR	C		
RxREN	[6]	2012	G	D	DR	C		
<i>Hara et al.</i>	[69]	2016	G	D	DR	C,R		
Tree Metrics	[26]	1998	G	D	DT	C		
CDT	[162]	2007	G	D	DT	C		
TSP	[179]	2016	G	D	DT	C		
TreeView	[181]	2016	G	D	DT	C,R		
GENESIM	[192]	2016	G	D	DT	C		Python
NID	[135]	2002	G	D	FI	C		
<i>Karpathy et al.</i>	[85]	2016	G	D	FI	U,C		Torch (Lua)
<i>Simonyan et al.</i>	[168]	2013	G	D	PR	C		
DeepVis	[204]	2015	G	D	PR	C		Python
<i>Nguyen et al.</i>	[132]	2016	G	D	PR	C		Python
<i>Zahavy et al.</i>	[205]	2016	G	P	*	RL		
<i>Schwartz-Ziv et al.</i>	[166]	2017	G	P	*	C		
<i>Craven et al.</i>	[31]	1994	G	P	DR	C		
REFNE	[210]	2003	G	P	DR	C		
G-REX	[83]	2004	G	P	DR	C,R		
ExOpaque	[66]	2007	G	P	DR	C,R		
<i>Johansson et al.</i>	[82]	2009	G	P	DR	C,R		
STEL	[38]	2014	G	P	DR	C,R		R
GPRL	[70]	2017	G	P	DR	RL		
MAGIX	[144]	2017	G	P	DR	C		
Trepan	[32]	1996	G	P	DT	C		
CMM	[41]	1998	G	P	DT	C		
<i>Krishnan et al.</i>	[96]	1999	G	P	DT	C		
DecText	[16]	2002	G	P	DT	C		
<i>Sánchez et al.</i>	[161]	2015	G	P	DT	U		
STA	[209]	2016	G	P	DT	C		
<i>Bastani et al.</i>	[10]	2017	G	P	DT	C,RL	✓	
PALM	[97]	2017	G	P	DT	C		
<i>Strobl et al.</i>	[172]	2008	G	P	FI	C		
GA ² M	[113]	2013	G	P	FI	C,R		Java
GoldenEye	[73]	2014	G	P	FI	C		R
OPIA	[2]	2015	G	P	FI	C,R		Python
VIN	[78]	2004	G	P	PDP	C,R		
ICE	[62]	2015	G	P	PDP	C,R		R
Prospector	[94]	2016	G	P	PDP	C	✓	Python
GFA	[3]	2018	G	P	PDP	C		Python
PS	[13]	2011	G	P	PS	U,C		
<i>Baehrens et al.</i>	[8]	2010	G	P	SA	C		
GSA	[29]	2011	G	P	SA	C,R		

Name	Ref.	Year	Scope	Appr.	Rep.	ML Type	H.Eval.	Code
Rationalization	[47]	2017	G	T	AN	RL	✓	
CMAR	[106]	2001	G	T	DR	C		
CPAR	[203]	2003	G	T	DR	C		
RuleFit	[54]	2008	G	T	DR	C,R		Python
BRL	[104]	2015	G	T	DR	C,R		Python
TLBR	[174]	2015	G	T	DR	C		
FRL	[195]	2015	G	T	DR	C		Python
IDS	[100]	2016	G	T	DR	C	✓	
1Rule	[116]	2017	G	T	DR	C		
Bayesian Rule Set	[197]	2017	G	T	DR	C		Python
DILSVM	[23]	2016	G	T	DR,LI	C		
SLIM	[190]	2016	G	T	DR,LI	C		
OT-SpAM	[196]	2015	G	T	DT	C,R		
BCM	[88]	2015	G	T	PS	U,C	✓	
<i>Tzeng et al.</i>	[188]	2005	G,L	D	FI	C		
QII	[36]	2016	G,L	P	FI	C		
SHAP	[114]	2016	G,L	P	FI	C,R		Python
LIME	[148]	2016	G,L	P	FI	C,R	✓	Python
Streak	[48]	2017	G,L	P	FI	C,R		Python
Anchor	[149]	2018	G,L	P	DR,PS	C		Python
LRP	[7]	2015	L	D	AM	C		
DTD	[128]	2017	L	D	AM	C		
IG	[175]	2017	L	D	AM	C		Python
Excit. Backprop	[207]	2017	L	D	AM	C		Python
FDS	[105]	2015	L	D	FI	C		
NeuralTalk	[201]	2015	L	D	FI	C		Torch (Lua)
LSTMVis	[171]	2018	L	D	FI	U,C		Python
<i>Mahendran et al.</i>	[115]	2015	L	D	PR	C		Matlab
DeConvNet	[206]	2014	L	D	SM	C		
GB	[170]	2015	L	D	SM	C		
CAM	[208]	2015	L	D	SM	C		Python
Grad-CAM	[163]	2016	L	D	SM	C		Torch (Lua)
DeepLIFT	[164]	2017	L	D	SM	C		Python
MES	[187]	2016	L	P	DR	C		
<i>Tolomei et al.</i>	[185]	2017	L	P	DR	C		
<i>Strumbelj et al.</i>	[173]	2010	L	P	FI	C		
SEDC	[119]	2014	L	P	FI	C		
<i>Fong et al.</i>	[50]	2017	L	P	SA	C		
Vis. Expl. Mod.	[72]	2016	L	T	AN	C		
<i>Lei et al.</i>	[103]	2016	L	T	FI	C		Python

Transparent design

Transparent local or global iML methods primarily focus on tabular data. Especially data sets with a large number of features are difficult to explain, when these features have intricate interactions. Instead, transparently designed methods oftentimes disregard some of the complexity of the underlying data to explain with a small number of features and interactions between these features. Because of their human-style interpretation, popular explanatory representations are decision lists, decision sets and linear models. For more complex problems, explanations can be incorporated in training or provided through examples. In the following paragraphs we describe the iML methods per explanatory representation.

Decision lists. For supervised learning tasks, RuleFit³ [54] derives a small number of rules that have a predictive accuracy comparable to tree ensemble methods. The simple rules it generates comprise a conjunction of a few statements describing decision paths in an interpretable manner. Additionally, RuleFit is able to show the most important rules and features for a given decision. Motivated by the need for interpretable predictive medical models for deciding on patient diagnosis, Letham et al. [104] introduce Bayesian Rule Lists (BRL)⁴—decision list classifiers that describe the feature space in accurate, sparse decision statements. By using permutations of pre-mined rules, BRL is able to greatly reduce the rule search space and consequently scales very well with a large number of features. Continuing on the work of Letham et al. [104], Falling Rule Lists (FRL)⁵ [195] are also constructed from pre-mined rules and aimed at healthcare. FRL are ordered decision lists that enforce monotonicity in their ordering to ensure that the topmost rules have the highest confidence in classification. Su et al. [174] create sparse decision lists—so-called Two-Level Boolean Rules (TLBR)—that form classification predictions by connecting features with logical statements AND, OR and NOT in simple rules. They propose two strategies for finding TLBR: one based on 0-1 classification error and another based on the Hamming distance from the current rule to the closest rule that correctly classifies a sample. While also optimizing for accuracy, both strategies simultaneously try to minimize the number of features used in rules to enforce sparsity.

Decision sets. However, decision lists still have drawbacks. The chaining *if-then-else* statements in decision lists create increasingly smaller partitions of the feature space, that become less and less interpretable the longer the statements become. To this end, Lakkaraju et al. [100] propose Interpretable Decision Sets (IDS)—*if-then* rules that can be considered in any order. For each class, a single rule is constructed connected with AND conjunctions. With one rule per class, decision sets remain interpretable even for multi-class classification with a larger number of classes. Similarly, 1Rule [116] learns a single decision rule (either with AND/OR conjunctions, or M-of-N rules) that accurately classifies an entire class—and are therefore very interpretable. Through linear programming, 1Rule obtains decision sets containing boolean statements that balance between interpretability and accuracy. Bayesian Rule Set⁶ [197] produces sparse classification de-

³ <https://github.com/christophM/rulefit>

⁴ https://users.cs.duke.edu/~cynthia/code/BRL_supplement_code.zip

⁵ https://users.cs.duke.edu/~cynthia/code/falling_rule_list.zip

⁶ <https://github.com/wangtongada/B0A>

cision sets that connect features with AND/OR conjunctions. Their method is based on Bayesian probabilities, that aim to maximize the likelihood that all instances are classified correctly.

The finding that decision lists can become difficult to comprehend in practice is not new. Rather than using a single high-confidence classification rule, Classification based on Multiple Association Rules (CMAR) [106] determines the class label by a set of rules. CMAR generates rules using association mining technique FP-growth, and selects a small number of rules that accurately classify with high confidence. Yin and Han [203] greatly improve on the running time of CMAR with their method Classification based on Predictive Association Rules (CPAR), that uses dynamic programming to generate a smaller set of rules with higher quality, lower redundancy and similar predictive performance.

Linear models. Ustun and Rudin [190] introduce Supersparse Linear Integer Models (SLIM): linear binary classification models that only contain a few terms that are subtracted, added and multiplied. To increase comprehensibility, these boolean terms only have an integer number of points attached. By adding up the number of points for all terms that hold on a decision, if the number of points is greater than or equal to the threshold for a given class the data point falls within that class. SLIM can be used by domain experts to easily classify manually—especially within healthcare.

Discrete Level Support Vector Machine (DILSVM) [23] is a linear SVM variety where each feature weight is represented as a *Likert scale*. DILSVM selects a small subset of features with non-zero weights, and shows their weights as a discretized ordinal rating scale (e.g., *strongly disagree, disagree, neutral, agree, strongly agree*)—allowing for the extraction of feature importance and manual analysis for classification.

Incorporating explanation in training. Recent approaches have considered explanation generation as an integral part of training the (black-box) model. Lei et al. [103]⁷ use an end-to-end encoder and generator network that can create interpretable summaries that can accurately predict text. These summaries, form coherent stretches of the input text (e.g., a phrase) that agrees with the overall classification (e.g., sentiment) for that text. The encoder and decoder networks are jointly trained to favor accurate, yet concise rationales. The Visual Explanation Model [72] creates class-specific text explanations for image classification. To generate explanations, it combines two key elements. First, an *image description* explaining the discriminative features present in the image to be explained (e.g., wheels and a trailer). Second, the *class definition* of the class outcome of the classifier (e.g., a truck is a tall vehicle with four or more wheels, a trailer). By combining the elements present in the image that are also part of the class description, their method can provide a *visual explanation*—combining the predicted feature with an explanation generated using an LSTM natural language generator for explanations. Rationalization [47] focuses on reinforcement learning (RL) use-cases where explanation is desired in a different format from the input data, such as textual explanations for images—similar to human-style explanations. They manually annotate inputs with explanations and use an LSTM recurrent neural network (RNN)

⁷ <https://github.com/taolei87/rcnn/tree/master/code/rationale>

to learn these representations alongside the task. They then use an RNN decoder to generate corresponding output rationales while performing the task.

Other approaches. To create small interpretable decision trees, Wang et al. [196] designed Oblique Treed Sparse Additive Models (OT-SpAMs). OT-SpAMs are made up of *region specific* decision trees, where the tree first splits up the feature space in small disjoint regions that each are predicted with their own comprehensible oblique supervised predictor. As comprehensible predictor, OT-SpAMs use *sparse additive models* [146], that form an interpretable complex model by adding linear models with very few terms per model. By increasing rule complexity per node, the size of the tree can be restricted to a very small number of decisions while still performing with high accuracy.

For unsupervised and supervised ML tasks, Kim et al. [88] propose the Bayesian Case Model (BCM). It builds on case-based reasoning, the idea that new situations can be addressed by experiences with previous examples. BCM extracts subclusters using unsupervised learning, where each subcluster is illustrated using *prototypes* (the most descriptive examples) and *subspaces* (features and values best describing the subcluster). For example, for a data set on recipes, BCM can discern different types of cuisines (e.g., TexMex cuisine, Mediterranean cuisine) and the key ingredients re-occurring within these cuisines.

Decompositional local explanation

A great deal of literature focuses on creating saliency maps for deep neural networks (DNNs), with an emphasis on *convolutional* NNs (CNNs) and *recurrent* NNs (RNNs). Recall that saliency methods broadly fall into three categories: *sensitivity analysis* (gradients), *signal methods* and *attribution methods* [89]. While sensitivity analysis only considers changes in input—and therefore is pedagogical—, signal methods and attribution methods decompose the underlying method through backpropagating the activation signal of the final layers of the NN back to its inputs.

Feed-forward signal methods. Signal methods reveal input stimuli of a layer by inverting the data flow of a feed-forward NN, passing on a signal back to the input space. This method was first proposed by Zeiler and Fergus [206]. For supervised image classification, DeConvNet map the feature activation in any layer of a CNN using deconvolutional approximations back into the input image—showing where the attention lies within that image. Guided Backprop (GB) [170] improves on DeConvNet by adding an additional guidance signal during backpropagation that combines deconvolutions with standard backpropagation, allowing for saliency maps created from lower layers and more accurate ones from higher layers. As a more coarse approach, Class Activation Mapping (CAM)⁸ [208] takes the weights and activations (reconstructed through *global average pooling*) of the convolutional layers before classification to create the highlight the discriminative regions for that class on the input image. The downside of this approach is that it cannot reconstruct local explanations for an output label if the model contains fully-connected layers. To circumvent this, Gradient-weighted CAM (Grad-CAM)⁹ [163] extends CAM by only tak-

⁸ <https://github.com/metalbubble/CAM>

⁹ <https://github.com/ramprs/grad-cam>

ing into consideration the final layers of the CNN before the output. Deep Learning Important FeaTures (DeepLIFT)¹⁰ [164] are more widely applicable than the aforementioned methods—not just CNN-specific. Using their recursive method they explain the difference in output based on the difference between a reference input and actual input. DeepLIFT overcomes two issues causing misleading importance scores: (i) even with a zero gradient the input can be important, and (ii) discontinuous gradients can cause sudden jumps in feature importance over tiny differences in inputs.

Feed-forward attribution methods. Attribution methods decompose a neuron at the output layer (rather than the layers before that) into contributions in the input layer. Layer-wise Relevance Propagation (LRP) [7] backpropagates a class-specific signal through a NN while multiplying it with each convolutional layer’s activations. This results in a fine-grained heatmap of the most important features (input pixels) for classification. Deep-Taylor Decomposition (DTD) [128] determines the saliency relative to a reference neuron. It obtains the relevance score for each neuron in the forward pass, and backpropagates a signal top-down from to obtain the pixel-wise relevance scores. Sundararajan et al. [175] propose Integrated Gradients (IG)¹¹—a method for attributing the prediction of a DNN to its input features, that requires no modification to the original method. Moreover, it has two desirable additional properties: (i) it produces the same result as long as the inputs and outputs are equal—regardless of NN architecture, which is unlike LRP and DeepLIFT—, and (ii) it guarantees that if a feature change causes a different prediction it shows up in the sensitivity analysis. Excitation Backprop¹² [207] uses the contrast between a backpropagated attribution for class and not-class to determine the importance for a class decision. In addition to CNNs, this method can also do this for individual words in captions generated by a RNN.

Recurrent neural networks. NeuralTalk¹³ [201] can generate captions for image using visual attention—restricting the focus on part of the input to produce more accurate results—, and show for each word in the generated caption where the the attention on the input image was for that word. For text inputs, First-Derivative Saliency (FDS) [105] visualizes how each unit in a RNN contributes to a decision by approximating the contribution of word embeddings using a linear function. LSTMVis¹⁴ [171] allows for LSTM RNN architecture comparison and debugging by visualization of the cells’ hidden states.

Prototype reconstruction. Mahendran and Vedaldi [115]¹⁵ provides insight into a CNN by showing *prototype reconstructions* of a reference image that are indistinguishable to a CNN. Their method aims to find a set of image perturbations of an image to explanation, that result in the

¹⁰ <https://github.com/kundajelab/deeplift>

¹¹ <https://github.com/ankurtaly/Integrated-Gradients>

¹² <https://github.com/jimmie33/Caffe-ExcitationBP>

¹³ <https://github.com/karpathy/neuraltalk2>

¹⁴ <https://github.com/HendrikStrobel/LSTMVis>

¹⁵ <https://github.com/aravindhm/deep-goggle>

same neuron outputs in a layer in a feed-forward CNN. By doing so, it shows the variance and abstract notions of concepts the CNN is considering in each layer.

Decompositional global explanation

Rather than explaining a single outcome, these iML methods aim to explain the entire model through decomposition. Their decompositional nature makes them model-type dependent. Below, we discuss several approaches for neural networks (NNs), support vector machines (SVMs) and tree ensembles (TEs).

Neural networks. Odajima et al. [134] propose to learn classification rules by decomposing the input space in the input layer of a NN. Their method, Greedy Rule Generation (GRG) can extract decision lists for discrete outputs by clustering hidden activations in neurons and seeing how rules perform on these activation values. Rule extraction by Reverse Engineering the Neural networks (RxREN) [6] can extract decision rules from NNs using a process consisting of two phases. The first phase prunes the insignificant neurons to discover their influence on classification. Next, in phase two they take the significant inputs and their data ranges to generate rules and prune them to create minimal length rules.

TreeView [181] creates a surrogate DT model of a supervised (deep) NN by decomposing the feature space into factor partitions. It builds *meta-features* (that are easy to interpret but still predict well) for the factors, and decides on the overall factor label by predicting instances within the factor cluster.

A large body of work has focused recently on *prototype reconstruction* (PR) methods for DNNs (CNNs and RNNs). Simonyan et al. [168] propose a decompositional approach for PR, that can generate class saliency images that are representative of a class for *convolutional* NNs (CNNs). In order to do so, it creates the derivative w of each pixel, and takes the maximum value across all channels (RGB) of w —a procedure called *activation maximization* (AM). As a result, it obtains the pixels which need to be changed least in order to affect the class score the most. By ranking these pixels, the ones with the highest influence can be visualized. DeepVis¹⁶ [204] allows for the interactive analysis of individual neurons in a trained (deep) feed-forward NN by showing (i) AM of the neuron for a class using gradient ascent, (ii) top images (prototype selection) that resulted in the highest activation for this neuron, and (iii) the corresponding deconvolution [206] for these prototypes. Furthermore, they introduce a regularized method for AM of classes of deep CNNs. Karpathy et al. [85]¹⁷ characterize the behavior of RNNs by visualizing patterns in LSTM units. By highlighting AM of the $\tanh(\cdot)$ function for each character in natural language text, their method shows where the model is sensitive to—showing patterns such as sensitivity to quotes (turns of inside quotes and turning off outside of quotes), sensitivity to sentence length, or activation of the cell in if-statements. Note that even though the behavior of some cells is easily interpretable, Karpathy et al. [85] found that in most cases their behavior is too complex to be

¹⁶ <https://github.com/yosinski/deep-visualization-toolbox>

¹⁷ <https://github.com/karpathy/char-rnn>

directly understood from highlighting attention. Nguyen et al. [132]¹⁸ observe that even though AM is an attractive approach due to its simplicity, it often produces uninterpretable images—because the images do not resemble the natural images that the neuron has learned to detect. Instead, their approach uses a *natural image prior* (i.e., a real image the data was trained on) in addition to the AM function to create human-recognizable images showing the prototype of what a neuron’s visual attention is at.

A recent promising approach by Shwartz-Ziv and Tishby [166] supports direct comparison of DNN architectures by studying their information paths in the information plane. It treats each layer in a NN as a single random variable, that shows how much mutual information a layer preserves between its input and output variables. By visualizing the network layers over training epochs, the network prediction and generalization quality can be assessed. These visualizations allow for preliminary actionable insights, that they also use to support previous claims of DNN: (i) hidden layers may dramatically reduce the number of epochs required for good generalizations, and (ii) that DNNs learn better when more data is available.

The Neural Interpretation Diagram (NID) [135] shows the *feature importance* (FI) of inputs features in a feed-forward NN by visualizing the significant positive and negative effects on the outcome on top of the NN using a randomization approach. These effects can then be partitioned to determine the relative FI of the input neuron features on outputs. Tzeng and Ma [188] visualize a NN weights for individual decisions or whole data sets, by showing the relative contribution of input neurons and weights as the sum of all hidden nodes affecting that input. It is shown graphically on top of the NN, where the color indicates the mean value and standard deviation for that node. By ordering on these two statistical indicators, the feature that most positively/negatively contribute to the decision can be determined.

Support vector machines. While SVMs remain popular, their mappings to feature space may result in very opaque decision boundaries. Two methods have tackled this issue by introducing rule-extraction methods for SVMs. SVM+Prototypes (SVM+P) [133] takes the separating hyperplanes determined by the SVM, and generates *if-then* rules (decision sets) by selecting areas a minimal number of areas around prototype points for each class—with maximal coverage. These prototypes (most descriptive examples for each class) are found by clustering the data points for each class. ExtractRules [56] consider the hyperplane in input space for any arbitrary linear classifier, and describe this region with non-overlapping decision rules.

Tree ensembles. The first popular approach for create more interpretable versions of TEs was extracting a small number of *decision trees* (DTs) from the ensemble that best describe the full behavior of the ensemble. In 1998, Chipman et al. [26] found that many trees in a TE are very similar in how they make a decision. To explain their behavior, Tree Metrics picks a small subset of the trees that describe the behavior of the group of tree ensembles most accurately. Confident Decision Tree (CDT) [162] finds a small number of DTs in the TE that most confidently classifies the data as correct, by iteratively removing misclassified instances. Tree Space Prototype

¹⁸ <https://github.com/Evolving-AI-Lab/synthesizing>

(TSP) [179] extracts the most descriptive prototypes for Random Forests by deciding on their proximity—based on which instances are classified in the same leaf nodes for two trees. This approach poses a number of benefits, such as being able to handle continuous and categorical features, the ability to handle missing data, and being insensitive to outliers. GENESIM¹⁹ [192] genetically constructs a single DT from a TE by traversing the space of model combinations and merging the hyperplanes they split the data set into. For an optimal DT, GENESIM uses multi-objective optimization of maximum accuracy and minimum tree size. Rather than extracting a DT, Hara and Hayashi [69] propose to reproduce a set of *decision rules* from a DT. First, they train an axis-aligned TE model on the input data. Next, their method extracts an interpretation model of that reduces the number of regions, while minimizing model error.

Pedagogical local explanation

These explanations are formed either by (i) perturbing the input data and measuring changes on outcome, or (ii) annotating individual images using a class definition.

Input perturbations. These explanations are formed to explain individual outcomes by perturbing the input data and observing how that affects the outcome.

The first methods for pedagogical local explanation focused on determining the *most important featured* that contributed to an individual decision. These general methods hold the benefit that they continue to work even when the model is modified or replaced. Strumbelj and Kononenko [173] propose to acquire these feature importance by perturbing the input data and seeing how changes in the input data correspond to changes in outcome. Their basic premise is that when a change in input feature greatly changes the outcome, this input perturbation has a large importance for determining the output. Instead of testing out all perturbations, they use an efficient sampling-based approach to acquire feature importance. Nevertheless, Martens and Provost [119] found that their method still does not scale well with large input spaces—text data required too many perturbations to efficiently provide explanations. Search for Explanations for Document Classification (SEDC) [119] uses a faster heuristic search for determining leaving out which words will change the classification of an instance. By doing this for multiple instances and combining their feature importance, they can globally estimate the words that contribute most to a decision. *Rivelo*[177] is an interactive interface using SEDC, which allows users to interact with the underlying data set by showing feature importance and example documents belonging to text classification decision.

Turner [187] proposes using a local interpretable predictor around a point to explain to provide the explanation for its outcome. By using this approach, the Model Explanation System (MES) [187] does not have to give into predictive accuracy while also being able to provide an explanation. To this end, MES can explain any arbitrary black-box classifier using *axis-aligned* or *linear* decision boundaries. They use data points generated based on the input distribution, and finding an explanation function that provides the most accurate explanation with highest confidence.

¹⁹ <https://github.com/IBCNServices/GENESIM>

Fong and Vedaldi [50] apply the simple principle of input data perturbations on *image data* to create *saliency maps*. By perturbing (deleting, introducing noise or blurring) regions of the input image, they can determine contribution of the regions to the image classification decision. These regions are found using meaningful image perturbations, that try to find the regions where blurring causes the largest output change using gradient descent.

While the majority of iML methods focus on explaining what caused a particular outcome, Tolomei et al. [185] argue that in *model debugging* and when we *disagree with the outcome* we can benefit from seeing how the input should have changed in order to change the outcome.²⁰ For example, in the healthcare domain if a patient is classified as a high-risk patient, it is a useful insight to see which clinical indicators should be changed to make this patient a low-risk patient. For use in tree ensembles, the method of Tolomei et al. [185] generates proposed instances based on the alternative paths in decision trees that change the prediction outcome—determining the minimal changes required in inputs to change the outcome from a true negative (TN) to a true positive (TP).

Pedagogical global explanation

Global pedagogical explanations come in many representations. They query a black-box model and create a single method for explaining (part of) the model, e.g. the entire model or an explanation per class. In the following paragraphs, we discuss the found iML methods per type of representation.

Decision rule generation. Many works have focused on creating *decision rules* for tabular data in a pedagogical manner. Craven and Shavlik [31] propose to create an accurate symbolic representation of a NN classifier by extracting *conjunctive if-then* and *M-of-N* rules based on sampling of how rules perform for a given class. Rules are learned by querying the NN as an oracle, where the NN is restricted to discrete outputs and input values that are either Boolean or nominal. Rule Extraction From Neural network Ensemble (REFNE) [210] generate instances using NN ensembles, and then try literals (i.e., feature-value pairs) that are above a fidelity threshold. It iteratively creates a decision list using a *sequential coverage algorithm*: first generating length one rules, removing the instances in the data that are correctly classified by these rules, then generating length two rules, etc. Genetic-Rule EXtraction (G-REX) [83] can extract Boolean and Fuzzy decision rules from an arbitrary opaque supervised model using *genetic programming* on the training data—generating candidate rules and evaluating them based on interpretability (rule length), accuracy and fidelity. Johansson and Niklasson [82] expands on G-REX by training the supervised model on (i) training data, (ii) training data with black-box model outputs, and (iii) oracle data (new instances generated a queried using the black-box model). Model Agnostic Globally Interpretable Explanations (MAGIX) [144] finds a single conjunctive rule per class (decision set) with the highest F_1 -score, minimal length and maximal coverage using genetic programming.

ExOpaque [66] was specifically designed to work for tree ensembles (TEs), but falls within the pedagogical approach as it does not decompose the TE model. It uses *Inductive Logical Programming*

²⁰ Note that altering the causes in order to change the outcome is *counterfactual* explanation.

to extract a set of Horn clauses (conjunctive decision sets, where all feature-value pairs are only allowed to be simple statements) from any arbitrary black-box model. These Horn clauses describe the black-box model with high fidelity, while maintaining the relative simplicity for human interpretation. Simplified Tree Ensemble Learner (STEL)²¹ [38] also focuses on supervised TEs, and can extract a minimal length decision set using a sequential coverage algorithm that is able to most accurately predict the highest frequency of the oracle data set.

Decision tree generation. The first widely recognized pedagogical for extracting decision trees (DTs) from classifiers is Trepan [32]. Following their work on decision rules, Craven and Shavlik [32] query the model as an oracle, where inputs are generated by randomly selecting feature values within constraints determined using the range on the input data. Rather than using standard DT induction algorithms, Trepan expands its tree based on the split that has the greatest potential of increasing the fidelity of the tree to the black-box model. Krishnan et al. [96] adopt the pedagogical paradigm of Craven and Shavlik [32] to inductively learn a decision tree (DT) for classification, by generating prototypes most representative of a class—merely taking into consideration the inputs and probabilistic outputs—, and then training a DT on these prototypes. While the latter two mainly focused on DT extraction from NNs, Combined Multiple Models (CMM) [41] was motivated by DT extraction from TEs—also querying the underlying model as an oracle based on input data distributions and approximating the classification task with a single, interpretable tree. 18 years later, Zhou and Hooker [209] continue on the idea of CMM by proposing a method that also pedagogically generates a single DT based on input distributions: Single Tree Approximation (STA). They improve the chosen splits to better mimic the behavior of the underlying TE, by using a stabilized node selection approach that ensures that enough samples are obtained to remove the variability of random data generation. DecText [16] creates DTs that maximize fidelity to the black-box model while minimizing the tree size using a special splitting technique. DecText can handle continuous features. It trains on unseen instances, a mathematical model of the data, or randomly generated data based on the original data distribution. Bastani et al. [10] extend pedagogical axis-aligned DT extraction to black-box reinforcement learning (RL) models that learn through a Q-function.

Feature importance. Strobl et al. [172] improve feature importance methods for tree ensembles by accounting for their bias resulting from correlations among features, the number of categories in discretization, and their scale of measurement. To combat this, they determine the feature importance based on the difference in accuracy with or without the input variable (permutation importance). Generalized Additive Models plus Interactions (GA²M)²² [113] are linear models of non-linearly shaped features that provide a good trade-off between intelligibility and accuracy. Their linear nature allows for visual inspection of the input features on the output—thereby unveiling feature importance. GoldenEye²³ [73] looks at feature sets and their influence on an outcome by comparing the consistency of a black-box predictor when randomly permuting the input

²¹ <https://cran.r-project.org/web/packages/inTrees/index.html>

²² <http://www.cs.cornell.edu/~yinlou/projects/gam/>

²³ <https://bitbucket.org/aheneliu/goldeneye>

data set. This allows the explainee to find groups of features whose interactions are most responsible for affecting the predictive performance of the black-box model. Orthogonal Projection of Input Attributes (OPIA)²⁴ [2] give black-box predictors two equal inputs, except for one changed feature. Depending on how this feature permutation changed the outcome, OPIA can define a feature ranking for any black-box regressor or classifier.

Partial Dependence Plots. *Partial dependence plots* (PDPs) show how changes in the input affect prediction outcomes. Variable Interaction Network (VIN) [78] observed that PDPs only show the influence of one variable on another, resulting in a difficult interpretation on a global level. Thus, they visualize the importance and feature interactions as a network—where the nodes represent features, and edges are shown when an interaction is present between these features. Individual Conditional Expectation (ICE)²⁵ [62] extends PDPs by showing one line on the plot for each input. Prospector²⁶ [94] acknowledges that simply sampling input values using interpolation from the observed data poses shortcomings when looking for input feature perturbations to obtain a PDP. It can miss feature values, or create feature values that are impossible to achieve for the predictive function. Instead, Prospector takes into account the original context of the data, and allows for local explanations and feature importance of individual outcomes. The human-in-the-loop approach allows data scientists to interactively and visually inspect a black-box model by seeing local and global PDPs and feature importance, allowing for the inspection and comparison of models. Gradient Feature Auditing (GFA)²⁷ [3] obscures the influence of a feature to see how this changes the outcome, determining the *indirect influence* of a given feature.

Sensitivity analysis. Sensitivity analysis (SA) is similar to PDPs in that they show the effect of features on the outcome, but rather than approximating this globally SA compares it to a reference—making them computationally more efficient. Baehrens et al. [8] aim to explain individual classification decisions of any model by using *explanation vectors*—directional arrows that show how the change the input features in order to change the classification decision. For individual points, these explanation vectors can yield feature importance and counterfactual changes resulting in a different classification outcome. Global Sensitivity Analysis (GSA) [29] is a generalized visualization approach based on SA, that is applicable to any supervised black-box model for any arbitrary type of input.

Other. Prototype Selection (PS) [13] allows for explanations of unsupervised and classification decisions by finding a small set of *prototypes* for each class/cluster region that cover as much of that region as possible, while minimizing coverage for other regions.

Zahavy et al. [205] propose a method for showing the state space of a *Deep Q Network* (DQN) by grouping their visual representations using t-SNE. As a result, groups with similar states end

²⁴ <https://github.com/adebayoj/FairML>

²⁵ <https://cran.r-project.org/web/packages/ICEbox/index.html>

²⁶ https://github.com/nyuvis/explanation_explorer

²⁷ <https://github.com/algofairness/BlackBoxAuditing>

up in the same area of the visualization, showing a generalized state space for the reinforcement learning (RL) model.

For model-based RL, Genetic Programming for Reinforcement Learning (GPRL) [70] can learn *policy equations* (decision rules true at a given time step) using genetic programming from pre-existing example tuples of the current state, action, reward and next state.

Sánchez et al. [161] create a Bayesian network decision tree and first-order logic (FOL) rules based on unsupervised *matrix-factorization* models. These extracted models serve as an interpretable proxy for the black-box models, that become unintelligible due to the large matrix size (e.g., for a movie recommendation system the rows are ll users and columns are all movies).

Combining local explanations to explain global behavior

A recent trend in iML methods is to leverage the benefit from local pedagogical explanation—i.e., they allow for relatively simplistic models as they only have to be locally faithful to the model-labelled data points—to create outcome explanations, and then combine these local models to explain the (complex) model behavior globally.

In 2016, Datta et al. [36] proposed Quantitative Input Influence (QII), that measures the degree of influence of inputs on outputs of the model. They can compute them locally for an individual outcome, while also accounting for correlations in influence. In addition, these values can be visualized globally to show the QII on all outcomes when varying the range of an individual variable—similar to partial dependence plots (PDPs).

Local Interpretable Model-Agnostic Explanations (LIME)²⁸ [148] locally generates data around a single prediction, and fits a linear model that shows the relative importance of features in that local neighborhood. It is able to do so for tabular, image and text data. By combining many local explanation (in a procedure referred to as *submodular pick*), they are able to form a global explanation of feature importance for a model. While the model-agnostic nature LIME makes it appealing in practice, due to its reliance on random data points its running time is slow on dense inputs like images. Therefore, Streak²⁹ [48] provides a speed-up of LIME by using a weakly submodular search for perturbations—that does not rely on a large set of random perturbations for defining an explanation but instead genetically searches for one.

SHapley Additive exPlanations (SHAP) [114]³⁰ show that *Shapley* values—a notion from game theory, showing how important each players contribution was to a game—can unify and justify multiple iML methods, such as LIME [148], DeepLIFT [164] and LRP [7]. They calculate the feature importance (FI) for individual predictions using *Expectation Shapley* values that are trained on the outputs of the black-box oracle and enforce monotonicity and are able to account for colinearity. By stacking FIs horizontally along a range of inputs, they are able to show the sign and magnitude of FI for each feature for the whole model.

²⁸ <https://github.com/marcotcr/lime>

²⁹ <https://github.com/eelenberg/streak>

³⁰ <https://github.com/slundberg/shap>

Anchors³¹ [149] improves on their previous work [148]. Ribeiro et al. [149] acknowledge that the shortcoming of linear explanations is that their *coverage* is not clear, i.e. it is unclear when an explanation does and does not apply. To tackle this, they propose to use Anchors—*if-then* rules that describe a local area of the model with a clear boundary—where for instances for which the anchor holds the prediction is (almost) always the same. For example, the anchor $A = \{ 'not', 'bad' \}$ illustrates that the presence of the words 'not' and 'bad' in a sentence will typically result in a positive prediction of a sentiment analysis model. Moreover, they combine these anchors with *prototype selection* on the generated neighborhood data, to enhance explanations with other useful examples and counterexamples for which the anchors does and does not hold.

4.5 Discussion & Conclusion

This section reports and discusses the main findings. We then examine the most relevant threats to validity, and outline the main conclusions.

Discussion of results

Overall, we identified 84 iML methods. The first identified method was published in 1994. Figure 4.6 shows a large increase in the number of iML methods from 2014 onwards. The number of methods peaked in 2016 with 19 methods published that year.

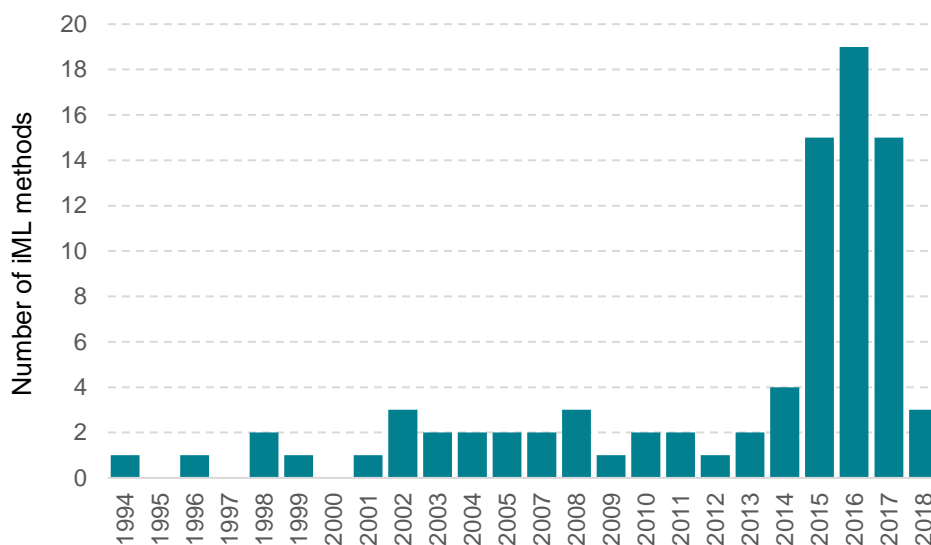


Figure 4.6 Number of iML methods per year

In the following paragraphs, we discuss four aspects of the literature review: the predominant explanation methods, popular representation types, human evaluation, and source code provision.

³¹ <https://github.com/marcotcr/anchor>

Explanation methods. Table 4.5 cross-tabulates the explanation scope and approach according to the taxonomy in Section 4.2. iML methods are predominantly focused on global explanation (69.0%, $n = 58$) rather than local (25.0%, $n = 21$). Five tools (6.0%) are able to provide both. In recent years, the focus has shifted from global explanation to local explanation, and extending local approaches to provide global explanations. Local models are able to structure a global model into more understandable chunks, that still perform well [157]. As such, they are able to capture patterns in subsets of the data in human-understandable ways [157]. The shift of focus to local models coincides with the need to explain more complex input data (e.g., images) and model types (e.g., DNNs).

Table 4.5 Number of iML methods taxonomized

Scope / Approach	Transparent	Decompositional	Pedagogical	Total
Global	14	15	29	58
Global & local	0	1	4	5
Local	2	13	6	21
<i>Total</i>	16	29	39	84

Most methods are pedagogical (46.4%, $n = 39$), closely follow by decompositional (34.5%, $n = 29$) methods. In addition, a large number of studies describes transparent (19.1%, $n = 16$) approaches.

Nearly all methods are able to explain classifiers (95.2%, $n = 80$). 17 of these methods provide interpretations for regression analysis models (20.2%). Five (6.0%) focus on unsupervised learning, and four (4.8%) on reinforcement learning. The majority of decompositional methods focus on explaining (deep) neural networks (75.9%, $n = 21$). Five focus specifically on tree ensembles (17.2%) and two on (6.9%) SVMs. They focus mostly on tabular input data (59.5%, $n = 50$) or explaining images (20.2%, $n = 17$). Eight methods (9.5%) provide explanations for text input data. Twelve methods (14.3%) are able to provide explanations for all three types of data. Note that one method is able to provide explanations for both tabular and text data, and two of these are able to provide explanations for image and text data.

While iML was focused primarily on explanations for tabular data in classification tasks, the field has seen increased interest in moving beyond these domains and creating more generic explanation methods for any type of input data, any type of ML task, and any type of model.

Representation types. Table 4.6 presents the number of iML methods per representation type as described in Section 4.3. Methods are predominantly explained using rule-based representations—i.e., decision rules (DR) or decision trees (DT). They account for nearly half (48.8%, $n = 41$) of all studied iML methods. Another popular representation for explanation is feature importance (FI) with 20.2% ($n = 17$). Twelve methods (14.3%) provide explanations through saliency maps, while six methods (7.1%) use prototype-based explanations.

The popularity of rule-based methods stems from that they are a natural representation to humans [80, 158] and that they allow for multiple types of explanation (e.g., counterfactual) [107]. However, especially with text and image inputs they pose the downside of becoming too large

to comprehend. As such, there is a preference for feature importance, prototypes and saliency in such input domains.

Table 4.6 Number of iML methods per representation type

AN	Rule-based			PDP	Prototypes		Saliency			Other
	DR	DT	FI		PR	PS	AM	SA	SM	
2	27	14	17	4	4	2	4	3	5	2

A lack of human evaluation. A mere 7.1% ($n = 6$) of methods perform human evaluation. The first empirical evaluations using human subjects was published in 2015. Before that, method evaluation typically consisted of measures such as fidelity, accuracy and explanation length. One of these human evaluations was a single case study, while five conducted a user experiment. While scarce, these empirical evaluations provide a baseline for future work in establishing interpretability in iML methods.

Krause et al. [94] performed a four-month longitudinal case study on five data scientists using their method Prospector to predict diabetes. During bi-weekly meetings, they found that data scientists were able to distinguish how different models treat features differently, that they were able to uncover how data imputation affected on predictions, and were able to extract actionable insights from the model decision-making process.

Kim et al. [88] found that BCM explanations improved classification accuracy of 24 participants compared to LDA groups in determining the correct recipe for a set of required ingredients. However, subjects were not quicker in determining the recipe for either method, and did not report a preference for BCM over LDA. In an experiment on 46 graduate students, Bastani et al. [10] found that users were able to perform similarly in terms of understandability using their method, while being able to come to these conclusions more quickly. Lakkaraju et al. [100] showed in an experiment on 47 students that decision sets resulted in more accurate decisions while also improving decision speed. Using 27 graduate students, Ribeiro et al. [148] demonstrated participants were able to more accurately detect bad classifiers when using their method.

Software availability. With the goal of application of interpretability methods in practice, software availability has been acknowledged as a significant factor [33]. One third ($n = 28$) of iML methods is implemented as open-source software, with the most popular language being Python ($n = 20$), followed by Lua and R (both $n = 3$). One tool is implemented in Java, and one in Matlab.

Limitations

We discuss four aspects of threats to validity [199] in this literature review.

Construct validity concerns consistent understanding of the used constructs. The key threat here is the ill-defined nature of the construct ‘interpretable machine learning’ (iML) and ‘explainable

artificial intelligence' (XAI). This risk re-occurs in the search strategy, as rather than a search string a number of previous overview studies were taken as a baseline for extracting relevant studies. Misunderstanding of constructs was minimized by using an individual researcher. For the systematic literature review, we followed the guidelines [90] to design the research goals, search and assessment criteria. We also reported the search process to address potential threats to construct validity.

Internal validity focuses on how the study minimizes systematic error. The main threat is related to the bias on the process of selection of the studies. To mitigate this issue we used the guideline to perform the systematic review according to a pre-defined search process as proposed by Kitchenham [90]. To increase cover, additional articles were manually introduced to the overview. Articles were extracted from the main software engineering electronic databases, which may have missed out on important results. Moreover, additional databases could have produced complementary missed information. Finally, there may be individual researcher bias introduced in their subjective interpretation during data collection.

External validity threats reduce the generalizability of the results. The extracted results spanned in the time period from 1994 to 2018, and may therefore not be generalizable across broader time periods. Currently these methods were studied qualitatively, further analysis may be performed quantitatively to enable analytical and statistical generalizations.

Reliability concerns the dependence of this research on specific researchers. With the aim of broad coverage and identifying tools, not all included studies were published in peer-reviewed scientific venues. In such cases, the face validity was assessed by the researcher before including it in the study. Articles were still judged according to the inclusion criteria. When replicating the study, due to the rigorous search process when the selected primary studies are equal we expect similar results.

Conclusions

This chapter systematically reviewed interpretable machine learning (iML) methods to analyze the state-of-the-art to make machine learning interpretable. Its main contribution is the exhaustive overview and summary of iML methods. Nevertheless, we also obtained some valuable insights.

How iML methods enhance interpretability in machine learning. Even though scarcely backed up by empirical evidence, *rule-based and feature importance explanations* are typically chosen as a representation. They pose the benefit of being natural representations to humans. *Local models* provide the benefit of being able to capture a part of the model in a more understandable manner, and may therefore be more valuable when the goal is to explain more complex models, data types and phenomena. Lastly, the approach taken to explain the model, and the representation used depend on (i) the type of input data it aims to explain, (ii) the range of models it aims to explain, (iii) the type of ML it aims to explain (e.g., supervised or reinforcement learning), and (iv) the goal and intended end-users of the explanation.

Implications for contrastive explanation. In recent years, many methods have proven to be effective in being able to provide model-agnostic explanations for any type of representation and ML type. Such generic methods pose the benefit that they are future-proof (e.g., able to provide explanations for ML models currently unknown) and generally applicable. Especially with tabular data, rule-based explanations have shown to be widely adopted and deemed interpretable in end-user experimental evaluations. As contrastive explanations provide explanation for a single data point, we can utilize the benefits of *local, pedagogical, rule-based* explanations for our approach. Notwithstanding, we should caution against claims of interpretability without empirical evidence.

FOIL TREES

Foil Trees provide targeted model-agnostic contrastive explanations for any foil. A contrastive explanation rule is extracted by training a foil-versus-all decision tree, and extracting the disjoint set of rules that causes the tree to predict the output as the foil instead of the fact. Our method show promising results as it is able to mimic the decision boundaries used by the model it aims to explain (94% fidelity), generalizes well on unseen data (88% accuracy), while providing 78% more concise explanations than non-contrastive ones (mean length 1.19, improvement of 4.18).

Published as

J. van der Waa, M. Robeer, J. van Diggelen, M. Brinkhuis, & M. Neerincx, "Contrastive Explanations with Local Foil Trees", in *2018 Workshop on Human Interpretability in Machine Learning (WHI 2018)*, 2018, pp. 41-47.

5 FOIL TREES: CONTRASTIVE EXPLANATIONS FOR MACHINE LEARNING

Chapter 4 unearthed that the majority of interpretable machine learning (iML) methods provide explanations in terms of their input features. Decision trees, decision rules, feature importance, saliency methods and partial dependence plots all use features as the main means for conveying explanations. Correspondingly, we build on using feature explanations by creating explanations using transparent locally faithful models in input space.

This chapter describes our approach and its implementation. Section 5.1 introduces the end-to-end approach for contrastive explanations in machine learning (ML). Section 5.2 provides details on the implementation of `ContrastiveExplanation`.

5.1 Contrastive Explanation as Binary Classification

We propose a model-agnostic (pedagogical) approach for local contrastive explanations, where we view contrastive explanations as a binary classification problem. It is pedagogical in the sense that we view the model to explain as a black-box—merely using its inputs and outputs to explain its behavior—, and local because we provide explanations for a questioned data point—only requiring the explanation to be locally faithful. We provide *targeted contrastive explanations* by training an arbitrary model to distinguish between fact and foil that is more accessible.

Because we aim to explain a single data point, this problem can be reduced to a foil-versus-rest classification problem. Given that the fact data point is outside of the foil class, the set of contrastive explanations are all explanations that cause the data point to be classified as the foil instead of the fact. Figure 5.1 illustrates this approach for an example classification problem, in which x is the questioned data point (of class 1), and the foil is determined to be class 2.¹

Below, we detail strategies for fact and foil extraction, and for neighborhood data generation.

Fact and foil extraction. Recall from Section 3.3 that the possible set of facts and foils (the *contrast class*) depends of the type of ML. As such, given the ML type we are able to provide the explainee with a set of potential foils. The fact is determined by applying the ML model to explain to the fact-sample (i.e., data point to be explained). We propose multiple strategies for determining the foil:

¹ Note that in the future this same approach can be extended to a *probabilistic* representation, where each data point is labeled with a probability that it is part of the foil class.

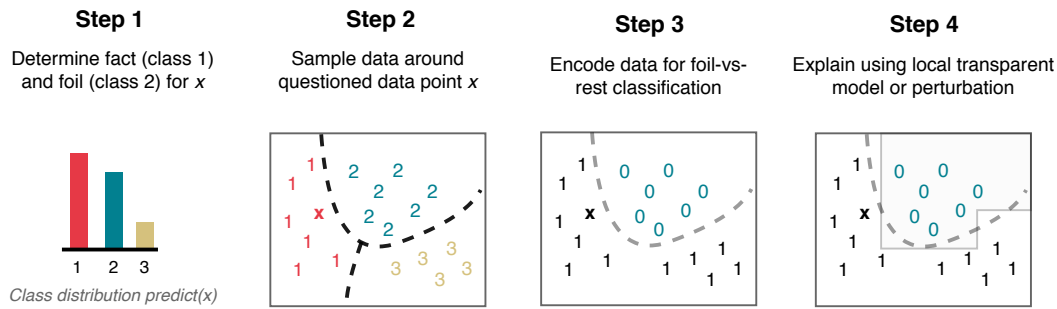


Figure 5.1 Overview of constructing targeted contrastive explanations for questioned data point x in an example classification problem

- ▶ Explicitly given by the explainee;
- ▶ Learned from previous preferences by explainees or similar explainees, or;
- ▶ Automatically extracted based on the model prediction (e.g., the second most likely class in classification, or a measure of central tendency in regression analysis).

Neighborhood data. Neighborhood data is sampled around the questioned data point x to construct an explanation. To do so, multiple sampling strategies can be used, such as:

- ▶ Randomly sampling from the original training data;
- ▶ Generated according to a normal distribution for each feature (e.g., [148]), or;
- ▶ Generating according to the marginal distribution of each feature in the training data (e.g., [19]).

To ensure local fidelity (i.e., local truthfulness to the model), similar to LIME [148], all sampled data points are weighed according to their similarity to questioned data point x . Currently, this is done using a *radial basis function* (RBF) kernel using the squared Euclidean distance. Alternative strategies for measuring distance between points could be their similarity after performing *principal component analysis* (PCA) or similarity in an *autoencoder* network [76].

Foil Trees

The transparent model we propose in this study are *Foil Trees*: a one-versus-all decision tree to recognize the foil class. From the Foil Tree we distill two set of rules; one used to identify data points as a fact and the other to identify data points as a foil. Given these two sets, we subtract the factual rule set from the foil rule set. This relative complement of the fact rules in the foil rules is used to construct our contrastive explanation. Figure 5.2 overviews the steps taken to construct an explanation using a Foil Tree.

The method we propose learns a decision tree centred around any questioned data point. It is trained locally to distinguish the foil-class from any other class, including the fact class (Step 4’).

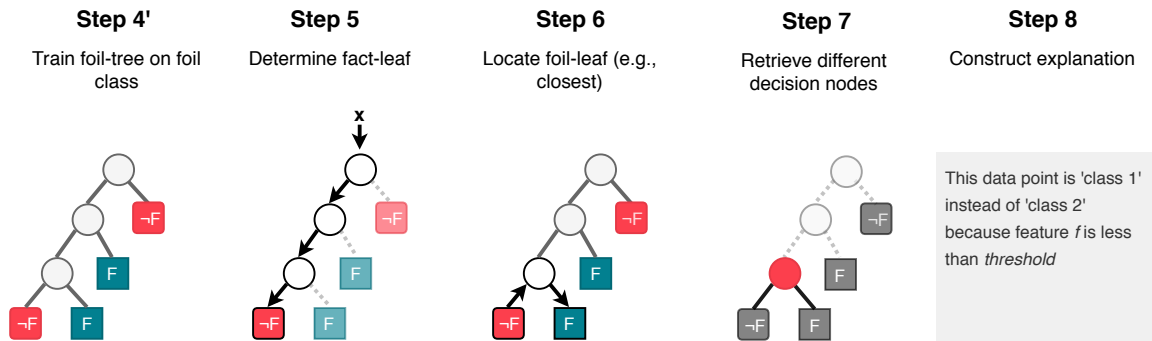


Figure 5.2 The steps required to construct and explanation using a Foil Tree, which distinguishes foil F from the rest ($\neg F$)

Given this tree, the ‘foil-tree’, we search for the leaf in which the data point in question resides (Step 5)—the ‘fact-leaf’. This gives us the set of rules that defines the data point as the not-foil class according to the foil-tree. These rules respect the decision boundary of the underlying ML model, as it is trained to distinguish the foil class outputs. Next, we use an arbitrary strategy to locate the ‘foil-leaf’ (Step 6)—providing us with the set of rules that define the foil. For example, we select the leaf that classifies data points as the foil class with the lowest number of nodes between itself and the fact-leaf.

This results in two rule sets, whose difference define how the data point in question differs from the foil data points as classified by the foil-leaf (Step 7). This explanation of difference is done in terms of the input features themselves. From the decision nodes that remain, those that regard the same feature are combined to form a single literal. Finally, from these literals we construct an explanation (Step 8), that forms the actual presentation of differences between the fact-leaf and foil-leaf.

In summary, the algorithm goes through the following steps to obtain a contrastive explanation using a foil-tree:

1. **Determine fact and foil** for the questioned data point using the model. The *fact* is the output class, and the *foil* is explicitly given or derived (e.g., second most likely class).
2. **Generate or sample a local data set**, either randomly sampled from an existing data set, generated according to a normal distribution, generated based on marginal distributions of feature values or more complex methods.
3. **Encode neighborhood data for foil-versus-rest classification**. For each sampled data point determine whether it is a fact (class 1) or a foil (class 0).
4. **Train a decision tree** with sample weights depending on the training point’s proximity or similarity to the data point in question.
5. **Determine fact-leaf**, the leaf in which the data point in question resides.
6. **Locate foil-leaf**, we select the leaf that classifies data points as part of the foil class with the smallest distance according to a foil-leaf selection (e.g., the lowest number of decision

nodes) between it and the fact-leaf.

7. **Retrieve different decision nodes** to obtain the two set of rules that define the difference between fact- and foil-leaf, all common parent decision nodes are removed from each rule sets. From the decision nodes that remain, those that regard the same feature are combined to form a single literal.
8. **Construct explanation**, the actual presentation of the differences between the fact-leaf and foil-leaf. For example using a fixed interaction with tabular data or text data, or as an image overlay for image data.

Algorithm 1 describes the steps taken after foil-tree construction to form a contrastive explanation. Note that we convert finding the foil-leaf into a shortest-path undirected graph search problem, to allow for different explanations to be provided depending on the selected foil-leaf. The ‘closest’ foil-leaf is then the foil-leaf that minimizes the total path length (i.e., total weight) from the fact-leaf. Observe when the weight for each edge is set to one, the selected foil-leaf is equal to the closest foil-leaf. In order to extract the foil-leaf, we introduce a sink, connected to the foil leaves with edge weight zero.

Algorithm 1: *ContrastiveExplanation(foil-tree, fact-sample, foil-strategy)*

input : *foil-tree*: foil-tree · *fact-sample*: sample to explain · *foil-strategy*: strategy to select to appropriate foil-leaf

output: contrastive explanation as an ordered list of literals

```
1 sink = -1;
2 fact-leaf = GetLeaf(foil-tree, fact-sample);           // get leaf where fact-sample resides in
3 foil-graph = ToGraph(foil-tree, 0);
4 foil-graph = ApplyWeights(foil-graph, foil-strategy); // convert to a weighted graph based
  on foil-strategy
5 foil-leaves = GetLeavesOfClass(foil-tree, class=0);    // get all foil leaves
6 for leaf in foil-leaves do
7   | foil-graph  $\cup$  (leaf, sink, weight=0, -)
8 end
9 explanation = ShortestPath(foil-graph, start=fact-leaf, end=sink);
10 contrastive-explanation = ToDifferencePath(explanation); // convert to contrastive
  explanation by considering both downward rule paths
11 return contrastive-explanation
```

We recursively convert a foil-tree into a graph, while retaining how each node relates itself to other nodes in the graph—ensuring that we can still trace which node we end up in when the expression

in the node evaluates to true or false. This is done according to the steps in Algorithm 2.

Algorithm 2: *ToGraph(tree, node)*

input : *tree*: tree to convert into graph · *node*: id of current node

output: graph with tuples (node_id, node_child_id, right_child_in_tree)

```
1 l = ChildrenLeft(tree);
2 r = ChildrenRight(tree);
3 if l ≠ leaf then
4   left-path = [(node, l, False)] ∪ ToGraph(tree, l);
5   right-path = [(node, r, True)] ∪ ToGraph(tree, r);
6   return left-path + right-path
7 else
8   return ∅
9 end
```

Foil-leaf strategies. Our approach allows for multiple strategies for deciding on an appropriate foil-leaf, by generalizing the foil-leaf selection to a graph-search from the fact vertex to the foil vertex, while minimizing edge weights. The most naive strategy is choosing the closest leaf. However, this strategy may not yield satisfactory results according to the explainee because (i) it may have a high misclassification rate (i.e., low certainty) for the foil class, or (ii) the foil-leaf may only classify a very small number of examples . We propose four strategies:

- ▶ **Closest:** that minimizes the distance between itself and the fact-leaf, creating the shortest explanation possible (i.e., each edge weight is one).
- ▶ **Size:** the foil-leaf which classifies the most number of examples, penalized by the distance in the tree. This mitigates explanations that only hold for very few data points, but rather creates explanations that hold more generally.
- ▶ **Accuracy:** the foil-leaf with the highest relative accuracy in foil classification (most pure leaf), also penalized by the distance in the tree. An explanation should be both accurate and fairly general [33]. Thus, this may result in somewhat more longer and more complex explanations, which nonetheless hold more generally and may therefore be more beneficial.
- ▶ **Combined:** a trade-off between the previous three strategies, where the relative weighting of each strategy is determined based on weights set by the user, or weights learned from user preferences.

5.2 Implementation

We created `ContrastiveExplanation`,² a proof-of-concept tool of our proposed method for extracting contrastive explanations for machine learning. `ContrastiveExplanation` is implemented in Python, with as main dependencies *NumPy* and *Scikit-learn*. For finding the shortest path to

² Source code available at <https://github.com/MarcelRobeer/ContrastiveExplanation>

the foil in the foil-tree, it uses package *networkx*. It currently supports explaining predictions with probabilistic outcomes (e.g., *Scikit-learn* and *Keras*).

In the next sections we describe the software architecture, and provide an example of how to use `ContrastiveExplanation`.

Software architecture

We overview the system functionalities and processes of our implementation with a *Functional Architecture Model* (FAM) [18] in Figure 5.3. The FAM describes the main modules of `ContrastiveExplanation` (CE) and the relations between them. CE has three modules: `FACTFOIL`, `EXPLANATOR` and `DOMAINMAPPER`—all implemented as Python classes.

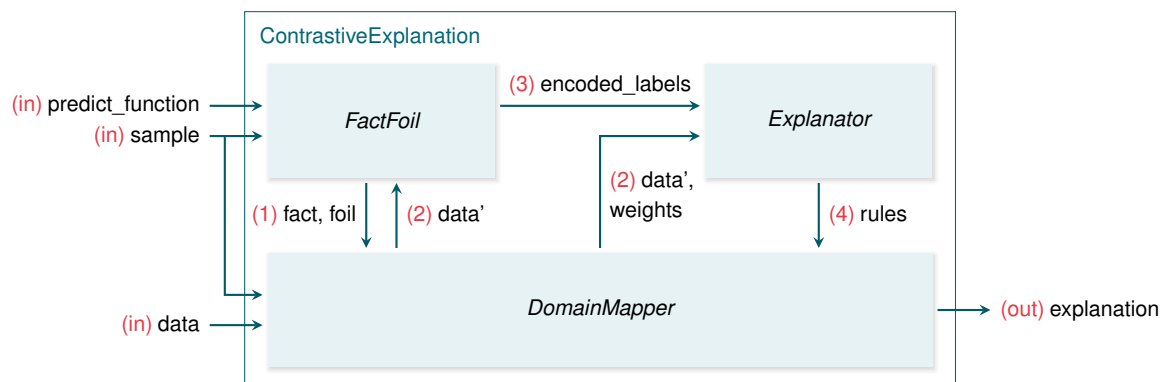


Figure 5.3 `ContrastiveExplanation` functional architecture

CE takes as inputs (i) `predict_function`: the predict function of the model it aims to explain; (ii) `sample`: questioned data point x , and; (iii) `data`: the training data to generate or sample neighborhood data from. As output, it produces a contrastive explanation `explanation`. The process taken to create a contrastive explanation starts with `FACTFOIL` determining the fact and foil for sample x . Next, we sample neighborhood data `data'` around sample x , and determine the weights for each data point based on similarity to questioned data point x . These are provided to `FACTFOIL` and the `EXPLANATOR`. `FACTFOIL` uses the neighborhood data to determine for each data point the `encoded_labels`, of whether the data point is a foil or not. The set of labeled samples (`data'`, `encoded_labels`) with weights `weights` is the used by `EXPLANATOR` to determine the contrastive rules. Finally, the fact, foil and rules are combined into a meaningful explanation in input space.

FACTFOIL is responsible for defining the *fact* and *foil* for a contrastive explanation, and then *encoding* neighborhood data such that the labels are either foil (1) or not-foil (0). To this end, `FACTFOIL` requires the predict function used by the model it aims to explain—to determine the predicted output for each data point—and the sample x it aims to explain. The fact can be determined by applying this prediction function to the sample, while the foil may be induced by looking at predictor probabilities. As a hyperparameter, the foil-search-strategy may be set (e.g., second most probable class, least probable class, or random pick). For classification tasks the implementation

provides FACTFOILCLASSIFICATION, and for regression analysis tasks it provides FACTFOILREGRESSION.

EXPLANATOR converts the weighed neighborhood data with foil-versus-all encoded labels into a set of contrastive rules. The example EXPLANATOR described in this chapter are *Foil Trees*, but other transparent models or perturbation methods may be used. Currently implemented is the TREEEXPLANATOR, that explains using a *Foil Tree*.

DOMAINMAPPER translates the input data in a generic representation, and maps the rules back from its generic representation into a concrete explanation in terms of its input features. By decoupling the domain mapping from explanation, this allows EXPLANATORS to also create explanations in terms of human-constructed or extracted high-level features, or explanations in lower dimensions than the input data. As input it requires training data to sample or generate the neighborhood data from. From this input data it can also infer meaningful labels for the contrast class instances. We provide full support for tabular data (rows and columns) with DOMAINMAPPERTABULAR, and preliminary support for image data with DOMAINMAPPERIMAGE. Here, a meaningful label for the members of the contrast class can be provided, as well as hyperparameters that determine how local the EXPLANATOR should create explanations for the neighborhood data.

Example usage

We illustrate ContrastiveExplanation on a Random Forest trained on the Iris data set [45]. The Iris data set is a multi-class classification problem that contains data about flower characteristics—sepal and petal length and width—and aims to predict whether the flower is one of three species of Iris: *setosa*, *virginica* or *versicolor*. The Random Forest is trained on the training set, which comprises 80% of the 150 data points in the data set.

```
from sklearn import datasets, model_selection, ensemble
seed = 1994

# Train black-box model on Iris data
data = datasets.load_iris()
train, test, y_train, y_test = model_selection.train_test_split(data.data,
                                                                data.target,
                                                                train_size=0.80,
                                                                random_state=seed)

model = ensemble.RandomForestClassifier(random_state=seed)
model.fit(train, y_train)
```

The data provided is tabular (four columns, 150 rows). Hence, we set the DOMAINMAPPER to DOMAINMAPPERTABULAR. We provide the feature names and contrast names as the feature names and class names of the data set, respectively. Next, we take a data point from the test set to provide an explanation for. The data point in question has four features: 'sepal length = 5.1 cm', 'sepal width = 3.3 cm', 'petal length = 1.7 cm' and 'petal width = 0.5 cm'.

```

# Contrastive explanation
import contrastive_explanation as ce

dm = ce.domain_mappers.DomainMapperTabular(train,
                                           feature_names=data.feature_names,
                                           contrast_names=data.target_names)
exp = ce.ContrastiveExplanation(dm, verbose=True)

# Explain first data point in test set
sample = test[0]
exp.explain_instance_domain(model.predict_proba, sample)

```

We explain the instance using the models' probability prediction to determine the foil (by default the second most probable class). We directly map it back to the input domain, to obtain the explanation. The model predicted 'setosa' instead of 'versicolor' because 'sepal width (cm) > 2.714'. Using to their internal representation, the same extracted explanations can also be mapped to a fixed interaction in a dialogue setting:

- CE** The flowertype is 'setosa'.
- User** Why 'setosa' and not 'versicolor'?
- CE** Because for it to be 'versicolor' the 'sepal width (cm)' needs to be smaller.
- User** How much smaller?
- CE** The 'sepal width (cm)' needs to be less than 2.714.

6 QUANTITATIVE VALIDATION

We quantitatively validate Foil Trees on eight benchmark supervised learning tasks from the UCI Machine Learning Repository [45]. To demonstrate the model-agnostic nature of our approach we apply multiple machine learning (ML) models to each task. Section 6.1 introduces the performance metrics used. Section 6.2 describes the validation setup and operationalization. Section 6.3 reports the results, which are discussed in Section 6.4.

6.1 Performance Metrics

In Section 2.5 we introduced multiple performance metrics for automatically evaluating global interpretable ML (iML) methods as defined by Craven and Shavlik [33]. We adapt these metrics and validate the accuracy, fidelity, time and explanation comprehensibility. The mean length serves as a proxy measure demonstrating the relative explanation comprehensibility (interpretability) [43]. The fidelity allows us to state how well the tree explains the underlying model, and the accuracy tells us how well its explanations generalize to unseen data points. Time indicates whether explanations can be extracted in real-time. While these measures were defined for global performance assessment of iML methods, we however aim to assess how the model performs locally. Thus, when assessing the performance of the model the data points are weighed based on their Euclidean similarity to the questioned data point. We detail each measure below:

- ▶ **Accuracy** (*Acc.*) tells us how well the explanations generated from the Foil Tree generalize to unseen data points. It is measured as the F_1 score of the foil-tree predictions on the test set compared to the true labels, weighed by the Euclidean distance from the test set points to the questioned data point.
- ▶ **Fidelity** (*Fid.*) allows us to state how well the Foil Tree explains the underlying model. It is the F_1 score over the foil-tree compared to the model output, also weighed by the Euclidean distance from the test set points to the questioned data point.
- ▶ **Time** is the number of seconds needed on average to explain a test data point.
- ▶ **Mean Length** (*Len.*) is the average length of the explanation in terms of decision nodes. The ideal value is in a range [1.0, Nr. features) since a length of zero means that no explanation is found, and a length near the number of features offers little gain compared to showing the entire ordered feature contribution list as in other iML methods. In addition, we show the improvement over the length of non-contrastive explanations. Non-contrastive explanation length is defined as the number of decision nodes from the root to the leaf the data point resides in, explained by a regular decision tree (i.e., classifier for classification tasks and regressor for regression analysis tasks) on the weighed neighborhood data.

In addition, we obtain two metrics during foil-tree construction and foil-leaf selection: the *confidence* and *local fidelity*. The confidence allows us to state how confident the foil-tree is that the extracted contrastive rule will convert the fact into a foil, while the local fidelity measures how well the foil-tree performs locally on black-box labeled data points.

- ▶ **Confidence** (*Conf.*) measures how certain the foil-tree is that contrastive explanation will ensure that the proposed explanation converts the fact into a foil. It is defined as the correct classification rate in the selected foil-leaf.
- ▶ **Local fidelity** (*Loc.*) describes how well the foil-tree performs locally on black-box labeled data points. It is measured as the accuracy of the foil-tree on the generated neighborhood data around the questioned data point, that the foil-tree was trained on.

6.2 Setup

Data sets

We validate Foil Trees on eight data sets from the UCI Machine Learning repository [45]; five classification tasks and three regression analysis tasks. Table 6.1 summarizes the characteristics of these data sets. Below, we describe each in more detail:

Table 6.1 Data sets for quantitative validation

Data set	Features	Rows	Problem type
IRIS	4	150	Classification (3 classes)
DIABETES	7	768	Classification (2 classes)
HEART DISEASE	13	297	Classification (5 classes)
SPECT	45	267	Classification (2 classes)
CENSUS INCOME	108	32561	Classification (2 classes)
WINE QUALITY	11	4898	Regression
PARKINSON	28	1040	Regression
STUDENT	58	649	Regression

- ▶ **IRIS.** *Iris* is a well-known classification tasks of plants based on four flower leaf characteristics. It has 150 data points and three classes.
- ▶ **DIABETES.** *PIMA Indians Diabetes* is a binary classification task to correctly diagnose the onset of diabetes, based on medical predictor variables. Its data set contains 769 data points and has nine features.
- ▶ **HEART DISEASE.** *Heart disease* aims at classifying the risk of heart disease from no presence (0) to presence (1–4), consisting of 297 instances with 13 features. The features describe patient demographics and summarized medical measurements.
- ▶ **SPECT.** The *Single Proton Emission Computed Tomography* (SPECT) data set contains summary statistics of cardiac images. It has the purpose of classifying patients into abnormal or normal based on 45 features. It comprises 267 data points.

- ▶ **CENSUS INCOME.** United States census data to predict whether a person’s income exceeds \$50,000 per year or not. It contains 32,561 data points with 108 features.
- ▶ **WINE QUALITY.** *Wine quality* is a regression analysis task that aims at predicting the quality score of white wine variants of the Portuguese ‘Vinho Verde’ wine based on physicochemical characteristics. It comprises 4898 data points with 11 features.
- ▶ **PARKINSON.** The *Parkinson Speech* data set contains extracted features of 26 speech recordings of 20 healthy patients and 20 patients diagnosed with Parkinson’s disease. It aims at predicting the Unified Parkinson’s Disease Rating (UDPR) score for each recording—making it a regression analysis task. In totality, it contains 1040 data points with 28 attributes.
- ▶ **STUDENT.** *Student Performance* is a regression analysis problem that aims to predict the grade of a student in a Portuguese language course. The data was taken from secondary schools in Portugal, and contains data about student demographics and school-related features. It contains 649 data points with 58 features.

Please note that these data sets cover a wide range in terms of number of features and rows, and in number of classes.

Machine learning models

We apply five ML models on classification tasks, and three on regression analysis tasks: a *Random Forest* (RF), *logistic regression* (LR; only for classification), *Support Vector Machine* (SVM), and *neural network* (NN). Model selection is performed by searching a grid of parameters, which are selected using ten-fold cross-validation (CV). Table 6.2 describes the used models, their fixed parameters and the grid-searched parameters.

Table 6.3 reports the final performance of the models as the F_1 score for classification (weighted if multi-class), and R^2 for regression analysis. Note that each supervised learning task has one data set with a lower performance than other models—CENSUS INCOME and WINE QUALITY, respectively. They are included on purpose, as they allow us to assess the effect of model performance on the performance of Foil Trees.

Operationalization

The quantitative validation is implemented in Python (3.6.5) using package `scikit-learn` (0.19.1) [139]. Each benchmark on a data set-model pair is cross-validated three times to account for randomness in foil-tree construction. For classification tasks, the foil-tree was constructed for the using as foil *second* most probable class according to the model prediction. For regression analysis tasks we applied two foils to each data point: all data points predicted smaller than the questioned data point, and all data points predicted to be larger. Results were then averaged out across both foils.

The black-box model was trained with a train-test split, resulting in two disjoint sets of 80% training data and 20% test data, respectively. To limit total validation time, for data sets with more than 1000 data points a sample of 1000 data points were randomly sampled as the combined training and test data. For reproducibility, random functions were set to seed 1994. The foil-tree

Table 6.2 Machine learning model settings

Model	Fixed parameters	Grid-Search Parameters
<i>Classification</i>		
RF Classifier	class_weight: balanced subsample	n_estimators: 10, 20, 40, 500
Logistic Regression (LR)	max_iter: 1000 solver: saga	penalty: ℓ_1, ℓ_2 class_weight: balanced, – multi_class: ovr, multinomial
SVM Classifier	kernel: linear	C: 1, 5, 10
NN	max_iter: 2000	hidden_layer_sizes: (50, 20, 10), (250, 100, 20), (500, 200, 100), (1000, 500, 300) alpha 0.01, 0.1, 1
<i>Regression</i>		
RF Regressor		n_estimators: 50, 100, 500
SVM Regressor	kernel: linear	C: 1, 5, 10
NN	max_iter: 2000	hidden_layer_sizes: (50, 20, 10), (250, 100, 20), (500, 200, 100), (1000, 500, 300) alpha 0.01, 0.1, 1

Table 6.3 Performance per model, measured as the F_1 score for classification problems and R^2 for regression analysis problems

Data set/model	LR	NN	RF	SVM	Average
IRIS	0.93	0.97	0.93	0.93	0.94
DIABETES	1.00	0.95	1.00	1.00	0.99
HEART DISEASE	1.00	1.00	0.94	1.00	0.99
SPECT	1.00	1.00	1.00	1.00	1.00
CENSUS INCOME	0.62	0.67	0.53	0.64	0.62
WINE QUALITY	–	0.27	0.36	0.26	0.30
STUDENT	–	0.73	0.87	0.82	0.81
PARKINSON	–	1.00	1.00	0.96	0.99

was trained on $n = 500$ data points sampled from the original training data also used to construct the black-box model. The time required to train the foil-tree and define an explanation was measured on a Dell Latitude E7470 Ultrabook (64-bit, 16GB RAM) running Windows 10 Enterprise (Version 10.0.16299) with an Intel® Core™ i7-6600U processor (2.80 GHz).

After training the black-box model the ground-truth labels and model-predicted labels for the test set are encoded in the same manner as the data used for the foil-tree (e.g., the second most probable class as foil with class label 0 with all other predicted classes labeled as 1). Resultingly, performance assessment can be seen as a binary classification problem—with the F_1 score for classification and R^2 score for regression analysis problems. Note that to measure the local performance, the samples in performance assessment are weighted based on their Euclidean distance from the data point to be explained.

6.3 Results

The results per data set for each measure in Section 6.1 are shown in their respective columns in Table 6.4.¹ In addition, in the second column the total number of explanations created for each data set is included, and the third column describes the number of features in each data set as the upper bound of the explanation length. For each column, the best value(s) are highlighted in green, while the worst performing value is highlighted in red.

On average, the Foil Tree is able to confidently distinguish a foil (*confidence* of 0.96), while having a high local truthfulness to the model it aims to explain (*local fidelity* of 0.99). It generalizes well to unseen data, with a mean *accuracy* of 0.88. Notably, the data sets with a lower accuracy also had lower model performance to begin with (see Table 6.3)—showing similar performance in terms of accuracy to the underlying model. Foil Trees are able to quickly provide explanations (average *time* needed to explain of 60 milliseconds), while accurately mimicking the decision boundaries used by the model well (*fidelity* of 0.94). The contrastive explanations provided are considerably more concise than non-contrastive ones (*mean length* of 1.19 over 5.37), with a mean improvement of explanation length of 4.18 decision nodes.

Table 6.4 Quantitative validation results per data set

Data set	# Explan.	# Features	Conf.	Loc.	Acc.	Fid.	Time (s)	Len.
IRIS	120	4	0.92	1.00	0.98	0.99	0.022	1.16 (+0.70)
DIABETES	614	7	0.99	0.99	1.00	1.00	0.058	1.46 (+4.48)
HEART DISEASE	237	13	0.94	0.99	0.92	0.92	0.027	1.32 (+3.90)
SPECT	213	45	1.00	1.00	1.00	1.00	0.051	1.26 (+4.98)
CENSUS INCOME	800	108	0.92	0.98	0.72	0.81	0.105	1.06 (−0.06)
WINE QUALITY	800	11	0.96	0.99	0.63	0.83	0.065	1.01 (+9.22)
PARKINSON	800	28	1.00	1.00	0.87	0.99	0.079	1.04 (+4.82)
STUDENT	519	58	0.98	1.00	0.89	0.95	0.085	1.12 (+7.69)
<i>Average</i>			0.96	0.99	0.88	0.94	0.060	1.19 (+4.18)

Even though not always fully confident, all data explained models on the data sets were consistently high-performing in terms of local fidelity (ranging from 0.98 to 0.99). Most data sets perform very well in terms of accuracy, apart from CENSUS INCOME and WINE QUALITY—as expected by their lower initial model performance. Fidelity performed consistently over 80%, where data sets with less accurate models also showed the lowest fidelity. The CENSUS INCOME data set took longest to explain. In addition, CENSUS INCOME was the only data set where non-contrastive explanations provided shorter explanations.

Table 6.5 summarizes the results per model.² Across the performance metrics, contrastive explanations performed best on LR. However, it also showed least improvement in terms of explanation length. Random Forests (RFs) and Support Vector Machines (SVMs) performed least well, even though they generally perform with over 90% confidence and fidelity. SVMs proved most difficult

¹ The full results are included in Appendix C.

² Note that logistic regression (LR) was only applied to classification tasks, and therefore was only validated on five data sets.

to generalize, with an accuracy of 0.82. Neural Networks (NNs), RFs and SVMs all performed similar in terms of length improvement over non-contrastive explanation.

Table 6.5 Quantitative validation results per model type

Model	# Data sets	Conf.	Loc.	Acc.	Fid.	Time (s)	Len.
LR	5	1.00	1.00	0.93	0.97	0.029	1.24 (+2.66)
NN	8	0.98	0.99	0.90	0.94	0.056	1.27 (+4.82)
RF	8	0.96	0.99	0.91	0.93	0.103	1.14 (+4.24)
SVM	8	0.92	0.99	0.82	0.93	0.040	1.13 (+4.41)

6.4 Discussion & Conclusion

We summarize and conclude the main findings of the quantitative validation. Next, we analyze and discuss the results in more detail. Lastly, we consider the main threats to validity [199].

Summary and conclusion

We validated ContrastiveExplanation (CE) using *Foil Trees* by applying multiple ML models on eight benchmark supervised learning tasks. Generally, CE performed well—with a fidelity to the underlying model of 94% and an improvement in rule length over non-contrastive explanations of 4.18 (78% improvement). CE is able to quickly provide explanations (60 milliseconds on average) that generalize well on unseen data points (88% accuracy). Across models and data sets, CE can confidently distinguish between foil and not-foil (96% confidence) and locally accurately follows the decision boundary used by the model it aims to explain (99% local fidelity).

Discussion and analysis

Important to note are that performance differed for data sets, as well as for model types.

Data sets. Data sets with lower initial performance also performed less well in terms of generalizability (accuracy) and faithfulness to the model (fidelity). For one data set (CENSUS INCOME) the mean explanation length increased (rather than decreased) with contrastive explanations. However, this simple decision boundary used by the model to explain also had lower performance on the initial data set. Explanation length did not increase with the number of features. Contrastive explanations showed consistent length (ranging from 1.01 to 1.46), while non-contrastive explanations varied greatly on average (1.00–10.46). Contrastive explanations are persistently able to provide shorter explanations that are equally valid, as generally preferred by humans [79].

Model types. Logistic regression (LR) provided the most confident, locally faithful, generalizable explanations—but showed least improvement in terms of explanation length over non-contrastive explanations. CE’s high performance on LR can be traced back to that unlike other model types

it only considers a single coefficient per feature, and thus disregards feature interactions. The same rationale applies inversely to SVMs (that performed least well), as their kernel trick may be difficult to be explain by a model that only creates decision boundary in input space. This trade-off between interpretability and faithfulness to the model has been widely acknowledged [110, 148, 165, 196], and depends on the explainee’s expertise as well as the desired goal of the explanation.

Threats to validity

Construct validity threats concern the degree to which a test measures what it intends to measure. We adapted measures from global interpretable machine learning (iML) methods and weighed them to measure locally. They show CE’s performance, transparency, generalizability and speed. However, the explanation length surrogate does not allow us to infer whether contrastive explanations are more understandable or preferable over non-contrastive explanations. In their adaptation, while the performance measures provide valuable insights into the local performance of CE the choice of similarity measure may affect results.

Internal validity threats focus on how the experiments were conducted. A set of eight benchmark data sets were selected, which were able to produce reasonably well performing models. In addition, the four model types and the hyperparameters that were searched for each model were also manually selected. A wider range of models and hyperparameters may affect the performance. Moreover, time was measure on a specific computer setup, where a different setup may result in different results.

External validity considers the generalizability of results. We validated CE on two types of supervised learning tasks with three or four model types each, with a wide range in terms of number of features and number of data points per data set. Regarding types of machine learning, as it is currently not supported in its implementation we did not cover reinforcement learning tasks—where it is currently unknown how it will perform. We expect similar results in unsupervised learning tasks, as after labeling they are similar to classification tasks. Regarding the types of data, we have only validated our approach on tabular data, and it is yet to be validated on text data and image data.

Reliability concerns the consistency of measurements. We argue that when the same setup is applied the validation should yield similar results. To mitigate these threats, the quantitative evaluation was performed in an automated fashion, and performed using the same seeds. For randomness in foil-tree construction, the validation was cross-validated three times.

EMPIRICAL EVALUATION

While the quantitative validation has shown the feasibility and objective transparency of contrastive explanations, it disregarded the contents of the explanations. We conducted a user experiment on 121 participants to establish how contrastive and non-contrastive explanations are perceived in terms of general preference, transparency and trust. We found that contrastive explanations are preferred over non-contrastive explanations in terms of understandability, informativeness of contents and alignment with their own decision-making. This preference also lead to an increased general preference and willingness to act upon the decision. The experimental evaluation did not confirm that these methods affected the agreement with a decision—regardless of whether the prediction was correct.

7 EXPERIMENT

Chapter 6 showed that Foil Trees are able to provide concise explanations that accurately mimic the decision boundary used by the model, and generalize well on unseen data. While this validation demonstrates the feasibility and objective transparency and generalizability of our approach, it disregards the actual *semantics* (contents) of the explanation. The difficulty in interpretability lies in that it depends on the perceptions and previous experiences of the user seeking an explanation [11]. What one person might call an interpretable explanation may not be interpretable to another person at all. Therefore, we perform a user experiment to establish the *interpretability*, *perceived transparency* and *trust* of our approach. The following subquestion is the focal point of in the current phase of our study:

5.2 *How are contrastive explanations versus non-contrastive explanations perceived in terms of general preference, transparency and trust?*

For this empirical study, a comparative experiment was designed and executed by evaluating contrastive explanations according to the steps of the *empirical cycle* in design science [198]. In previous chapters, we have defined and analyzed the research problem of contrastive explanations for machine learning (ML). Next, we created an experiment design—that describes the objects of measurement, hypotheses, sample, treatments and inferences—and validated this design. We conducted the experiment in July 2018, and used the observations from the collected data to test the hypotheses, and discuss potential explanations, generalizations and limitations to these.

The chapter is organized as follows. Section 7.1 introduces the experiment case used to create the scenario performed by participants. Section 7.2 defines the experimental design and how the experiment was implemented. Section 7.3 presents the results. Section 7.4 shows the threats to validity that were identified when running the experiment, and discusses general findings and generalizations. Lastly, Section 7.5 reports the conclusions.

7.1 Glucose Level Prediction Data

Section 1.1 introduced the action-recommendation use-case children with type I diabetes mellitus (T1DM) that we plan to implement within the problem context. To initially develop and validate the practical applicability and interpretability of contrastive explanations for the action-recommendation use case we have developed an ML model trained on simulated data generated using an extended version of the `simglucose` package, a Python implementation of the FDA-approved *UVa/PADOVA Type 1 Diabetes Simulator* [34]. For various virtual subjects, we are able to simulate the blood sugar (glucose) levels over time as measured by a sensor by providing a diary of meals, exercises and manual insulin injections. The data consists of 138 observations of 20 features, each taken around a time-slice of one hour before and after a food item was consumed. We trained an ML model (dense feed-forward neural network classifier) on this data that is able

to predict whether the food item consumed at a given time will result in a hyper (too high glucose), hypo (too low glucose) or neither within the next hour. It does this by also taking into account the performed and planned activities, and the patient's previous blood glucose levels.

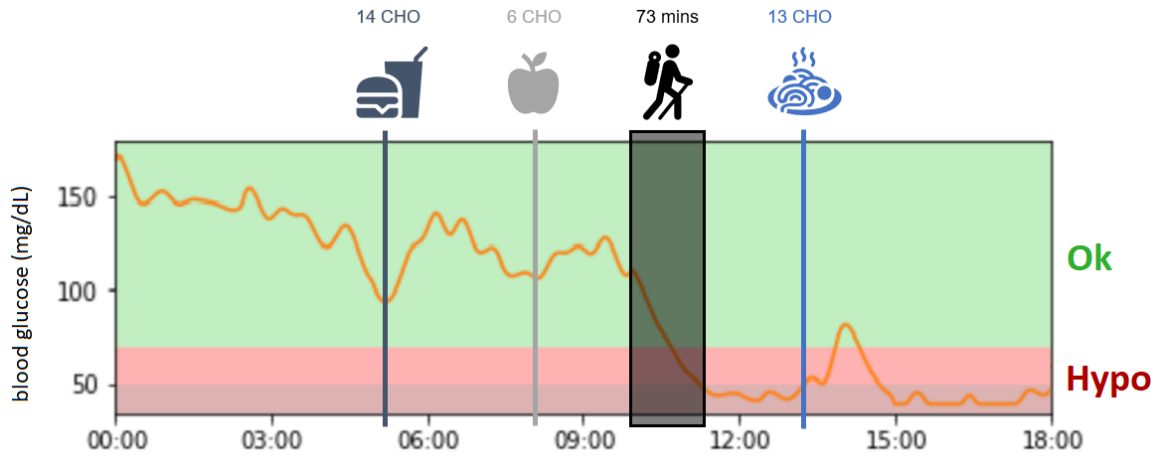


Figure 7.1 Example agenda showing the *sim*Glucose output blood glucose levels based on the patients' diary of meals and exercises

We apply Foil Trees to the prediction model, to generate contrastive explanations that are then evaluated by participants in an experiment. As a basis for comparison, we also provide non-contrastive explanations that are extracted by fitting a decision-tree model on the three classes that is locally faithful and model-agnostic. Figure 7.2 illustrates the four parts of information used to predict the health status of the patient: (1) the *current time*, (2) the *food* consumed at this time and the grams of carbohydrates (CHO), (3) an *agenda of activities* planned and performed, and (4) the *health status* in the previous hour. Note that in order to reduce task complexity the model and explanations disregard manual insulin injections, and are all generated for the same patient.



1	Time Emily asks for a decision	 Current time	17:01
2	Food she wants to eat at current time	 Food	Cereal & milk (40g CHO)
3	Activities performed and planned	 Agenda	16:35 43 minutes of skating (sport)
4	Health status in hour before current time	 Health	Had hyper (high blood sugar) in the past hour

Figure 7.2 Example case with annotations for the four parts of information provided to a participant

7.2 Experimental Design

This experiment aims to investigate the effects of explanation types on user perceptions regarding transparency and trust for ML systems. It constitutes two parts. Part *Preference* aims at establishing the relative preference of explanation types, by comparing non-contrastive explanations, contrastive explanations and no explanations for a given data point of a ML algorithm and corresponding prediction with value judgments. Part *Agreement* has the purpose of determining whether different types of explanations can facilitate trust in a system without causing under-reliance or over-reliance on its decision-making.

Goal

This experiment stems from the premise that the type of explanation offered to a user influences transparency and trust, and affects under-reliance and over-reliance on an ML system. The goal of this experiment is to compare contrastive and non-contrastive explanations for the purpose of establishing the perceived quality in terms of transparency, trust and general preference. To estimate the size of the effect of the difference in perceived quality, we also provide decisions without any explanation. We further subdivide the goal into three subgoals:

1. determine whether there is a general preference for contrastive explanations over non-contrastive and no explanation;
2. establish which value judgments may be responsible for a preference for contrastive explanation over the other two, and;
3. see how different explanation methods lead to over-reliance and under-reliance (a part of trust) when using the system.

Subgoals one and two are covered by 'Preference', while 'Agreement' tests sub-goal three.

Factors and treatments

Each *case* constitutes three parts: (1) an *agenda item*, (2) an *explanation*, and (3) a *prediction* for the agenda item made by the ML system. This experiment encompasses two factors that vary parts two and three of a case, respectively:

1. the *explanation method* offered to the participant (no explanation, non-contrastive explanation or contrastive explanation), and;
2. the *prediction correctness* (correct or wrong prediction made by the ML model, given the labeled data).

We refer to the explanation together with a prediction as a *decision*.

Dependent (outcome) variables, metrics and hypothesis formulation

People are known to process contrastive explanations and apply counterfactual thinking in day-to-day life [180]. As a result, a contrastive explanation will be more intuitive and therefore arguably

more valuable to a user seeking an explanation [123]. To establish whether this preference for contrastive explanations over non-contrastive ones and no explanations holds for ML, we define this hypothesis in terms of the null-hypothesis (stating there is no effect):

H₀1 *The means for general preference are the same for all explanation methods.*

Transparency & trust. While this general preference can be used as a general argument for the usage of contrastive explanations, it does not provide us with insights into *why* it is preferred, and whether this reason for preference corresponds to our goals of transparency and trust. There are a multitude of reasons for preferring one explanation over another, corresponding to the goal of the explanation at hand. In the validation in Chapter 6 we have shown that the explanations provided are truthful to the model it aims to explain (i.e., transparent). However, in order to assess whether the contents of the explanation are also interpretable to users we consider the perceived transparency [30, 143]. Transparency can facilitate perceived understanding of the model and thereby trust [74]. We hypothesize that explanations are perceived as more transparent than providing no explanation:

H₀2 *The mean preferences for perceived transparency are the same for all explanation methods.*

In order for people to apply ML models in real-world tasks, they need to trust that it will perform well and understand beforehand when it will not perform well [43, 74, 110, 123]. Verification that the model uses sensible inferences for decision making can ensure reliability, robustness and safety when it is applied on a real-world task [43, 160]. However, trust is a long-term relationship between the user and a system that develops over time through interactions [57, 143]. To measure how the types of explanations might facilitate trust, we decompose trust into variables according to Cramer et al. [30] and Tintarev and Masthoff [184].

In evaluating trust, we look for the characteristics that distinguish contrastive explanations from non-contrastive ones. Firstly, we expect that contrastive explanations are preferred in terms of the informativeness of the features used in the explanation [30]. Similar to human explanatory behavior, contrastive explanations only explain a decision in terms of features used to distinguish fact from foil. Second, trust is strongly tied to the perceived competence of a system [30]. A more competent system, i.e. one with a better alignment with their own decision-making process, will lead to higher acceptance of its decisions and general acceptance of the system [30]. Moreover, Pu and Chen [142] found when the system is perceived as more competent, users are more willing to return to the system. This again stems from the premise that contrastive explanation is inherent to human explanation and therefore the preferred choice in terms of alignment with a users' own decision-making process. We capture these two dependent variables in the following null hypotheses:

H₀3 *The mean preferences for informativeness are the same for all explanation methods.*

H₀4 *The mean preferences for alignment are the same for all explanation methods.*

Effect of transparency & trust on general preference and persuasiveness. If there indeed is a preference for contrastive explanations over non-contrastive ones and non explanation, we additionally want to establish whether this preference also leads to the general preference of this type of explanation. Decisions that have a higher perceived transparency, informativeness and decision-making alignment are expected to be generally preferred. We establish their associations using hypotheses H_05 , H_06 and H_07 :

H_05 *Decisions that are perceived as more transparent are generally preferred.*

H_06 *Decisions that are perceived as more informative are generally preferred.*

H_07 *Decisions that are perceived as aligning well with their own decision-making are generally preferred.*

Moreover, this preference in terms of transparency in trust that leads to an increased system and decision acceptance is a form of persuasion: the system has the ability to persuade the user to agree with its decisions [184]. We therefore hypothesize that a people who find a decision more transparent, informative and aligning better with their own decision making are also more likely to act upon this decision—viewing it as more persuasive. H_08 through H_010 test this relationship. In addition, this persuasiveness is also strongly tied to a general preference as people who are more satisfied with an explanation are also more likely to act upon it [184]:

H_08 *Decisions that are perceived as more transparent do not cause explanations to be perceived as more persuasive.*

H_09 *Decisions that are perceived as more informative do not cause explanations to be perceived as more persuasive.*

H_010 *Decisions that are perceived as aligning well with their own decision-making do not cause explanations to be perceived as more persuasive.*

H_011 *There is no relationship between preference and persuasiveness of decisions.*

Agreement. While contrastive explanations may perform well in persuading the user to agree with its decisions, previous research has shown that explanations may lead to insensitivity to reliability of automated systems. Bussone et al. [22] found that placing too much trust in the system may lead to over-reliance, where a user agrees with the system even if it makes a faulty decision. For our healthcare context, where decision may affect the short-term and long-term health of a T1DM patient, this is undesirable behavior. Similarly, providing no explanations or wrong explanations can lead to users ignoring incorrect suggestions (under-reliance of the system). In addition, appropriate reliance improves system' perceptions and intention to use the system [58, 143]. To the best of our knowledge, no current studies have comparatively evaluated contrastive and non-contrastive explanations. Therefore, we aim to test how contrastive and non-contrastive explanations compare in terms of an appropriate level of reliance:

H_012 *There is no association between the explanation method and decision agreement.*

H_013 *There is no association between the prediction correctness and decision agreement.*

Overview. All independent variables (IVs), dependent variables (DVs) and hypotheses are summarized graphically in Figure 7.3. We test directly for three *value* judgments (perceived transparency, informativeness and alignment) and see their effect on the *perceptions* (preference and persuasiveness). In addition, we determine the effect of the explanation method and prediction correctness on willingness to agree with decisions made by the system (reliance).

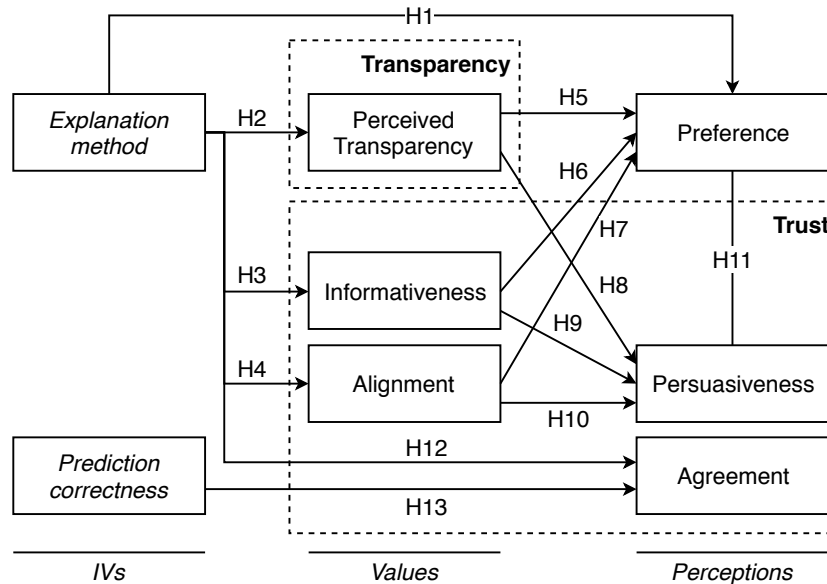


Figure 7.3 Theoretical framework

Experiment design, participants and objects

Each participant is assigned each factor.

Sample. The experimental participants were recruited on *Amazon Mechanical Turk* (MTurk).¹ All participants were at least 18 years old (in line with MTurks' Participation Agreement). Participants received a small monetary reward in case of questionnaire completion. In addition, they were provided a small bonus when performing well on a test quiz on the scenario and type I diabetes mellitus (T1DM) management. Questionnaires were set-up to disallow repeat workers. In case of participant drop-out—i.e., a participant not finishing the questionnaire—, MTurk recruited new participants and the participant will not receive their reward. In total, 148 participants took part in the experiment—of which 121 completed it in full. None of the drop-out participants completed a part beyond the demographics and test quiz, and are therefore excluded from further analysis. Table 7.1 describes the participant demographics and their familiarity with T1DM (management).

Experimental objects. We define the procedure for constructing the questionnaires as follows. For *part 'Preference'* participants will receive ten (10) agenda items that broadly represent the un-

¹ <https://www.mturk.com/>

Table 7.1 Participant demographic and T1DM management familiarity characteristics ($N = 121$)

Demographics			Diabetes management		
Characteristic	<i>N</i>	%	Characteristic	<i>N</i>	%
Age			Familiarity with diabetes management (self-rep.)		
18–24	2	1.7	<i>Mean</i>	6.14	
25–34	52	43.0	<i>SD</i>	3.02	
35–44	37	31.4			
45–54	20	16.5	Family history of diabetes		
55–64	6	5.0	Yes	85	70.2
65+	3	2.5	No	33	27.2
Sex			<i>Unknown</i>	3	2.5
Male	64	52.9			
Female	57	47.1	Occupation/study related to healthcare		
Region			No	106	87.6
Africa	1	0.8	Yes, studying medicine	2	1.7
Americas	83	68.6	Yes, direct occupation*	10	8.3
Asia	36	29.8	Yes, indirect occupation [†]	3	2.5
Oceania	0	0.0			
Education (highest completed)					
High school	33	28.1			
Undergraduate	70	57.9			
Graduate	17	14.0			
English reading/writing					
Not well at all	0	0.0			
Not very well	0	0.0			
Well	7	5.8			
Pretty well	11	9.1			
Very well	103	85.1			

* e.g., nurse, doctor; [†] e.g., pharmaceutical

derlying data set, and for each explanation type pair (i.e., no explanation–contrastive explanation, contrastive explanation–non-contrastive explanation, no explanation–non-contrastive explanation) they will be asked to determine their preferences. This totals to 30 pairwise comparisons. *Part ‘Agreement’* will use six (6) different agenda items (three of which have a correct prediction, and three have an incorrect prediction). They are shown each agenda item three times, including one of three types of explanation types (no explanation, contrastive explanation, non-contrastive explanation) and they are asked whether they agree or disagree with the prediction made.

Data collection

Data was collected with an anonymous online questionnaire implemented in Survey Monkey.² For reference, questionnaire materials are included in Appendix D.

² <https://www.surveymonkey.com/>

Instrumentation. Before the experiment, each participant is asked to fill in a demographic questionnaire, which contains questions about sex, age group, region, highest-attained education, level of English and their background and previous experience with the diabetes management subject area. Next, we generate the questionnaire which forms the main body for parts ‘Preference’ and ‘Agreement’ as described in the experimental objects earlier in this section.

Procedure. The diagram in Figure 7.4 outlines the procedure for the online experiment. Table 7.2 describes the variables measured per activity. Participants are first provided with a general introduction about the experiment set-up and use case and asked to accept to the terms of an informed consent. Next, they are given with a demographic questionnaire. Participants are trained with an elaborate description of the scenario of the experiment—a case for an adolescent type I diabetes mellitus (T1DM) patient using an intelligent app to predict whether she will have a hypo, hyper or neither based on the current time, the food she intends to eat, exercises performed and planned, and the prior health status—and how T1DM (management) works. Participants are then tasked to fill in a test quiz about the scenario and T1DM (management), and are provided with the scenario & T1DM explanation page again if they were uncertain about responses given in the test.

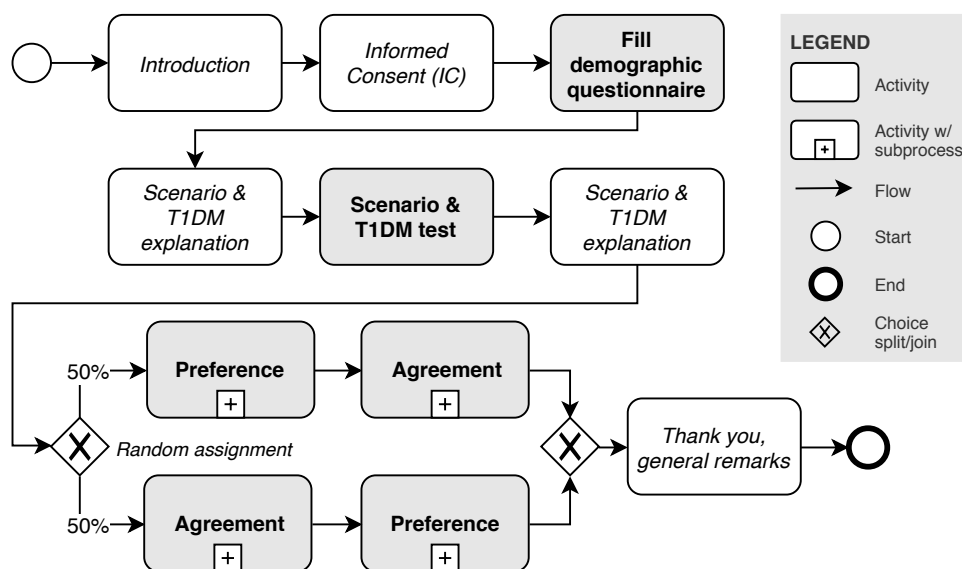


Figure 7.4 Experiment procedure overview

The body of the experiment constitutes two parts: ‘Preference’ and ‘Agreement’. All participants perform both parts, but are randomly assigned a part to perform first to mitigate order effects. Figure 7.5 describes each part in more detail.

In the ‘Preference’ part they are given 30 pairwise comparisons and asked to determine their preferences. They are then asked for their general perceptions of the cases in the pairwise comparisons. In part ‘Agreement’ they are provided with 18 cases and asked for each case whether they agree or disagree with the provided decision given the data and provided explanation and

decision. For both parts, they are given information about the task, and an example cases to familiarize themselves with the task. In addition, after executing the task for both parts they are asked to perform the task one more time, and provide the reason for the choices they made. Finally, participants are thanked for their participation and given a form and contact information for further remarks.

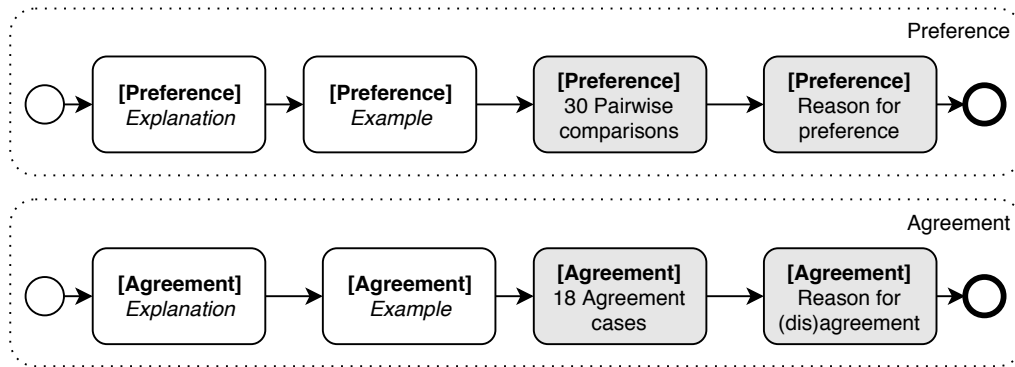


Figure 7.5 Experiment procedure for parts ‘Preference’ and ‘Agreement’

Part ‘Preference’ aims to test hypotheses H1–H11, while part ‘Agreement’ aims to test hypotheses H12 and H13.

Table 7.2 Measured variables per activity

Activity	Measured variables
<i>Part ‘Preference’</i>	
Pairwise comparison	Preference, perceived transparency, informativeness, alignment, persuasiveness
Reason for preference	–
<i>Part ‘Agreement’</i>	
Over-reliance / under-reliance	Agreement
Reason for (dis)agreement	–

7.3 Results

This section reports the quantitative results that are obtained from the participants, and describes the analyses performed on the results. The analyses were executed using Python 3.6 using packages SciPy, statsmodels and choix.

Descriptive statistics

Participants performed well in the test quiz, with a mean score of 80% ($N = 121$, $SD = 0.21$, $Min = 0.22$, $Max = 1.00$).

Figure 7.6 summarizes the preferences for the pairwise comparisons, for each of the five statements (persuasiveness, alignment, informativeness, perceived transparency and preference). On

average, contrastive explanations ($M = 715.2$, $SD = 25.9$) were preferred over no explanations ($M = 307.6$, $SD = 24.4$). In 187.2 cases on average there was no preference for either ($SD = 15.2$). Non-contrastive explanations ($M = 623.2$, $SD = 27.9$) were also preferred over no explanations ($M = 375.6$, $SD = 27.2$)—but lead to more ties ($M = 211.2$, $SD = 15.7$). Contrastive explanations ($M = 569.8$, $SD = 28.7$) were somewhat preferred over non-contrastive explanations ($M = 368.8$, $SD = 13.1$). Here, on average 271.4 judgments lead to a tie ($SD = 18.4$).

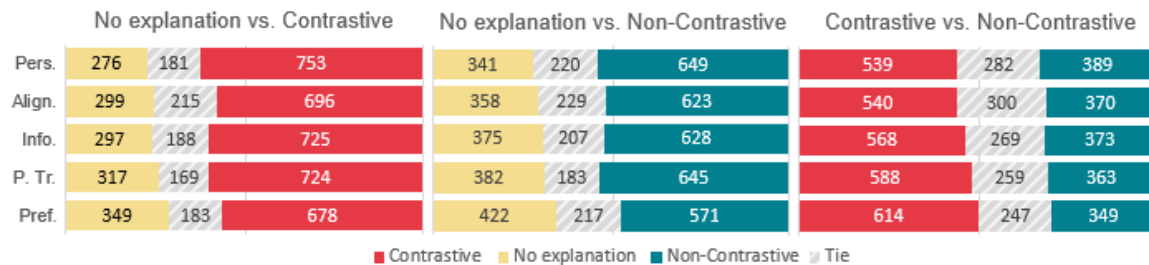


Figure 7.6 Overview of pairwise preferences ($N = 1210$), where *tie* indicates no preference

Table 7.3 details the number of times participants agreed or disagreed with the decisions made for the agenda items, grouped by explanation method and decision correctness. It shows that participants were on average twice as likely to agree with a statement ($M = 242.0$, $SD = 4.6$) rather than disagree ($M = 121.0$, $SD = 4.6$).

Table 7.3 Overview of agreement per explanation method and decision correctness ($N = 2178$)

Explanation method	Decision correctness	Agreement	
		Agree	Disagree
No explanation	True	236	127
	False	241	122
Contrastive	True	249	114
	False	237	126
Non-Contrastive	True	243	120
	False	246	117

Data set preparation

The data set contains one row per participant of all experiment parts. For the preference part only preference judgments for agenda items 0 through 9 were considered (the ten given during the 30 pairwise comparisons), and for the agreement part only the 18 (dis)agreement decision responses of the six agenda items (10–15) were considered. The reasons for each part are considered separately.

For each of the preference judgments, the pairwise comparisons were transformed into a set of win-loss pairs per agenda item. Ties were equally distributed as wins across both explanation methods.³ They were then transformed into a coefficient estimate using the *Bradley-Terry* method

³ No current Bradley-Terry method currently is able to handle tie results. As such, we take this naive approach to

[17], which indicates the probability of that explanation method winning over the other two methods.

Hypothesis testing

All hypotheses were tested with a critical value of $\alpha = 0.05$ with Bonferroni correction.

Preferences. First, we aim to verify which *explanation method* is preferred generally, and in terms of perceived transparency, informativeness and alignment. Table 7.4 shows the mean Bradley-Terry estimate of each explanation method per dependent variable (DV), and the results of a one-way ANOVA on the Bradley-Terry estimates of each agenda item. The results show that for each DV there is a significant effect of the explanation method on the preference of decisions in terms of general preference, perceived transparency, informativeness, and decision-making alignment. In each case, contrastive explanations are generally preferred over both non-contrastive explanations and no explanation. Non-contrastive explanation are typically preferred over no explanation. Based on the significant differences in the ANOVA results (all with $p < 0.001$), we reject hypotheses H_01 through H_04 .

Table 7.4 Mean Bradley-Terry estimates (and resulting rank & standard deviation) per explanation method, and the reported differences in estimates for preference, perceived transparency, informativeness and alignment ($N = 10$)

Hypothesis	DV	Bradley-Terry estimate (rank, <i>SD</i>)			ANOVA	
		No explanation	Contrastive	Non-Contr.	<i>F</i> -stat.	Sig.
H_01	Preference	-0.27 (3, 0.13)	0.35 (1, 0.19)	-0.07 (2, 0.24)	26.18	<0.001
H_02	P. Transparency	-0.39 (3, 0.11)	0.37 (1, 0.17)	0.02 (2, 0.22)	48.68	<0.001
H_03	Informativeness	-0.39 (3, 0.11)	0.36 (1, 0.18)	0.03 (2, 0.28)	42.77	<0.001
H_04	Alignment	-0.38 (3, 0.15)	0.33 (1, 0.15)	0.05 (2, 0.24)	37.10	<0.001

Post-hoc analysis using the Tukey HSD test indicated at $\alpha < 0.004^4$ that contrastive explanations were always preferred over no explanation, and contrastive explanations were preferred over non-contrastive explanations in terms of preference, perceived transparency and informativeness. Regarding decision-making alignment, contrastive explanations were not significantly preferred over non-contrastive explanations. Non-contrastive explanations were significantly preferred over no explanation, except for general preference.

Effect of value preferences on perceptions. We aim to verify whether *general preference* and *persuasiveness* depend on the three values *perceived transparency*, *informativeness* and *alignment*. In addition, we test the correlation between preference and persuasiveness. Table 7.5 reports the correlations between the ranks given by participants to the (DVs) dependent variables using the Kendall rank correlation coefficient τ . There was a strong ($r_\tau > 0.7$) positive correlation for all

handling ties, and plan to leave extending such models (e.g., using the Davidson model [37]) for future work.

⁴ $\alpha = 0.05$ Bonferroni correct with $3 \times 4 = 12$ hypotheses tests.

seven hypotheses tests. Thus, we reject null-hypotheses H_05 through H_011 . This indicates that explanations that are preferred in terms of perceived transparency, informativeness and alignment with their own decision-making are also generally performed, and perceived as more persuasive. Moreover, general preference is strongly correlated with the persuasiveness of an explanation.

Table 7.5 Kendall's correlation statistics for hypotheses H_05 – H_011 ($N = 3630$)

Hyp.	DV1	DV2	Kendall's r_τ	Sig.
H_05	P. Transparency	Preference	0.781	<0.001
H_06	Informativeness	Preference	0.754	<0.001
H_07	Alignment	Preference	0.800	<0.001
H_08	P. Transparency	Persuasiveness	0.755	<0.001
H_09	Informativeness	Persuasiveness	0.778	<0.001
H_010	Alignment	Persuasiveness	0.792	<0.001
H_011	Preference	Persuasiveness	0.774	<0.001

Agreement. Finally, we aim to assess whether willingness to agree with a decision depends on the explanation method and the prediction correctness. A chi-square test of independence was performed to examine the relation between *explanation method* and *agreement*. The relation between these variables was not significant, $\chi^2(2, N = 2178) = 0.01, p = 0.892$. Moreover, a chi-square test of independence was conducted to examine the association between *prediction correctness* and *agreement*, which also showed no significant relation: $\chi^2(3, N = 2178) = 0.48, p = 0.785$.

Reasons for preferences and agreement

Besides, we also used the experiment to gain some qualitatively insights into reasons for why decisions were preferred, and why people tended to (dis)agree with decisions.

Preference. The reasons given for the preference one type of explanation over the other are shown in Figure 7.7, where the colors indicate the type of explanation preferred. For determining their preferences, participants looked at the amount of information given by a decision (i.e., explanation–prediction pair), their understanding of the information and its alignment and informativeness. Longer length decisions were preferred over shorter length explanations. Regardless of the general preference for contrastive explanations, the comparison made in these explanations between two predictions was only cited as a reason for preference by 17 (14.0%) of participants.

In general, there is no difference between reasons for the explanation types. However, it should be noted that non-contrastive explanations were also seen as making a comparison between predictions by multiple participants—even though this is the core explicit concept making contrastive explanation unique.

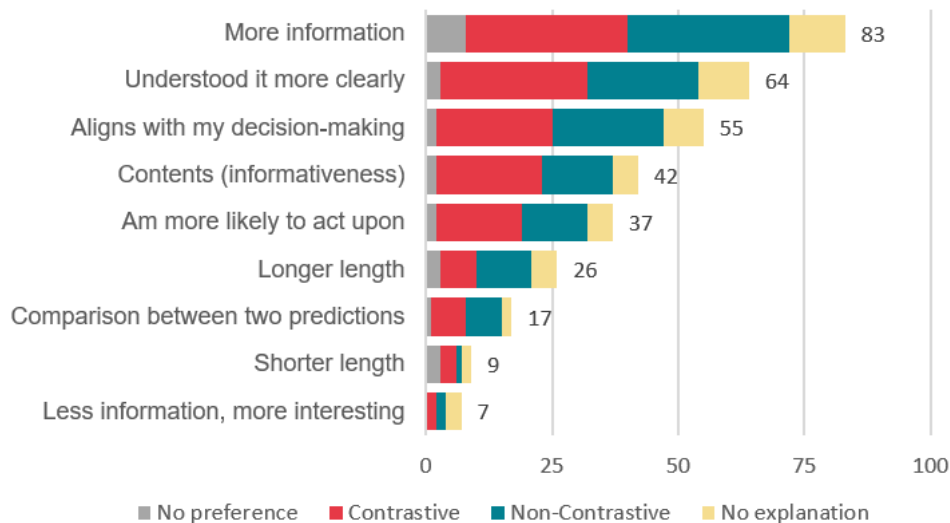


Figure 7.7 Main considerations made to determine preference for a decision

Agreement. The reasons for agreeing/disagreeing were given in free-form text. Typically, the reasons that were cited were the contents of the agenda item and decision. The given agenda item was equal to the one shown in Figure 7.2. Fifty-seven (57) participants cited the fact that she had a hyper in the past hour as a consideration. Fifty-one (51) looked at the food and the amount of carbohydrates (CHO) in the food. Twenty-eight (28) considered the skating exercise. Six participants (6) expected a different prediction.

7.4 Discussion

Threats to validity

In this section we discuss the identified threats to the validity of this experiment, including how threats were mitigated. The threats are grouped into four different types of validity based on Wohlin et al. [199].

Conclusion validity is concerned with the relationship between treatment and outcome. The **sample size** of 121 was large enough to conduct the experiment. **Heterogeneity of subjects** is a possible threat to this research, as some participants had previous experience or reported familiarity with T1DM management. This might influence the distribution of results. To reduce this threat, all participants were given an equal training of the required knowledge to infer their own decisions for the provided agenda items. **Hypothesis fishing** was avoided by determining the hypotheses before conducting the experiment. The **reliability of measures** was threatened as to minimize the task difficulty for the comparison a number of constructs had to be reduced to a single sentence, even though they comprised multiple statements in studied literature. It was reduced by using appropriate wording to draw conclusions upon for contrastive explanations.

Internal validity considers how the experiment minimizes bias. Threats introduced by the effects of **maturation**—i.e., ordering effects—were avoided by randomly assigning half of the participants

to perform the 'Preference' part first and half performing the 'Agreement' part first. In addition, the pairwise comparisons and (dis)agreement cases were shown in random order. However, the reason for each part may have been affected by maturation as this was always shown at the part end. The threat of **undermotivation** was minimized by providing a small monetary reward, and an additional positive reward when performing well on a test quiz of the scenario and T1DM (management). **Mortality** was reduced by allowing Mechanical Turk to re-recruit participants in case of drop-out.

Construct validity involves the relation between theory and observation. **Hypothesis guessing** was mitigated by determining a theoretical framework and the hypotheses before experiment execution. This was enforced by the experiment having to pass through the ethical committee of TNO before conducting the experiment. Potential **confounding variables** for the statistical tests were acquired using demographic and T1DM (management) familiarity questions. The effect of participant perceptions of machine-generated explanations was reduced by not informing them that the explanations were extracted automatically. **Experimenter expectancies** were reduced by letting multiple researchers validate the experiment design and questions before execution.

External validity threats reduce generalization of the findings. **Subject population** is a possible threat, as the participants predominantly resided in the Americas and Asia. This introduces cultural differences, which may reduce the generalizability of these findings to the first region of implementation of the tool: Europe. In addition, these findings are difficult to generalize to health practitioners, and diabetes patients and their caregivers—who have a more intricate understanding of T1DM (management). We mitigated the threat of **interaction of setting and treatment** by using a T1DM simulator that is able to generate realistic blood glucose levels for virtual subjects, operating on actual T1DM patient characteristics. However, tweaks to this simulator to generate a data set fit-for-purpose may have introduced a toy-problem not completely representing all facets of T1DM management in a realistic manner.

Inferences

This empirical experiment has been able to show that contrastive explanations were generally preferred over non-contrastive explanations *and* no explanations, and also preferred in terms of perceived transparency, informativeness and alignment with their own decision-making. Contrastive explanations were also found to be more persuasive. Moreover, association tests indicate that this general preference is correlated with perceived transparency, informativeness and alignment, and explanations that were more transparent, informative and aligned well also were more persuasive.

Even though these findings are difficult to generalize to the intended users of the T1DM management use-case, they indicate a more broad generalization beyond specific contexts that explanations that are transparent, understood more clearly, and contain information that is valuable and more similar to the contents people would use in explanation are also generally preferred and thereby more persuasive. Contrastive explanations are a valuable means for creating explanations that are preferable in all of these criteria.

Regarding under-reliance and over-reliance, this study has been unable to show the effects of prediction correctness and explanation type on the willingness of participants to agree with decisions for the T1DM management use-case. This is possibly due to the fact that in the reasons for their agreement, participants showed the ability to make their own inferences regarding the agenda items regardless of the decision. Moreover, participants were shown the same agenda item three times (each time with a different explanation, but the same prediction)—which may have affected the willingness to agree.

7.5 Conclusion

The purpose of this user experiment was to establish whether contrastive explanations were preferred over non-contrastive explanations and no explanations in terms of general preference, transparency and trust. Contrastive explanations have been brought forward as an intuitive and human-like means to limit an explanation to only the key differences [109]. With an empirical experiment on 121 participants, our study has shown that indeed contrastive explanations were aligned better with the user's decision-making process (*alignment*) and his/her considered features (*informativeness*), and that they were also perceived as more understandable (*transparency*). The preference regarding these three values also lead to an increased general preference (*preference*) and willingness to act upon the decision (*persuasiveness*). The experimental evaluation did not confirm that these methods affected the agreement with a decision—regardless of whether the decision was correct—, potentially due to maturation effects of the participants.

The implications of these findings are in line with the theorized natural tendency of humans to perform contrastive explanations as hypothesized in philosophy of science and psychology literature (e.g., [109, 117, 180]) and as argued in interpretable machine learning (iML) literature (e.g., [110]). In addition, we found that this preference can increase the trust and convincingness of an explanation. However, it remains unclear whether this also holds for the caregivers and in particular children in the T1DM management case.



CONCLUSION AND OUTLOOK

8 CONCLUSION AND OUTLOOK

This study investigated how contrastive explanation can be defined and operationalized in machine learning (ML), and which benefits it provides over non-contrastive explanations for the person the explanations are provided to. The study was structured around the following research question:

RQ *How can contrastive explanation improve the explanatory capability of machine learning methods?*

The main research question was investigated using five subquestions. This chapter concludes the study by answering the research questions in Section 8.1, and we present directions for future research in Section 8.2.

8.1 Conclusion

Subquestions

This section summarizes the main findings for each subquestion (SQ).

SQ1 *What are current approaches to automatically acquire explanations from machine learning systems?*

In Chapter 4 we systematically reviewed 84 interpretable machine learning (iML) methods, that show that there is a manifold of means for improving interpretability in machine learning. They cover a wide range of approaches (pedagogical, decompositional and transparent), and are able to provide local and/or global explanations using a variety of explanatory representations (e.g., rule-based, feature importance, explanation by example or saliency). While many focus on supervised learning (in particular classification), in recent years approaches have also started to cover reinforcement learning. For contrastive explanations, *local*, *pedagogical*, *rule-based* explanations seem most beneficial. They are model-agnostic, and able to provide explanations in a representation that is natural to humans.

SQ2 *How to automatically acquire decision rule explanations from machine learning systems?*

For our approach (see Chapter 5), we locally sample or generate data points with labels determined by the model. This data is used as training data for a local rule-based model, such as a decision tree. The decision rule explanations can then be directly extracted from this decision tree. For explanations of individual data points, neighborhood data is weighed based on the similarity to the data point in question—ensuring that the local explainer is faithful in close

proximity to the model. The decision rule explanation can then be extracted by applying this model to the questioned data point and combining the traversed rule evaluations.

SQ3 *How can we determine the foil for a required explanation?*

The contrast class contains the fact and all foils (potential alternative outcomes) under consideration. For machine learning, this depends on the machine learning task (elaborated further in Chapter 3). For example, for classification this contrast class are the classes, while for regression it can be a different value, binned values, or a statistical norm. In Chapter 5 we argued that the foil of interest to the explainee can either be (i) explicitly given in a static interaction, (ii) inferred based on model characteristics (e.g., class probabilities), or (iii) learned from interaction of the explainee with the explanation system.

SQ4 *How can we generate human-understandable contrastive explanations?*

After determining an appropriate foil, the explanation task can be reduced to a foil-versus-rest binary classification problem. On the weighed neighborhood data a local transparent model or perturbation can be used to define the contrastive explanation counterfactually. In this study, we propose *Foil Trees* (Chapter 5) as such a local transparent model, where we form an explanation by taking the difference between the rule set that holds for the fact, and the rule set that holds for a selected foil-leaf. By providing explanations in the input space, we are able to yield meaningful contrastive explanations for the end-user for any questioned data point and for an arbitrary foil.

SQ5 *What is the quality of the contrastive explanations?*

Quantitative validation (Chapter 6) showed that Foil Trees are able to accurately mimic the decision boundaries of the model it aims to explain (94% *fidelity*), generalizes well on unseen data (88% *accuracy*), provides 78% shorter explanations than their non-contrastive counterparts (*mean length* of 1.19 over 5.37) and does this all in real-time (60ms on average per explanation). This demonstrates the feasibility, objective transparency and generalizability of our contrastive explanation approach.

In a user experiment (Chapter 7) we evaluated the explanation contents in order to establish how contrastive explanations, non-contrastive explanations and no explanation are perceived in terms of *general preference*, *transparency* and *trust*. For agenda items in a diabetes management case using explanations generated using Foil Trees, participants ($N = 121$) were tasked with determining their pairwise preferences regarding three value judgments (perceived transparency, informativeness & alignment) and two perception judgments (general preference & persuasiveness), and whether they agreed with the decision made given an explanation type. We found that contrastive explanations are preferred over non-contrastive explanations and no explanation in terms of *transparency* (understandability), *informativeness* of contents and *alignment* with their own decision-making. This preference lead to an increased *general preference* and *willingness to act upon the decision*. The empirical evaluation did not confirm that these methods affected the agreement with a decision made—regardless of whether the prediction was correct.

Research question

Finally, we use insights and inferences from the subquestions to define an answer to our main research question:

RQ *How can contrastive explanation improve the explanatory capability of machine learning methods?*

Contrastive explanations provide an intuitive, human-like way to reduce the number of elements required to form an explanation. This is done by setting a contrast between the actual output (fact) and the output of interest (foil). For supervised machine learning tasks, we are able to generate human-understandable contrastive rule-based explanations using a locally faithful, model-agnostic explanation method: *Foil Trees*. For any of the potential foils of the machine learning model, we can extract in real-time meaningful explanations in terms of model inputs for any questioned data point and for any arbitrary foil (either given, inferred from model characteristics or learned from interaction with the explaineé). These contrastive explanations are truthful to the model they aim to explain, generalize well to unseen data and provide considerably shorter explanations than non-contrastive ones. Moreover, they are deemed more understandable (transparent), informative and aligned better with individual decision-making behavior than non-contrastive explanations. These values lead to an increased general preference and willingness of end-users to act upon the decision.

8.2 Future Directions

During this study, we have identified interesting directions for further research.

Approach generalization: new domains, new machine learning types

Previous iML methods have successfully proven to be able to provide generalized model-agnostic explanations using a single explainer (e.g., Ribeiro et al. [148, 149]), that can explain tabular, text, and image data. ContrastiveExplanation (CE) currently supports tabular data explanations and has rudimentary support for images, but we aim to extend this to text data and image data. For handling the large search-space—caused by the high number of features in text and especially image data—we intend to explore extracting meaningful high-level features (e.g., using an autoencoder neural network), to investigate explaining using an efficient directed explanation through foil-sensitive (i.e., targeted) data point perturbation [39], and we consider to investigate the possibility of explaining using a domain different than the input (e.g., text explanations for images) [47, 72].

Moreover, while we currently are able to explain classification and regression analysis tasks, we intend to transfer our contrastive explanation approach to *reinforcement learning*. This introduces a time component, as well as a need to adjust sampling methods to account for only the possible actions in a state to be considered as possible next actions.

Adaptive Contrastive Explanations (ACE)

Adaptive Contrastive Explanations (ACE) extend *Foil Trees* and the ContrastiveExplanation (CE) tool by adapting the contrastive explanations to the explainee. It will be able to learn the foil from previous interactions with ACE, or from explainees similar to the one asking for an explanation. In addition, we envision that it will be able to learn the appropriate foil-leaf selection strategy, such that the right type of explanation is provided to the explainee based on user characteristics and situational demands. In addition, this requires a meaningful indicator for measuring the optimal explanation, as well as improved end-user interaction with a more sophisticated interface.

Local robustness and validation

Recent studies in iML has shown that iML methods are sensitive to small changes in input values of the data point it aims to explain, and the size of the local neighborhood considered for explanation [4, 101]. This effect has most strongly been observed for model-agnostic approaches [4]. In future work we intend to assess the *locality* (i.e., the relative size of the region the transparent model is able to explain) and *robustness* (i.e., the effect of small changes in input change on the explanation, especially around decision boundaries) of Foil Trees in order to improve the stability and generalizability of our approach across data sets. Moreover, we intend to change the contrastive explanation problem from a binary classification problem into probabilities that a data point is a foil or not-foil, so that explanation certainty can be increased by explaining further away from the decision boundary between the foil decision region and the rest of the data points.

Quantitative and qualitative insights

The quantitative and qualitative data collected in this study provide many opportunities for further analysis. Interesting trajectories to pursue are the effect of outliers on model performance and explanation contents, determining the effect of various demographic and diabetes knowledge characteristics on preferences and willingness to agree in the user experiment, and obtaining quantitative data for the diabetes management use-case data set to allow for data analysis with the integrated quantitative and experiment data set. In addition, further empirical evaluation should consider the effect of user expertise in a subject area on their willingness to agree, compare contrastive explanations to other explanation methods (e.g., feature importance), and analyze decisions for multiple use-cases and multiple model types. Lastly, we intend to validate whether the findings regarding the perceptions of contrastive explanations and their preference over non-contrastive explanations also hold for caregivers in the PAL use-case.

BIBLIOGRAPHY

- [1] A. Abdul, J. Vermeulen, D. Wang, B. Y. Lim, and M. Kankanhalli, "Trends and Trajectories for Explainable, Accountable and Intelligent Systems: An HCI Research Agenda," in *International Conference on Human Factors in Computing Systems (CHI '18), Proceedings*, 2018.
- [2] J. Adebayo and L. Kagal, "Iterative Orthogonal Feature Projection for Diagnosing Bias in Black-Box Models," *arXiv preprint*, 2015. [Online]. Available: <http://arxiv.org/abs/1611.04967>
- [3] P. Adler, C. Falk, S. A. Friedler, T. Nix, G. Rybeck, C. Scheidegger, B. Smith, and S. Venkatasubramanian, "Auditing Black-Box Models for Indirect Influence," *Knowledge and Information Systems*, vol. 54, no. 1, pp. 95–122, 2018.
- [4] D. Alvarez-Melis and T. S. Jaakkola, "On the Robustness of Interpretability Methods," in *2018 ICML Workshop on Human Interpretability in Machine Learning (WHI)*, 2018, pp. 66–71.
- [5] R. Andrews, J. Diederich, and A. B. Tickle, "Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks," *Knowledge-Based Systems*, vol. 8, no. 6, pp. 373–389, 1995.
- [6] M. G. Augasta and T. Kathirvalavakumar, "Reverse Engineering the Neural Networks for Rule Extraction in Classification Problems," *Neural Processing Letters*, vol. 35, no. 2, pp. 131–150, 2012.
- [7] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek, "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation," *PLoS ONE*, vol. 10, no. 7, 2015.
- [8] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Mueller, "How to Explain Individual Classification Decisions," *Journal of Machine Learning Research*, vol. 11, pp. 1803–1831, 2010.
- [9] E. Barnes, "Why P rather than Q? The Curiosities of Fact and Foil," *Philosophical Studies*, vol. 73, no. 1, pp. 35–53, 1994.
- [10] O. Bastani, C. Kim, and H. Bastani, "Interpreting Blackbox Models via Model Extraction," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1705.08504>
- [11] A. Bibal and B. Frénay, "Interpretability of Machine Learning Models and Representations: An Introduction," *European Symposium on Artificial Neural Networks (ESANN)*, no. April, pp. 27–29, 2016.
- [12] J. Bieger, K. R. Thorisson, and B. Steunebrink, "Evaluating Understanding," in *IJCAI Workshop on Evaluating General-Purpose AI*, 2017.
- [13] J. Bien and R. Tibshirani, "Prototype Selection for Interpretable Classification," *The Annals of Applied Statistics*, vol. 5, no. 4, pp. 2403–2424, 2011.
- [14] R. Binns, M. Van Kleek, M. Veale, U. Lyngs, J. Zhao, and N. Shadboldt, "'It's Reducing a Human Being to a Percentage'; Perceptions of Justice in Algorithmic Decisions," in *ACM Conference on Human Factors in Computing Systems (CHI'18), Proceedings*, 2018.
- [15] O. Biran and C. Cotton, "Explanation and Justification in Machine Learning: A Survey," in *International Joint Conference on Artificial Intelligence (IJCAI), Proceedings*, 2017, pp. 8–13.
- [16] O. Boz, "Extracting Decision Trees From Trained Neural Networks," in *8th SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02), Proceedings*, 2002, pp. 456–461.
- [17] R. A. Bradley and M. E. Terry, "Rank Analysis of Incomplete Block Designs: The Method of Paired

- Comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.
- [18] S. Brinkkemper and S. Pachidi, "Functional Architecture Modeling for the Software Product Industry," *Software Architecture*, vol. 6285, pp. 198–213, 2010.
- [19] C. Buciluă, R. Caruana, and A. Niculescu-Mizil, "Model Compression," *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06), Proceedings*, pp. 535–541, 2006.
- [20] D. Budgen and P. Brereton, "Performing Systematic Literature Reviews in Software Engineering," in *28th International Conference on Software Engineering, Proceedings*, 2006, pp. 1051–1052.
- [21] J. Burrell, "How the Machine 'Thinks': Understanding Opacity in Machine Learning Algorithms," *Big Data & Society*, vol. 3, no. 1, 2016.
- [22] A. Bussone, S. Stumpf, and D. O'Sullivan, "The Role of Explanations on Trust and Reliance in Clinical Decision Support Systems," in *2015 International Conference on Healthcare Informatics*, vol. 44. IEEE, 2015, pp. 160–169.
- [23] E. Carrizosa, A. Nogales-Gómez, and D. R. Morales, "Strongly Agree or Strongly Disagree?: Rating Features in Support Vector Machines," *Information Sciences*, vol. 329, pp. 256–273, 2016.
- [24] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad, "Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission," in *21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15), Proceedings*, 2015, pp. 1721–1730.
- [25] S. Chakraborty, R. Tomsett, R. Raghavendra, D. Harborne, M. Alzantot, F. Cerutti, M. Srivastava, A. Preece, S. Julier, R. M. Rao, T. D. Kelley, D. Braines, M. Sensoy, C. J. Willis, and P. Gurram, "Interpretability of Deep Learning Models: A Survey of Results," in *IEEE Smart World Congress: DAIS - Workshop on Distributed Analytics Infrastructure and Algorithms for Multi-Organization Federations*. IEEE, 2017.
- [26] H. A. Chipman, E. I. George, and R. E. McCulloch, "Making Sense of a Forest of Trees," in *30th Symposium on the Interface, Proceedings*, 1998, pp. 84–92.
- [27] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," *arXiv preprint*, 2014. [Online]. Available: <http://arxiv.org/abs/1409.1259>
- [28] M. Colombo, "Experimental Philosophy of Explanation Rising: The Case for a Plurality of Concepts of Explanation," *Cognitive Science*, vol. 41, no. 2, pp. 503–517, 2017.
- [29] P. Cortez and M. J. Embrechts, "Opening Black Box Data Mining Models Using Sensitivity Analysis," in *IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Proceedings*. IEEE, 2011, pp. 341–348.
- [30] H. Cramer, V. Evers, S. Ramlal, M. Van Someren, L. Rutledge, N. Stash, L. Aroyo, and B. Wielinga, *The Effects of Transparency on Trust in and Acceptance of a Content-Based Art Recommender*, 2008, vol. 18, no. 5.
- [31] M. W. Craven and J. W. Shavlik, "Using Sampling and Queries to Extract Rules from Trained Neural Networks," *Eleventh International Conference on Machine Learning (ICML), Proceedings*, pp. 37–45, 1994.
- [32] —, "Extracting Tree-Structured Representations of Trained Neural Networks," *Advances in Neural Information Processing Systems (NIPS)*, vol. 8, pp. 24–30, 1996.
- [33] —, "Rule Extraction: Where Do We Go from Here?" University of Wisconsin Machine Learning Research Group, Tech. Rep., 1999.
- [34] C. Dalla Man, F. Micheletto, D. Lv, M. Breton, B. Kovatchev, and C. Cobelli, "The UVA/PADOVA Type 1 Diabetes Simulator: New Features," *Journal of Diabetes Science and Technology*, vol. 8, no. 1, pp. 26–34, 2014.

- [35] A. Datta, M. C. Tschantz, and A. Datta, "Automated Experiments on Ad Privacy Settings: A Tale of Opacity, Choice, and Discrimination," in *Privacy Enhancing Technologies, Proceedings*, 2015, pp. 92–112.
- [36] A. Datta, S. Sen, and Y. Zick, "Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems," in *2016 IEEE Symposium on Security and Privacy (SP 2016), Proceedings*. IEEE, 2016, pp. 598–617.
- [37] R. R. Davidson, "On Extending the Bradley-Terry Model to Accommodate Ties in Paired Comparison Experiments," *Journal of the American Statistical Association*, vol. 65, no. 329, pp. 317–328, 1970.
- [38] H. Deng, "Interpreting Tree Ensembles with inTrees," *arXiv preprint*, 2014. [Online]. Available: <http://arxiv.org/abs/1408.5456>
- [39] A. Dhurandhar, P. Chen, R. Luss, C. Tu, P. Ting, K. Shanmugam, and P. Das, "Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives," *arXiv preprint*, 2018. [Online]. Available: <http://arxiv.org/abs/1802.07623>
- [40] Diabetes Control and Complications Trial Research Group, D. Nathan, S. Genuth, J. Lachin, P. Cleary, O. Crofford, M. Davis, L. Rand, and C. Siebert, "The Effect of Intensive Treatment of Diabetes on the Development and Progression of Long-Term Complications in Insulin-Dependent Diabetes Mellitus," *The New England Journal of Medicine*, vol. 329, pp. 162–167, 1993.
- [41] P. Domingos, "Knowledge Discovery via Multiple Models," *Intelligent Data Analysis*, vol. 2, no. 3, pp. 187–202, 1998.
- [42] D. Doran, S. Schulz, and T. R. Besold, "What Does Explainable AI Really Mean? A New Conceptualization of Perspectives," *arXiv preprint*, 2017.
- [43] F. Doshi-Velez and B. Kim, "Towards A Rigorous Science of Interpretable Machine Learning," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1702.08608>
- [44] F. Doshi-Velez, M. Kortz, R. Budish, C. Bavitz, S. Gershman, D. O'Brien, S. Schieber, J. Waldo, D. Weinberger, and A. Wood, "Accountability of AI Under the Law: The Role of Explanation," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1711.01134>
- [45] D. Dua and E. Karra Taniskidou, "UCI Machine Learning Repository," 2017. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [46] L. Edwards and M. Veale, "Slave to the Algorithm? Why a Right to Explanation is Probably Not the Remedy You are Looking for," *Duke Law & Technology Review*, vol. 16, no. 1, pp. 18–84, 2017.
- [47] U. Ehsan, B. Harrison, L. Chan, and M. O. Riedl, "Rationalization: A Neural Machine Translation Approach to Generating Natural Language Explanations," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1702.07826>
- [48] E. R. Elenberg, A. G. Dimakis, M. Feldman, and A. Karbasi, "Streaming Weak Submodularity: Interpreting Neural Networks on the Fly," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1703.02647>
- [49] P. Flach, *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012.
- [50] R. Fong and A. Vedaldi, "Interpretable Explanations of Black Boxes by Meaningful Perturbation," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1704.03296>
- [51] A. A. Freitas, "Comprehensible Classification Models – A Position Paper," *ACM SIGKDD Explorations Newsletter*, vol. 15, no. 1, pp. 1–10, 2014.
- [52] B. Friedman and H. Nissenbaum, "Bias in Computer Systems," *ACM Transactions on Information Systems*, vol. 14, no. 3, pp. 330–347, 1996.

- [53] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.
- [54] J. H. Friedman and B. E. Popescu, "Predictive Learning via Rule Ensembles," *Annals of Applied Statistics*, vol. 2, no. 3, pp. 916–954, 2008.
- [55] M. Fumagalli, B. Bierman, B. Kiefer, J. Broekens, and M. Neerincx, "DR 5.1: PAL Technical Architecture and Software Architecture," EU H2020 PAL, Tech. Rep., 2016.
- [56] G. Fung, S. Sandilya, and R. B. Rao, "Rule Extraction from Linear Support Vector Machines," *11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'05), Proceedings*, pp. 32–40, 2005.
- [57] F. Gedikli, D. Jannach, and M. Ge, "How Should I Explain? a Comparison of Different Explanation Types for Recommender Systems," *International Journal of Human Computer Studies*, vol. 72, no. 4, pp. 367–382, 2014.
- [58] Gefen, Karahanna, and Straub, "Trust and TAM in Online Shopping: An Integrated Model," *MIS Quarterly*, vol. 27, no. 1, p. 51, 2003. [Online]. Available: <http://www.jstor.org/stable/10.2307/30036519>
- [59] A. Ghorbani, A. Abid, and J. Zou, "Interpretation of Neural Networks is Fragile," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1710.10547>
- [60] R. D. Gibbons, G. Hooker, M. D. Finkelman, D. J. Weiss, P. A. Pilkonis, E. Frank, T. Moore, and D. J. Kupfer, "The Computerized Adaptive Diagnostic Test for Major Depressive Disorder (CAD-MDD)," *The Journal of Clinical Psychiatry*, vol. 74, no. 07, pp. 669–674, jul 2013.
- [61] V. Gijsbers, "Reconciling Contrastive and Non-contrastive Explanation," *Erkenntnis*, pp. 1–15, 2017.
- [62] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin, "Peeking Inside the Black Box: Visualizing Statistical Learning With Plots of Individual Conditional Expectation," *Journal of Computational and Graphical Statistics*, vol. 24, no. 1, pp. 44–65, 2015.
- [63] B. Goodman and S. Flaxman, "European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation"," in *2016 ICML Workshop on Human Interpretability in Machine Learning (WHI)*, 2016, pp. 1–9.
- [64] R. Guidotti, A. Monreale, F. Turini, D. Pedreschi, and F. Giannotti, "A Survey Of Methods For Explaining Black Box Models," *arXiv preprint*, 2018. [Online]. Available: <http://arxiv.org/abs/1802.01933>
- [65] D. Gunning, "Explainable Artificial Intelligence (XAI)," 2016. [Online]. Available: <https://www.darpa.mil/program/explainable-artificial-intelligence>
- [66] Y. Guo and B. Selman, "ExOpaque: A Framework to Explain Opaque Machine Learning Models Using Inductive Logic Programming," in *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*. IEEE, oct 2007, pp. 226–22–.
- [67] J. Y. Halpern and J. Pearl, "Causes and Explanations: A Structural-Model Approach - Part I: Causes," in *Seventeenth Conference on Uncertainty in Artificial Intelligence, Proceedings*. San Francisco, CA: Morgan, 2001, pp. 194–202.
- [68] —, "Causes and Explanations: A Structural-Model Approach - Part II: Explanations," in *International Joint Conference on Artificial Intelligence (IJCAI), Proceedings*, 2001.
- [69] S. Hara and K. Hayashi, "Making Tree Ensembles Interpretable," *arXiv preprint*, 2016. [Online]. Available: <http://arxiv.org/abs/1606.05390>
- [70] D. Hein, S. Udluft, and T. A. Runkler, "Interpretable Policies for Reinforcement Learning by Genetic

- Programming,” *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1712.04170>
- [71] C. G. Hempel, *Aspects of Scientific Explanation and Other Essays in the Philosophy of Science*. New York, NY: The Free Press, 1965.
- [72] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell, “Generating Visual Explanations,” in *European Conference on Computer Vision (ECCV)*, 2016, pp. 3–19.
- [73] A. Henelius, K. Puolamäki, H. Boström, L. Asker, and P. Papapetrou, “A Peek into the Black Box: Exploring Classifiers by Randomization,” *Data Mining and Knowledge Discovery*, vol. 28, no. 5-6, pp. 1503–1529, 2014.
- [74] B. Herman, “The Promise and Peril of Human Evaluation for Model Interpretability,” in *Conference on Neural Information Processing Systems (NIPS)*, 2017.
- [75] M. Hildebrandt, “The New Imbroglia: Living with Machine Algorithms,” *The Art of Ethics in the Information Society. Mind you*, pp. 55–60, 2016.
- [76] G. E. Hinton and R. R. Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [77] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [78] G. Hooker, “Discovering Additive Structure in Black Box Functions,” in *10th ACM SIGKDD International Conference on Knowledge discovery in data mining (KDD’04), Proceedings*, 2004, pp. 575–580.
- [79] J. Huysmans, K. Dejaeger, C. Mues, J. Vanthienen, and B. Baesens, “An Empirical Evaluation of the Comprehensibility of Decision Table, Tree and Rule Based Predictive Models,” *Decision Support Systems*, vol. 51, no. 1, pp. 141–154, 2011.
- [80] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. New York, NY: Springer, 2014.
- [81] D. L. Jennings, T. M. Amabile, and L. Ross, “Informal Covariation Assessment: Data-based versus Theory Based Judgments,” in *Judgment Under Uncertainty: Heuristics and Biases*. New York, NY: Cambridge University Press, 1982, pp. 211–230.
- [82] U. Johansson and L. Niklasson, “Evolving Decision Trees Using Oracle Guides,” in *2009 IEEE Symposium on Computational Intelligence and Data Mining*. IEEE, 2009, pp. 238–244.
- [83] U. Johansson, R. König, and L. Niklasson, “The Truth is In There - Rule Extraction from Opaque Models Using Genetic Programming,” in *FLAIRS Conference*, 2004, pp. 658–663.
- [84] M. Kalisch and P. Bühlmann, “Causal Structure Learning and Inference: A Selective Review,” *Quality Technology & Quantitative Management*, vol. 11, no. 1, pp. 3–21, 2014.
- [85] A. Karpathy, J. Johnson, and L. Fei-Fei, “Visualizing and Understanding Recurrent Networks,” *arXiv preprint*, 2015. [Online]. Available: <http://arxiv.org/abs/1506.02078>
- [86] F. C. Keil, “Explaining and Understanding,” *Annual Review of Psychology*, vol. 57, pp. 227–254, 2006.
- [87] B. Kim and F. Doshi-Velez, “Interpretable Machine Learning: The Fuss, the Concrete and the Questions,” in *International Conference on Machine Learning (ICML)*, 2017.
- [88] B. Kim, C. Rudin, and J. Shah, “The Bayesian Case Model: A Generative Approach for Case-Based Reasoning and Prototype Classification,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015.
- [89] P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim, “The (Un)reliability of Saliency Methods,” *arXiv preprint*, 2017. [Online]. Available:

<http://arxiv.org/abs/1711.00867>

- [90] B. Kitchenham, "Procedures for Performing Systematic Reviews," Keele University, Tech. Rep. TR/SE-0401, 2004.
- [91] D. Knowles, *Explanation and its Limits*. Cambridge: Cambridge University Press, 1990.
- [92] D. J. Koehler, "Explanation, Imagination, and Confidence in Judgment," *Psychological Bulletin*, vol. 110, no. 3, pp. 499–519, 1991.
- [93] J. Koo, J. Kwac, W. Ju, M. Steinert, L. Leifer, and C. Nass, "Why did my car just do that? Explaining Semi-Autonomous Driving Actions to Improve Driver Understanding, Trust, and Performance," *International Journal on Interactive Design and Manufacturing*, vol. 9, no. 4, pp. 269–275, 2015.
- [94] J. Krause, A. Perer, and K. Ng, "Interacting with Predictions: Visual Inspection of Black-box Machine Learning Models," *ACM Conference on Human Factors in Computing Systems*, pp. 5686–5697, 2016.
- [95] J. Krause, A. Dasgupta, J. Swartz, Y. Aphinyanaphongs, and E. Bertini, "A Workflow for Visual Diagnostics of Binary Classifiers using Instance-Level Explanations," in *IEEE Conference on Visual Analytics Science and Technology (IEEE VAST 2017)*. IEEE, 2017.
- [96] R. Krishnan, G. Sivakumar, and P. Bhattacharya, "Extracting Decision Trees From Trained Neural Networks," *Pattern Recognition*, vol. 32, pp. 1999–2009, 1999.
- [97] S. Krishnan and E. Wu, "PALM: Machine Learning Explanations For Iterative Debugging," in *2nd Workshop on Human-In-the-Loop Data Analytics (HILDA'17), Proceedings*, 2017.
- [98] T. Kulesza, S. Stumpf, M. Burnett, S. Yang, I. Kwan, and W.-K. Wong, "Too Much, Too Little, or Just Right? Ways Explanations Impact End Users' Mental Models," in *IEEE Symposium on Visual Languages and Human-Centric Computing*. IEEE, 2013, pp. 3–10.
- [99] T. Kulesza, M. Burnett, W.-K. Wong, and S. Stumpf, "Principles of Explanatory Debugging to Personalize Interactive Machine Learning," in *20th International Conference on Intelligent User Interfaces (IUI'15), Proceedings*, 2015, pp. 126–137.
- [100] H. Lakkaraju, S. H. Bach, and L. Jure, "Interpretable Decision Sets: A Joint Framework for Description and Prediction," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16), Proceedings*, 2016, pp. 1675–1684.
- [101] T. Laugel, X. Renard, M.-J. Lesot, C. Marsala, and M. Detryniecki, "Defining Locality for Surrogates in Post-hoc Interpretability," in *2018 ICML Workshop on Human Interpretability in Machine Learning (WHI)*, 2018, pp. 47–53.
- [102] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [103] T. Lei, R. Barzilay, and T. Jaakkola, "Rationalizing Neural Predictions," *arXiv preprint*, 2016. [Online]. Available: <http://arxiv.org/abs/1606.04155>
- [104] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, "Interpretable Classifiers Using Rules and Bayesian Analysis: Building a Better Stroke Prediction Model," *Annals of Applied Statistics*, vol. 9, no. 3, pp. 1350–1371, sep 2015.
- [105] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and Understanding Neural Models in NLP," *arXiv preprint*, 2015. [Online]. Available: <http://arxiv.org/abs/1506.01066>
- [106] W. Li, J. Han, and J. Pei, "CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules," in *2001 IEEE International Conference on Data Mining, Proceedings*, 2001, pp. 369–376.
- [107] B. Y. Lim, A. K. Dey, and D. Avrahami, "Why and Why Not Explanations Improve the Intelligibility of Context-Aware Intelligent Systems," in *27th International Conference on Human Factors in Computing*

Systems (CHI'09), Proceedings, 2009, pp. 2119–2129.

- [108] P. Lipton, “Contrastive Explanation and Triangulation,” *Philosophy of Science*, vol. 58, no. 4, pp. 687–697, 1991.
- [109] —, *Inference to the Best Explanation*, 2nd ed. London: Routledge, 2004.
- [110] Z. C. Lipton, “The Mythos of Model Interpretability,” in *2016 ICML Workshop on Human Interpretability in Machine Learning (WHI)*, 2016.
- [111] S. Liu, X. Wang, M. Liu, and J. Zhu, “Towards Better Analysis of Machine Learning Models: A Visual Analytics Perspective,” *Visual Informatics*, vol. 1, no. 1, pp. 48–56, 2017.
- [112] T. Lombrozo, “The Structure and Function of Explanations,” *Trends in Cognitive Sciences*, vol. 10, no. 10, pp. 464–470, 2006.
- [113] Y. Lou, R. Caruana, J. Gehrke, and G. Hooker, “Accurate Intelligible Models with Pairwise Interactions,” in *19th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'13), Proceedings*, 2013, pp. 623–631.
- [114] S. Lundberg and S.-I. Lee, “An Unexpected Unity Among Methods for Interpreting Model Predictions,” in *29th Conference on Neural Information Processing Systems (NIPS 2016)*, 2016.
- [115] A. Mahendran and A. Vedaldi, “Understanding Deep Image Representations by Inverting Them,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Proceedings*. IEEE, nov 2015, pp. 5188–5196.
- [116] D. M. Malioutov, K. R. Varshney, A. Emad, and S. Dash, “Learning Interpretable Classification Rules with Boolean Compressed Sensing,” *Transparent Data Mining for Big and Small Data. Studies in Big Data*, vol. 32, 2017.
- [117] D. R. Mandel, “Counterfactual and Causal Explanation: From Early Theoretical Views to New Frontiers,” in *The Psychology of Counterfactual Thinking*, D. R. Mandel, D. J. Hilton, and P. Catellani, Eds. London: Routledge, 2005, pp. 11–27.
- [118] C. D. Manning, P. Ragahvan, and H. Schutze, *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press, 2009.
- [119] D. Martens and F. Provost, “Explaining Data-Driven Document Classifications,” *MIS Quarterly*, vol. 38, no. 1, pp. 73–99, 2014.
- [120] D. Martens, J. Vanthienen, W. Verbeke, and B. Baesens, “Performance of Classification Models from a User Perspective,” *Decision Support Systems*, vol. 51, no. 4, pp. 782–793, 2011.
- [121] A. L. McGill and J. G. Klein, “Contrastive and Counterfactual Reasoning in Causal Judgment,” *Journal of Personality and Social Psychology*, vol. 64, no. 6, pp. 897–905, 1993.
- [122] G. A. Miller, “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information,” *Psychological Review*, vol. 63, no. 2, pp. 81–97, 1956.
- [123] T. Miller, “Explanation in Artificial Intelligence: Insights from the Social Sciences,” *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1706.07269>
- [124] T. Miller, P. Howe, and L. Sonenberg, “Explainable AI: Beware of Inmates Running the Asylum,” in *International Joint Conference on Artificial Intelligence (IJCAI), Proceedings*, 2017, pp. 36–41.
- [125] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with Deep Reinforcement Learning,” in *NIPS Deep Learning Workshop*, 2013.
- [126] D. Moher, A. Liberati, J. Tetzlaff, D. G. Altman, D. Altman, G. Antes, D. Atkins, V. Barbour, N. Barrowman, J. A. Berlin, J. Clark, M. Clarke, D. Cook, R. D’Amico, J. J. Deeks, P. J. Devereaux, K. Dickersin,

- M. Egger, E. Ernst, P. C. Gøtzsche, J. Grimshaw, G. Guyatt, J. Higgins, J. P. Ioannidis, J. Kleijnen, T. Lang, N. Magrini, D. McNamee, L. Moja, C. Mulrow, M. Napoli, A. Oxman, B. Pham, D. Rennie, M. Sampson, K. F. Schulz, P. G. Shekelle, D. Tovey, and P. Tugwell, "Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement," *PLoS Medicine*, vol. 6, no. 7, 2009.
- [127] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. MIT Press, 2012.
- [128] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K. R. Müller, "Explaining Nonlinear Classification Decisions with Deep Taylor Decomposition," *Pattern Recognition*, vol. 65, no. C, pp. 211–222, 2017.
- [129] G. Montavon, W. Samek, and K.-R. Müller, "Methods for Interpreting and Understanding Deep Neural Networks," *Digital Signal Processing: A Review Journal*, vol. 73, pp. 1–15, jun 2017.
- [130] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Proceedings*. IEEE, 2015, pp. 2574–2582.
- [131] M. A. Neerincx, F. Kaptein, M. A. van Bekkum, H.-U. Krieger, B. Kiefer, R. Peters, J. Broekens, Y. Demiris, and M. Sapelli, "Ontologies for Social, Cognitive and Affective Agent-Based Support of Child's Diabetes Self-Management," in *Workshop on Artificial Intelligence for Diabetes. Artificial Intelligence for Diabetes (AID), located at ECAI 2016*, 2016, pp. 35–38.
- [132] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the Preferred Inputs for Neurons in Neural Networks via Deep Generator Networks," *Advances in Neural Information Processing Systems (NIPS)*, vol. 29, 2016.
- [133] H. Núñez, C. Angulo, and A. Català, "Rule Extraction from Support Vector Machines," in *European Symposium on Artificial Neural Networks, Proceedings*, 2002, pp. 107–112.
- [134] K. Odajima, Y. Hayashi, G. Tianxia, and R. Setiono, "Greedy Rule Generation from Discrete Data and Its Use in Neural Network Rule Extraction," *Neural Networks*, vol. 21, no. 7, pp. 1020–1028, 2008.
- [135] J. D. Olden and D. A. Jackson, "Illuminating the "Black Box": a Randomization Approach for Understanding Variable Contributions in Artificial Neural Networks," *Ecological Modelling*, vol. 154, no. 1-2, pp. 135–150, 2002.
- [136] M. Pacer and T. Lombrozo, "Ockham's Razor Cuts to the Root: Simplicity in Causal Explanation," *Journal of Experimental Psychology: General*, vol. 146, no. 12, pp. 1761–1780, 2017.
- [137] Parliament and Council of the European Union, "General Data Protection Regulation," 2016.
- [138] K. Patel, N. Bancroft, S. M. Drucker, J. Fogarty, A. J. Ko, and J. Landay, "Gestalt: Integrated Support for Implementation and Analysis in Machine Learning," in *23rd annual ACM symposium on User Interface Software and Technology (UIST '10), Proceedings*, 2010, pp. 37–46.
- [139] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [140] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007.
- [141] D. L. Poole and A. K. Mackworth, *Artificial Intelligence: Foundations of Computational Agents*. Cambridge: Cambridge University Press, 2010.
- [142] P. Pu and L. Chen, "Trust-Inspiring Explanation Interfaces for Recommender Systems," *Knowledge-Based Systems*, vol. 20, no. 6, pp. 542–556, 2007.

- [143] P. Pu, L. Chen, and R. Hu, "A User-Centric Evaluation Framework for Recommender Systems," *5th ACM Conference on Recommender systems (RecSys '11), Proceedings*, pp. 157–164, 2011.
- [144] N. Puri, P. Gupta, P. Agarwal, S. Verma, and B. Krishnamurthy, "MAGIX: Model Agnostic Globally Interpretable Explanations," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1706.07160>
- [145] A. Radford, R. Jozefowicz, and I. Sutskever, "Learning to Generate Reviews and Discovering Sentiment," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1704.01444>
- [146] P. Ravikumar, J. Lafferty, H. Liu, and L. Wasserman, "Sparse Additive Models," *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, vol. 71, no. 5, pp. 1009–1030, 2009.
- [147] D. G. Ray, J. Neugebauer, K. Sassenberg, and F. W. Hesse, "Motivated Shortcomings in Explanation: The Role of Comparative Self-Evaluation and Awareness of Explanation Recipient's Knowledge," *Journal of Experimental Psychology: General*, vol. 142, no. 2, pp. 445–457, 2013.
- [148] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," in *22nd ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'16), Proceedings*, 2016, pp. 1135–1144.
- [149] —, "Anchors: High-Precision Model-Agnostic Explanations," in *AAAI Conference on Artificial Intelligence*, 2018.
- [150] L. Rieber, S. Tzeng, and K. Tribble, "Discovery Learning, Representation, and Explanation within a Computer-Based Simulation: Finding the Right Mix," *Learning and Instruction*, vol. 14, no. 3, pp. 307–323, 2004.
- [151] R. L. Rivest, "Learning Decision Lists," *Machine Learning*, vol. 2, no. 3, pp. 229–246, 1987.
- [152] M. Robnik-Šikonja and I. Kononenko, "Explaining Classifications for Individual Instances," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, pp. 589–600, 2008.
- [153] N. J. Roese, "Counterfactual Thinking," *Psychological Bulletin*, vol. 121, no. 1, pp. 133–148, 1997.
- [154] N. J. Roese and J. M. Olson, "Counterfactual Thinking: A Critical Overview," in *What might have been: The social psychology of counterfactual thinking*. Hillsdale, NJ: Lawrence Erlbaum Associates, 1995, pp. 1–55.
- [155] A. S. Ross, M. C. Hughes, and F. Doshi-Velez, "Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations," in *International Joint Conference on Artificial Intelligence (IJCAI), Proceedings*, 2017, pp. 2662–2670.
- [156] D. H. Ruben, *Explaining Explanation*. London: Routledge, 2004.
- [157] S. Ruping, "Learning Interpretable Models," Ph.D. dissertation, TU Dortmund University, 2006.
- [158] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2014.
- [159] L. Saitta and F. Neri, "Learning in the 'Real World'," *Machine Learning*, vol. 30, no. 2-3, pp. 133–163, 1998.
- [160] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1708.08296>
- [161] I. Sánchez, T. Rocktäschel, S. Riedel, and S. Singh, "Towards Extracting Faithful and Descriptive Representations of Latent Variable Models," in *AAAI Spring Symposium on Knowledge Representation and Reasoning*, vol. 3, 2015, pp. 35–38.
- [162] V. Schetinin, J. E. Fieldsend, D. Partridge, T. J. Coats, W. J. Krzanowski, R. M. Everson, T. C. Bailey, and A. Hernandez, "Confident Interpretation of Bayesian Decision Tree Ensembles for Clinical

- Applications," *IEEE Transactions on Information Technology in Biomedicine*, vol. 11, no. 3, pp. 312–319, 2007.
- [163] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization," in *NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems*, 2016.
- [164] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning Important Features Through Propagating Activation Differences," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1704.02685>
- [165] P. K. Shukla and S. P. Tripathi, "A Review on the Interpretability-Accuracy Trade-Off in Evolutionary Multi-Objective Fuzzy Systems (EMOFS)," *Information (Switzerland)*, vol. 3, no. 3, pp. 256–277, 2012.
- [166] R. Shwartz-Ziv and N. Tishby, "Opening the Black Box of Deep Neural Networks via Information," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1703.00810>
- [167] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
- [168] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," *arXiv preprint*, 2013. [Online]. Available: <http://arxiv.org/abs/1312.6034>
- [169] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, 2nd ed. London: MIT Press, 2000.
- [170] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for Simplicity: The All Convolutional Net," in *International Conference on Learning Representations (ICLR), Workshop*, 2015.
- [171] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush, "LSTMVis: A Tool for Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 667–676, 2018.
- [172] C. Strobl, A. L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis, "Conditional Variable Importance for Random Forests," *BMC Bioinformatics*, vol. 9, pp. 1–11, 2008.
- [173] E. Strumbelj and I. Kononenko, "An Efficient Explanation of Individual Classifications using Game Theory," *Journal of Machine Learning Research*, vol. 11, pp. 1–18, 2010.
- [174] G. Su, D. Wei, K. R. Varshney, and D. M. Malioutov, "Learning Sparse Two-Level Boolean Rules," in *IEEE International Workshop on Machine Learning for Signal Processing, MLSP*, vol. 2016-Novem. IEEE, sep 2016, pp. 1–6.
- [175] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks," in *34th International Conference on Machine Learning (ICML), Proceedings*, 2017.
- [176] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, 1998.
- [177] P. Tamagnini, J. Krause, A. Dasgupta, and E. Bertini, "Interpreting Black-Box Classifiers Using Instance-Level Visual Explanations," in *2nd Workshop on Human-In-the-Loop Data Analytics (HILDA'17), Proceedings*, 2017.
- [178] T. Tamayo, J. Rosenbauer, S. H. Wild, A. M. Spijkerman, C. Baan, N. G. Forouhi, C. Herder, and W. Rathmann, "Diabetes in Europe: An Update," *Diabetes Research and Clinical Practice*, vol. 103, no. 2, pp. 206–217, 2014.

- [179] H. F. Tan, G. Hooker, and M. T. Wells, "Tree Space Prototypes: Another Look at Making Tree Ensembles Interpretable," in *NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems*, 2016.
- [180] K. H. Teigen, A. B. Kanten, and J. A. Terum, "Going to the Other Extreme: Counterfactual Thinking Leads to Polarised Judgments," *Thinking and Reasoning*, vol. 17, no. 1, pp. 1–29, 2011.
- [181] J. J. Thiagarajan, B. Kailkhura, P. Sattigeri, and K. N. Ramamurthy, "TreeView: Peeking into Deep Neural Networks Via Feature-Space Partitioning," in *NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems*, 2016.
- [182] J. J. Thomas and K. A. Cook, *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center, 2005.
- [183] R. Tibshirani, "Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- [184] N. Tintarev and J. Masthoff, "Designing and Evaluating Explanations for Recommender Systems," in *Recommender Systems Handbook*. Boston, MA: Springer US, 2011, pp. 479–510.
- [185] G. Tolomei, F. Silvestri, A. Haines, and M. Lalmas, "Interpretable Predictions of Tree-based Ensembles via Actionable Feature Tweaking," in *23rd ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'17), Proceedings*, 2017, pp. 465–474.
- [186] G. G. Towell and J. W. Shavlik, "Extracting Refined Rules from Knowledge-Based Neural Networks," *Machine Learning*, vol. 13, no. 1, pp. 71–101, 1993.
- [187] R. Turner, "A Model Explanation System," *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016.
- [188] F. Y. Tzeng and K. L. Ma, "Opening the Black Box-Data Driven Visualization of Neural Networks," in *IEEE Visualization*, 2005, pp. 383–390.
- [189] B. Ustun and C. Rudin, "Methods and Models for Interpretable Linear Classification," *arXiv preprint*, 2014. [Online]. Available: <http://arxiv.org/abs/1405.4047>
- [190] —, "Supersparse Linear Integer Models for Optimized Medical Scoring Systems," *Machine Learning*, vol. 102, no. 3, pp. 349–391, 2016.
- [191] B. C. Van Fraassen, *The Scientific Image*. Oxford University Press, 1980.
- [192] G. Vandewiele, O. Janssens, F. Ongenae, F. De Turck, and S. Van Hoecke, "GENESIM: Genetic Extraction of a Single, Interpretable Model," in *NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems*, 2016.
- [193] A. Vedaldi, "Understanding Models via Visualizations, Attribution, and Semantic Identification," in *Tutorial on Interpretable Machine Learning for Computer Vision (CVPR 2018)*, 2018.
- [194] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR," *arXiv preprint*, 2017. [Online]. Available: <http://arxiv.org/abs/1711.00399>
- [195] F. Wang and C. Rudin, "Falling Rule Lists," in *Artificial Intelligence and Statistics (AISTATS)*, vol. 38, 2015.
- [196] J. Wang, R. Fujimaki, and Y. Motohashi, "Trading Interpretability for Accuracy," in *21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15), Proceedings International Conference on Knowledge Discovery in Data Mining*. New York, New York, USA: ACM Press, 2015, pp. 1245–1254.
- [197] T. Wang, C. Rudin, F. Velez-Doshi, Y. Liu, E. Klampfl, and P. Macneille, "Bayesian Rule Sets for Interpretable Classification," in *IEEE International Conference on Data Mining (ICDM), Proceedings*. IEEE,

2017, pp. 1269–1274.

- [198] R. J. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
- [199] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.
- [200] World Health Organization, *Global Report on Diabetes*, 2016, vol. 978.
- [201] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention,” *arXiv preprint*, 2015. [Online]. Available: <http://arxiv.org/abs/1502.03044>
- [202] S. C.-H. Yang and P. Shafto, “Explainable Artificial Intelligence via Bayesian Teaching,” in *Conference on Neural Information Processing Systems*, 2017.
- [203] X. Yin and J. Han, “CPAR: Classification based on Predictive Association Rules,” in *2003 SIAM International Conference on Data Mining, Proceedings*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2003, pp. 331–335.
- [204] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, “Understanding Neural Networks Through Deep Visualization,” *arXiv preprint*, 2015. [Online]. Available: <http://arxiv.org/abs/1506.06579>
- [205] T. Zahavy, N. B. Zrihem, and S. Mannor, “Graying the Black Box: Understanding DQNs,” in *33rd International Conference on Machine Learning (ICML), Proceedings*, 2016, pp. 1–32.
- [206] M. D. Zeiler and R. Fergus, “Visualizing and Understanding Convolutional Networks,” *Computer Vision (ECCV 2014)*, vol. 8689, pp. 818–833, 2014.
- [207] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, “Top-Down Neural Attention by Excitation Backprop,” *International Journal of Computer Vision*, pp. 1–19, 2017.
- [208] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning Deep Features for Discriminative Localization,” in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 2015, pp. 2921–2929.
- [209] Y. Zhou and G. Hooker, “Interpreting Models via Single Tree Approximation,” *arXiv preprint*, 2016. [Online]. Available: <http://arxiv.org/abs/1610.09036>
- [210] Z.-H. Zhou, Y. Jiang, and S.-F. Chen, “Extracting Symbolic Rules from Trained Neural Network Ensembles,” *AI Communications*, vol. 16, no. 1, pp. 3–15, 2003.

APPENDIX

A Machine Learning Model Evaluation

To select the model with the best performance on a task, machine learning (ML) algorithms are *evaluated* on their performance. In this section, we introduce ML evaluation methods for supervised and reinforcement learning.

Classification

Classification performance assessment is best illustrated using a *confusion matrix* (contingency table), that counts the equal instances of the predicted label $\hat{y}_i = \hat{f}(x_i)$ and actual label $y_i = f(x_i)$ for each instance i in the test set T . The *classifier* is a mapping $\hat{c} : \mathcal{X} \rightarrow \mathcal{C}$, where $\mathcal{C} = \{c_1, c_2, \dots, c_k\}$ is a finite set of k classes. To indicate the equality of labels, we use indicator function $I[\cdot]$ that evaluates to 1 if the enclosed logical statement evaluates to true (e.g., we predict it to be class one and the actual label is also class one), and 0 otherwise [49].

For a given class $c \in \mathcal{C}$ we use a short-hand notation where (i) TP_c (*true positives*) are the number of correctly classified instances, (ii) FP_c (*false positives*) are the number of instances wrongly predicted as c , (iii) FN_c (*false negatives*) are the instances the classifier missed out on—i.e., did not predict as c while they were c —, and (iv) TN_c (*true negatives*) are the instances not of class c that are correctly identified as not being c .¹ They are defined as

$$TP_c = \sum_{i=1}^n I[\hat{c}(x_i) = y_i = c] \quad (\text{A.1})$$

$$FP_c = \sum_{i=1}^n I[\hat{c}(x_i) = c \wedge y_i \neq c] \quad (\text{A.2})$$

$$FN_c = \sum_{i=1}^n I[\hat{c}(x_i) \neq c \wedge y_i = c] \quad (\text{A.3})$$

$$TN_c = \sum_{i=1}^n I[\hat{c}(x_i) \neq c \wedge f(x_i) \neq c] \quad (\text{A.4})$$

Table A.1 shows a confusion matrix for an arbitrary multi-class classifier that distinguishes between animals, vehicles and objects. Each row shows the actual label $f(\cdot)$ as recorded in the test set, and each column shows the accompanying prediction $\hat{c}(\cdot)$ of the classifier. For instance, the first row shows that of the instances that are actually animals 50 were classified correctly as animal, and another 50 were classified incorrectly: 30 as vehicle and 20 as object. The confusion

¹ False negatives (FN) are also referred to as Type-I errors, while false positives (FP) are also denoted as Type-II errors.

matrix shows the results of the classifier on a test set with $n = 240$ instances. We denote the classes $\mathcal{C} = \{animal, vehicle, object\}$, where the number of classes $|\mathcal{C}| = 3$.

Table A.1 Example confusion matrix for multi-class classification on a test set with $n = 240$ instances

		Predicted		
		Animal	Vehicle	Object
Actual	Animal	50	30	20
	Vehicle	10	90	0
	Object	5	25	10

Accuracy describes the proportion of correctly classified instances. *Precision* (positive predicted value; PPV) indicates the fraction of relevant instances among the identified instances. *Recall* (sensitivity) measures what part of the class instances the classifier can distinguish. *Specificity* (true negative rate; TNR) indicates the proportion of instances not of a class that are correctly classified as such. They are calculated using the marginals (row-wise and column-wise sums) of each class, and the mean precision, recall and specificity can be determined by summing the individual scores per class and dividing them by the number of classes, e.g. for precision $P = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} P_c$. In binary classification, the precision, recall and specificity are directly calculated as their respective scores class of interest—typically the majority class (class with most instances).

Oftentimes, a trade-off between precision and recall is required to take into account the false positives as well as the false negatives. As a measure for this trade-off, the F_1 score can be used: the harmonic mean of precision and recall [49, 158]. Note that each measure takes on a value in the range $[0, 1]$, where zero indicates low performance and one indicates a perfect score. Table A.2 shows the aforementioned performance measures and how they are calculated. In addition, for the example in Table A.1 the table includes a class-specific example and the average score for the example.

Table A.2 Overview of classification performance measures, with class-specific examples and average values for Table A.1

Measure		Class-specific example	Average
<i>Accuracy</i>	$A = \frac{1}{ N } \sum_{c \in \mathcal{C}} TP_c$	-	$A = \frac{50+90+10}{240} = 0.625$
<i>Precision</i>	$P_c = TP_c / (TP_c + FP_c)$	$P_{animal} = \frac{50}{50+10+5} \approx 0.77$	$P \approx \frac{0.77+0.62+0.33}{3} \approx 0.57$
<i>Recall</i>	$R_c = TP_c / (TP_c + FN_c)$	$R_{vehicle} = \frac{90}{10+90+0} = 0.9$	$R = \frac{0.5+0.9+0.25}{3} = 0.55$
<i>Specificity</i>	$S_c = TN_c / (TN_c + FP_c)$	$S_{object} = \frac{50+30+10+90}{50+30+10+90+20+0} = 0.9$	$S \approx \frac{0.89+0.61+0.9}{3} = 0.8$
<i>F₁ score</i>	$F_1 = 2 \cdot P \cdot R / (P + R)$	-	$F_1 \approx \frac{2 \cdot 0.55 \cdot 0.57}{0.55+0.57} \approx 0.56$

The preceding measures are typically sensitive to class distribution (e.g., unbalanced classes) and the number of instances used for testing [120]. Instead, the *receiver operating characteristic* (ROC) curve is commonly used when comparing across data sets and class distributions [49, 127]. The ROC curve plots the ranked relationship between *recall* R on the y-axis and the *false positive rate* $1 - S$ on the x-axis [127]. As a comparative measure, the *area under the ROC curve* (AUC) is a performance indicator in the range $[0, 1]$, where 0.5 indicates similar performance to random

classification and 1 is a classifier that perfectly predicts all instances.

Regression analysis

The performance of regression analysis, where $h : \mathcal{X} \rightarrow \mathbb{R}$, is evaluated by comparing the value of the predicted outcome $\hat{y}_i = \hat{f}(x_i)$ to the actual outcome y_i for a data point x_i . Evaluation is done using a loss function that is based around the *residual* $e_i = y_i - \hat{y}_i$. A popular measure of fit based on Euclidean distance is the *mean squared error* (MSE) or *root mean squared error* (RMSE), given by

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (\text{A.5})$$

$$\text{RMSE} = \sqrt{\text{MSE}} \quad (\text{A.6})$$

where n are the number of instances. A lower MSE/RMSE indicates a better performance. The main downside of MSE and RMSE is that they are in scale of measurement of the predictor variable y , making them incompatible when comparing across data sets. Instead, the R^2 *statistic* measures the goodness of fit independent of the measurement scale of y . A higher R^2 indicates a better performance. Important to note is that the value of R^2 is still affected by the number of features considered in the regression. To compare models with different numbers of variables, a popular approach is using the *adjusted* R^2 statistic that penalizes adding non-descriptive variables to the model.² They are given by

$$R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (\text{A.7})$$

$$R_{adj}^2 = 1 - \left(\frac{(1 - R^2)(n - 1)}{n - k - 1} \right) \quad (\text{A.8})$$

where $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ denotes the average value of the observed data and k is the number of independent variables used in the model [80].

Cross-validation

During training, the principle of performance measurement can also be applied iteratively to assess performance of the current version of the model, a process referred to as *cross-validation* (CV). In this resampling method, a small part of the training data is held out to assess the model's performance on. After assessing performance, a different part of the data is held-out and the previously held-out data is used for training. By immediately checking the model performance, we can determine during training whether the model generalizes well on unseen data. If the data is split up into k approximately equal randomly sampled parts (where one fold is used for testing and $k - 1$ folds are used for training) it is referred to as *k-fold CV*. In *stratified k-fold CV* the data per fold is not sampled randomly, but the data is rearranged to ensure that each fold is a

² Other popular measures include *Akaike's information criterion* (AIC) and *Bayesian information criterion* (BIC). We refer the reader to James et al. [80, ch. 6] for a more detailed overview.

good representation of the whole training set. If the held-out part for CV is a single instance it is referred to as *leave-one-out* CV (LOOCV). Figure A.1 illustrates k-fold CV on a training set split into ten folds.

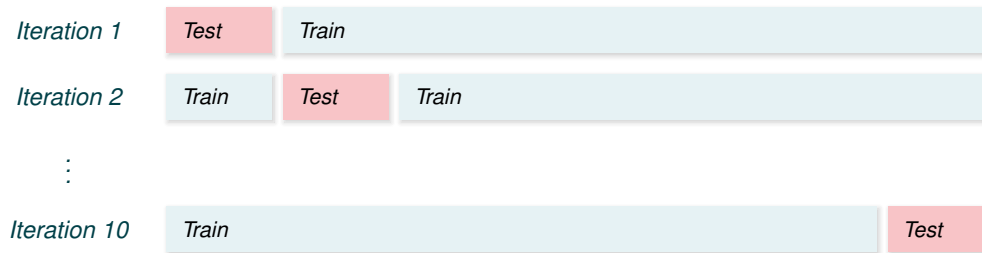


Figure A.1 k-Fold cross-validation with $k = 10$ folds

Reinforcement learning performance

In Section 2.1 we mentioned that in reinforcement learning (RL) the goal is to maximize the cumulative reward the agent receives in the long run. The total reward R_t from time step t can be defined as the *sum of rewards*

$$R_t = r_t + r_{t+1} + r_{t+2} + \dots + r_n = \sum_{i=0}^n r_{t+i} \quad (\text{A.9})$$

where n is the final time step.

However, usually the behavior the environment is *stochastic*—there may be some randomness involved in selecting the next state. This uncertainty increases the more actions the agent performs, possibly resulting in a very different actual sum of rewards compared to the expected one when performing the same policy. In addition, it cannot always be assumed that the task will terminate, i.e. the environment can have an infinite horizon $n = \infty$. We denote tasks with a final terminal state *episodic*, while ones with an infinite horizon are called *continuous*.

To counter these issues, RL techniques typically use a *discount factor* $\gamma \in [0, 1]$ describing the preference of an agent for immediate rewards over future ones. The *discounted future reward* takes into account this discount factor:

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{n-t} r_n = \sum_{i=0}^n \gamma^i r_{t+i} \quad (\text{A.10})$$

If we only want to rely on immediate rewards we can set discount factor $\gamma = 0$. Agents only concerned with direct rewards are referred to as *myopic*. If instead we want to balance the two, we select γ anywhere inbetween zero and one. As γ approaches one the agent takes future rewards more into account for their objective. Observe that for $\gamma = 1$ the discounted future reward (Eq. A.10) equals the sum of rewards (Eq. A.9) [176].

B Machine Learning Techniques

We overview popular ML techniques used for learning models that form the predominant foundation for interpretable ML (iML).³ In particular, we detail:

- ▶ Regression analysis techniques (linear regression and generalized additive models);
- ▶ Probabilistic classification techniques (logistic regression and naive Bayes);
- ▶ Decision boundary classification techniques (support vector machines, decision trees and rule-based methods);
- ▶ Neural networks;
- ▶ Ensemble methods (bagging and boosting), and;
- ▶ Reinforcement learning techniques.

Regression analysis

Regression analysis techniques estimate the quantitative response variable (dependent variable) y by fitting a predictor function that estimates the relationship between y and predictors (independent variables) a_1, a_2, \dots, a_k . We discuss linear regression, a popular, relatively comprehensible method that provides the baseline for regression models. Furthermore, we briefly discuss a less restrictive approach to regression: generalized additive models.

Linear regression. *Linear regression* models predict the quantitative response y by fitting a linear function on the predictors. We multiply each independent variable a_k with a weight β_k , that indicates the increase in y a one unit increase in a_k results in. In addition, the intercept term β_0 that gives the expected value of y when $a_1 = a_2 = \dots = a_k = 0$. In general, a linear regression function is defined as

$$y = \beta_0 + \beta_1 a_1 + \beta_2 a_2 + \dots + \beta_k a_k + \epsilon \quad (\text{A.11})$$

where ϵ is an error term that cannot be predicted by the independent variables.

If only terms β_0 and $\beta_1 a_1$ are considered, we refer to the problem as *simple linear regression*, while if it includes more terms it is called *multiple linear regression*.

The goal of regression is to estimate coefficients $\beta_0, \beta_1, \dots, \beta_k$. This involves finding estimators $\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_k$ that minimize the error on the data, to obtain an estimator function $\hat{y} = \hat{f}(x)$. The estimator function is given by

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 a_1 + \hat{\beta}_2 a_2 + \dots + \hat{\beta}_k a_k \quad (\text{A.12})$$

³ A large body of work considers ML techniques in more detail. We refer the interested reader to works such as Russell and Norvig [158], Manning et al. [118] and James et al. [80] for a more elaborate discussion of ML techniques.

Finding the estimators is done through *maximum likelihood* estimation. The error is measured using the *residual sum of squares* (RSS)—that also is the basis of the previously introduced MSE—that estimates the error for all n data points in the training set:

$$\text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (\text{A.13})$$

Due to their simple relationship between input features and the output, linear regression models are generally perceived as a good first approach to predict an outcome variable [80]. In addition, *variable selection* methods (e.g., forward selection or backward selection) can reduce the number of predictors to only include the ones contributing most to the prediction. *Lasso regression* [183] incorporates feature selection and regularization to combine high accuracy with good generalizability and interpretability through the production of a *sparse* linear model (i.e., one using fewer variables in its prediction). For its regularization, Lasso uses ℓ_1 *regularization* that penalizes a regressor based on the absolute values of the β -coefficients. *Ridge regression* penalizes the coefficients using ℓ_2 *regularization*—based on the squared values of the coefficients. Ridge regression performs best when the actual relationship is close to linear—trading off a little increase in bias for a major decrease in variance [80].

Generalized Additive Models (GAM). Even though linear regression models have advantages in terms of interpretability, the restrictive nature of the linearity assumption for all predictors can impose limitations on the predictive power of the model. Whereas linear regression models restrict each term to only include a numeric coefficient β , *Generalized Additive Models* (GAMs) allow for a more flexible representation by also allowing non-linear relationships between the independent and dependent variables. Each term a_i is represented by a function $f_i(a_i)$ such that

$$y = \beta_0 + f_1(a_1) + f_2(a_2) + \dots + f_k(a_k) + \epsilon \quad (\text{A.14})$$

where like linear regression β_0 is intercept and ϵ the error term [80]. For example, function $f(\cdot)$ can be a *polynomial* $f_i(a_i) = \beta_{i1}a_i + \beta_{i2}a_i^2 + \beta_{i3}a_i^3$. The restriction that each term a_i is additive has the advantage that individual effects of each term can still be examined. However, note that fitting more complex functions (i) is susceptible to *overfitting* to the training data (following the relationship too well and thereby not generalizing well on unseen data), and (ii) is computationally more expensive than simple linear coefficients.

Probabilistic classification

The first discussed approach for classification is *probabilistic classification*, where a classifier $\hat{c}(\cdot)$ outputs the probability $P(c | x_i)$ that an instance falls in a particular class c . This type of classifier is called a *Bayes classifier* [80]. Given a probability for each class $c \in \mathcal{C}$, the predicted class can then be obtained by selecting the class with the highest probability using

$$\hat{y}_i = \arg \max_{c \in \mathcal{C}} P(c | x_i) \quad (\text{A.15})$$

where $x_i = \langle a_{i1}, a_{i2}, \dots, a_{ik} \rangle$ is the instance to predict and \hat{y}_i is the corresponding class prediction.

We discuss two methods for probabilistic classification: *logistic regression* and the *naive Bayes* classifier.

Logistic regression. Regression techniques can be extended to *classification* problems by transforming output variable y to a probability $P(x)$ that given the dependent variables $x = \langle a_1, a_2, \dots, a_k \rangle$ the outcome belongs to a particular class. This requires that the outcome is *binary*, where if the probability of one increases the probability of the other class decreases equivalently. Because the probability lies in range $[0, 1]$, we also impose this restriction on $y = P(x)$ by using a logistic function expanding on multiple linear regression of Eq. A.12, such that

$$\ln \left(\frac{P(x)}{1 - P(x)} \right) = \beta_0 + \beta_1 a_1 + \beta_2 a_2 + \dots + \beta_k a_k \quad (\text{A.16})$$

where we refer to the left-hand side of Eq. A.16 as the *log-odds* [80]. To acquire $P(x)$ directly, Eq. A.16 can be rewritten as

$$P(x) = \frac{\exp(\beta_0 + \beta_1 a_1 + \beta_2 a_2 + \dots + \beta_k a_k)}{1 + \exp(\beta_0 + \beta_1 a_1 + \beta_2 a_2 + \dots + \beta_k a_k)} \quad (\text{A.17})$$

Naive Bayes classifier. A Bayesian classifier uses the observed probabilities in the independent variables to determine the probability that a new data point falls within a particular class. In its simplest form, the *naive Bayes* classifier assumes that all observed independent variables are independent of each other in predicting the dependent variable. In other words, it disregards all correlations between the features. It learns from the distributions of $P(c)$ and $P(x_i | c)$ for the input features in the data, and learns the probability for each class given by

$$P(c | x) = P(c) \prod_{a \in x} P(a | c) \quad (\text{A.18})$$

where x are all observed feature values in the data set.

Support vector machine (SVM)

A support vector machine (SVM) is a classification algorithm that aims to find a decision hyperplane with maximum distance from any point in the training data [118]. To construct the separator, the SVM uses support vectors formed by a small number of data points in the input data, and it aims to maximize the margin between the *maximal margin separator* and these support vectors [118, 158]. Even though data may not be linearly separable in the input space, the SVM can use a *kernel trick* to embed the data in a higher dimensional space where the data can be separated linearly by a decision hyperplane [158]. For instance, a data set with two features a_1 and a_2 may not be linearly separable in the input space, but be linearly separable in feature space with features a_1^2 and $\sqrt{a_2 - a_1}$.

If in case of noisy data we do not want or it is difficult to linearly separate the data, SVM can use

a *soft margin classifier* instead of a strict decision boundary [158]. A soft margin classifier allows for some observations to fall on the wrong side of the decision boundary, and instead penalizes them based on their distance from the boundary [158].

While in its base form an SVM is used for binary classification, SVMs can also be used for multi-class classification (*Multi-Class SVM*) or for regression (*support vector regression; SVR*) [127].

Decision trees and rule-based methods

Decision trees (DT) and *decision rules* (DR) reach an output decision by performing a sequence of tests on the input values of an instance. They are popular approaches for supervised learning and anomaly detection—most commonly used for classification but also applicable to regression tasks. The popularity of decision trees and rule-based approaches originates from that they are natural representations for humans [80, 158].

DTs and DRs are constructed around the evaluation of *Boolean expressions*, that group the input space into areas (*decision regions*) by evaluating to either *true* or *false*. If the logical statement evaluates to true for a given instance, that instance falls within the decision region for instances where the expression holds. Else, it falls outside the decision region. We continue this process recursively until a decision is reached regarding the instance. We refer to the hyperplane that separates the decision region from the remainder of the feature space as the *decision boundary*. The most simple form of these logical expressions, *literals*, are equalities of the form *feature = value*, or inequalities of a form such as *feature < value* or *feature ≠ value* [49]. Note that inequalities (expect not being equal to a value) require that the feature under consideration has an ordering—i.e., is ordinal or continuous. More complex Boolean expressions can be formed (i) using logical connectives (e.g., *conjunction* (AND; \wedge), *disjunction* (OR; \vee) and *negation* (NOT; \neg)), or (ii) by allowing non-axis aligned *oblique rules*: linear combinations of variables such as $a_1 \cdot a_2 < \text{value}$ [49, 79].

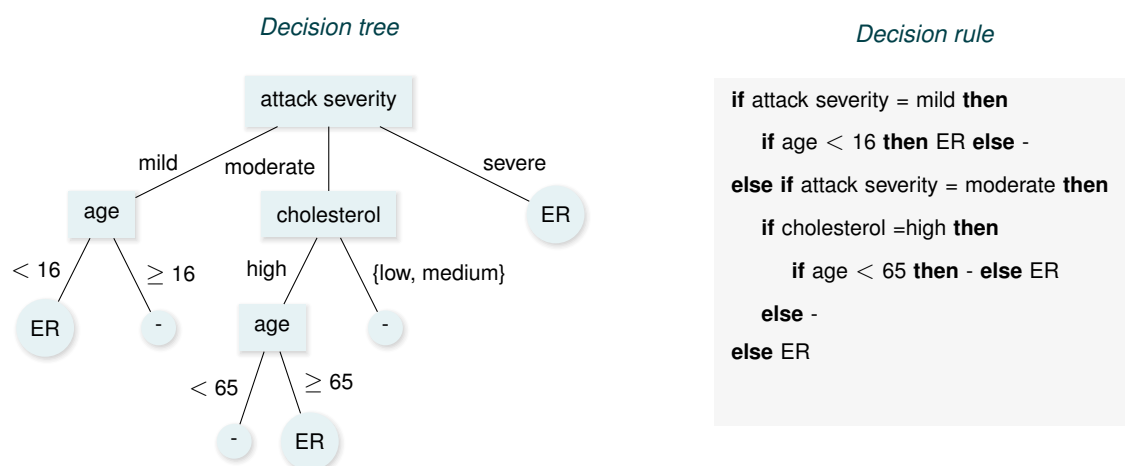


Figure A.2 Example decision tree on ER admission with corresponding if-then-else decision rule

Decision trees (DT). Decision trees consist of a set of *nodes* connected by *branches*, where based on the values of the input features a sequence of tests is performed from the *root* node to a *leaf* to reach a decision. The decisions in the leaves capture exhaustive, mutually exclusive regions of the input space and the corresponding (probability of a) decision for data points in that decision region. For classification trees, the decision for a leaf region is a class label, while in regression trees it typically is the mean response of the training instances in that region [80]. While they are predominantly used for their graphical representation, decision paths from the DT root to a node can be represented as *if-then-else* rules, and can therefore be converted to some DR representations [51]. Conversely, if-then-else decision rules can be represented as decision trees. Figure A.2 depicts an example DT on whether a patient needs to be admitted to the emergency room (ER) or not, along with the corresponding conversion to a DR. In this DT, the top node *a* is the root and the leaves are shown as circles.

While the example in Figure A.2 shows more than two branches as a split after a node, the most popular implementations (e.g., CART and C4.5) of *decision tree induction*—the learning of a DT model—typically only create a single binary split per node (i.e., have two outgoing branches). These approaches are based on Hunt’s algorithm: a depth-first greedy algorithm where each split recursively subdivides the data set into mutually exclusive decision regions by only considering a single feature, until all instances in a single node are of one class—making that node a leaf.

A desirable property of DTs is that they only consider a minimal number of relevant attributes for their decisions [51], by only considering the most informative splits and including tunable hyperparameters for minimizing a tree size. The optimal splits are decided on by finding splits that most cleanly divide the considered data into homogeneous groups of maximal size. For instance, for a binary classifier with 10 instances (5 of class one, 5 of class two) we prefer a split where all 5 of class one are distinguished, over one where 2 of class one and 3 of class two are put into one decision region. This concept, referred to as *node purity*, can be computed using various measures, such as the *residual sum of squares* (RSS) for regression, and the *resubstitution error*, *Gini index* or *entropy* for classification.

As the objective of DTs is to divide the input space into decision regions that most purely split the data, they are prone to *overfitting* on the training data. For instance, in their last splits they may only create a decision boundary between two individual instances. To avoid overfitting, algorithms include hyperparameters that can be set—like early stopping when a given number of instances is in a node, or a maximum tree depth. The downside of early stopping rules is that they may not recognize situations where combinations of attributes are informative instead of a split on a single attribute. Therefore, we might prefer to first induce a DT that overfits slightly on the data but recognizes these complex splits. To combat overfitting, *DT pruning* can cut back the nodes that are irrelevant—replacing uninformative branches with leaf nodes [158].

Decision rules (DR). Decision rules take on the form of an *antecedent* and *consequent*, where if the expression in the antecedent is true then the consequent also holds. In other words, they are of the general form if antecedent then consequent. In some types of decision rules, else indicates how the decision process continues when the antecedent does not hold. Their natural language

representation often makes them a preferred choice when interpretability is required [79].

Decision rules take on a multitude of forms, such as *decision sets* (if-then rules), *decision lists* (if-then-else rules) and *M-of-N rules* (where M of the N antecedents must hold to result in a given consequent) [100, 151, 186]. Graphically, a decision rule set may also be represented as a *decision table* or *decision tree* [79]. Their less restrictive nature over decision trees poses the benefit that they do not enforce mutually exclusive rules [79]. Similar to decision trees, some rule generation algorithms may construct oblique, conjunctive, disjunctive, and negated expressions.

Neural Networks (NN)

Artificial neural networks are a class of techniques that mimic the behavior of neurons in the biological brain, where an *artificial neuron (node)* fires (i.e., passes on a signal) based on the inputs it receives. This mapping from inputs to outputs is done using an *activation function* $g(\cdot)$ for a node j . Neurons are organized in a network, and connected by directed *links* that serve to propagate the activation from a node i to node j . Each link has a numeric weight $w_{i,j}$ attached to it, that determines the strength and sign of the connection. In addition, the neuron has a *bias* b_j for when all weights are zero—similar to the intercept term of regression. The neuron computes the *input function*, a weighted sum of its n input links

$$in_j = \sum_{i=1}^n w_{i,j} a_i + b_j \quad (\text{A.19})$$

and takes this as the value to determine its output *activation* a_j by applying the activation function to its input $a_j = g(in_j)$ [158]. Popular activation functions include the *binary step function*, *sigmoid function*, *tanh function* and *rectified linear unit* (ReLU).

Architectures. Even though a single neuron in a network may represent a simple relationship between inputs and outputs, the power of NNs lies in connecting multiple neurons to form a network. When deployed, NNs have a decided upon structure that is responsible for mapping its inputs into outputs, called the *neural network architecture*. Different architectures make NNs able to support a wide variety of ML tasks, within (semi-)supervised learning, unsupervised learning and reinforcement learning.

The neurons in a neural network are organized in *layers*. We denote the neurons that receive input for the network as the *input layer*, while the neurons that produce output are the *output layer* [158]. In the final layer of the NN, functions can be used to restrict its output, such as the *softmax* function that ensures that the sum of all outputs adds to one—resulting in a probability per output neuron that can be used for probabilistic classification.

There are two distinct approaches to connecting neurons together in a network: *feed-forward* and *recurrent*. First, *feed-forward* NNs form a directed acyclic graph—where neurons receive inputs from upstream nodes and pass them down to downstream ones. The first layer are a fixed number of inputs, while the final layer are the network outputs. Thus, they are used for tasks with fixed input and output sizes, such as supervised learning on tabular data or image classifica-

tion. *Deep neural networks* (DNNs) organize the neurons in the network in multiple layers [129]. Unlike their shallow counterparts, they contain multiple *hidden layers*—i.e., neurons that are not connected to the outputs of the network. These multi-layer networks are able to represent more complicated functions [158]. Figure A.3 shows example architectures for shallow and deep feed-forward NNs, each with three inputs i and two outputs o . The unnamed middle nodes represent hidden neurons.

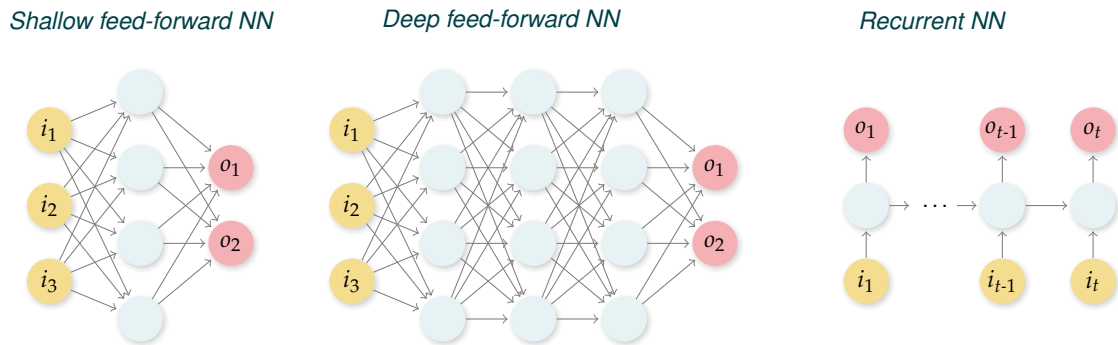


Figure A.3 Example architectures of shallow and deep feed-forward NNs, and an example unfolded recurrent NN architecture with t time steps

Second, in contrast to feed-forward NNs *recurrent NNs* (RNNs) feed their outputs back into its own inputs [158]. Each activation level depends on the previous activation, making them able to form a dynamic system with a non-fixed number of inputs and outputs. In addition, the knowledge of previous activations makes them able to support short-term memory. However, note that their opaque structure also makes the less intelligible. Typical applications of RNNs are ones that do not have a fixed number of inputs or outputs, such as speech recognition, (visual) question answering, natural language generation, and machine translation. Figure A.3 shows an RNN with n inputs and outputs, that is unfolded to show each state separately. Note that different RNN architectures are also supported:⁴

- *One-to-many* RNNs have a single output and many outputs (e.g., caption generation for a single image);
- *Many-to-one* RNNs have multiple inputs and a single output (e.g., sentiment analysis), and;
- *Many-to-many* RNNs with many inputs and many outputs (e.g., time series prediction).

Short-term memory in RNNs requires special units that are able to keep track of the internal state of the network. These units use logic gates that conditionally process and pass on signals based on previous states and inputs. *Long short-term memory* (LSTM) [77] units comprise a memory cell, and three neurons—the input gate, output gate and forget gate. As the name suggests, the memory cell responsible for remembering values over an arbitrary time period. As a result, it can capture long-time dependencies between inputs and outputs. The three gates regulate the memory cell. They are responsible for deciding which values in the cell to update, which ones to

⁴ While technically one-to-one RNNs exist, they are functionally equivalent to feed-forward NNs. To avoid confusion, we refer to them as feed-forward NNs.

output and which ones to forget, respectively. Alongside the previous state, an LSTM also passes on a signal of how much to forget in the next state. *Gated recurrent units* (GRUs) [27] are recent, simplified version of LSTM that do not require a memory cell, but rather only two gates: update and reset.

The NNs we considered so far do not scale well to a large number of inputs. For example, small images with three channels (red, green and blue) of size 100×100 followed by a fully connected layer of 100 neurons already require three million weights. This not only increases the memory needed to store the network, but also the amount of training data that is necessary to train the network. In addition, feed-forward architectures disregard the structure of the inputs, while inputs in for instance images are highly correlated—representing structures such as objects, corners or edges. *Convolutional NNs* (CNNs) [102] address these issues using layers with convolution units. Convolution units consider a small neighborhood of neurons (e.g., an input section of 20×20 of an image) and combines them into a smaller output—the convolved feature—by multiplying the input weights with a filter (*convolution*) and then extracting a single weight for the filtered weights (*pooling*). Typical use-cases for CNNs are object recognition and image segmentation. Figure A.4 shows an example architecture for a CNN that convolves and pools the 64 inputs into two convolved feature matrices in the third layer, which are then fed into a feed-forward NN.

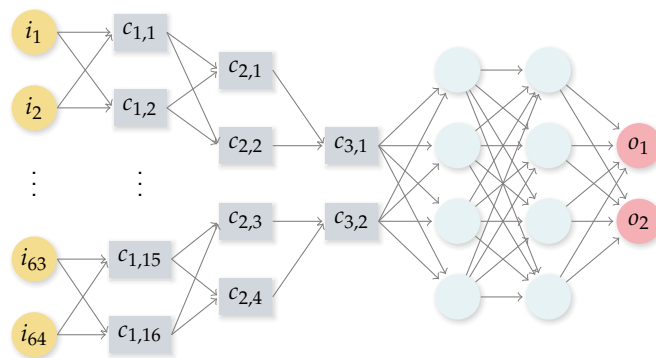


Figure A.4 Convolution neural network (CNN) architecture example

Learning weights and biases. NNs learn to perform a task by considering examples and then finding weights and biases that transform the inputs into a desired output. *Backpropagation* is a form of *gradient descent* search that minimizes the sum of squares (see Eq. A.13) error by iterating over the examples [141]. In the forward pass it obtains the outputs for the given weights and biases, and then in the backward pass it finds the corresponding errors for all output and hidden neurons and updates them to minimize the error.⁵

Ensemble methods

Ensemble methods combine multiple predictions for an input made by base predictors to produce a prediction for an input [158]. By combining predictions, the likelihood of an incorrect prediction

⁵ While there are other approaches to find weights and biases, such as *genetic algorithms*, backpropagation is the main approach taken for decompositional iML methods for opening up NNs.

decreases—improving the generalizability and robustness of an ensemble method over a single predictor. For instance, when using simple majority voting where out of five base learners four found the instance to be of class A instead of B , it is more likely that it is indeed class A rather than we used a single base learner that could have simply predicted B . The most popular application of ensembles are *tree ensembles*, that train multiple decision trees on data (for classification or regression) and aggregate them into a single model so that when they applied their predictions are consolidated into a single prediction.

There are two widely-used approaches to ensemble learning [158]:

- ▶ **Bagging** combines predictions learned from multiple bootstrap data sets, each generated by uniformly sampling from the original data set with replacement. It trains a base learner on each bootstrap data set. Bagging models are combined by performing a majority vote on the new instance. A popular method using bagging is *Random Forests*.
- ▶ **Boosting** incrementally builds a predictor by adjusting the learning algorithm based on the weights of instances in a weighted training set. If an instance is misclassified or its prediction is far from the truth, the instance will receive a higher weight. Because the instances are weighted, the predictor will consider some instances more often—ensuring that previously wrongly predicted instances will sequentially be predicted more correctly. The final model is a weighted average of all earlier models. Widely used implementations of boosting include *AdaBoost* and *XGBoost*.

Reinforcement learning

The goal of an RL agent is to perform *planning* in an environment—i.e., execute a series of actions with a maximal expected reward. This can be done through two approaches. *Model-based* RL builds an explicit model of the environment. The policy can then simply be executed by following the sequence of actions most appropriate according to the model. However, we are also able to find the optimal policy without having an explicit model: *model-free* RL. Model-free RL estimates the optimal policy directly from experience. This is done through either iteratively finding the policy directly (*policy learning*) or finding values for the expected optimal actions in a given state (*value learning*). The remainder of this section will focus specifically on value learning.

Value learning. To evaluate a policy, we use a *value function* $V^\pi(s)$ —the expected return of following a policy π starting from state s —, defined as

$$V^\pi(s) = E_\pi\{R_t \mid s_t = s\} \tag{A.20}$$

where R_t is the discounted future reward from current state t onwards (defined in Eq. A.10) and $E_\pi\{\cdot\}$ is the expected reward given that the agent will follow policy π [176].

Recall from Appendix A that RL tasks are either *episodic* (tasks that end in a terminal state) or *continuous* (tasks that do not). *Monte Carlo* learning updates the value function when reaching the terminal state (full look-ahead). It cannot learn the value function for continuous tasks, because

they do not have a guarantee of a terminal state. Instead, *Temporal-Difference* (TD) learning only considers the current state and follow-up action, and incrementally builds a model (one-step look-ahead). The additional benefit of TD learning, alongside its application to both episodic and continuous tasks, is that TD learning is able to react quickly to recent trends [176].

To find an optimal action-selection policy, we need to ensure that all actions are selected infinitely often. Therefore, each of these learning techniques is used with either *on-policy* methods—where it attempts to use and evaluate the same policy that they use to make decisions—, or *off-policy* methods—using two distinct policies: one for behavior and one for value estimation [176].

Q-learning. Q-learning is an off-policy TD learning algorithm that is used to learn the Q-function. The *Q-function* $Q^\pi(s, a)$ denotes the expected reward when taking an action a in state s under policy π :

$$Q^\pi(s, a) = E_\pi\{R_t \mid s_t = s, a_t = a\} \quad (\text{A.21})$$

We obtain the values for the Q-function for the state-action pairs by iteratively updating their Q-values. First, we initialize it arbitrarily (e.g., randomly or with zeros), then take an action $a \in A_s$ in state s and observe its direct reward r_{t+1} . In addition, we take into account the maximal Q-value of the state we end up in after taking the action, denoted by $\max Q(s_{t+1}, a)$. Jointly, these parts update the Q-function for the taken state-action pair by updating its current value as

$$Q(s_t, a_t) += \alpha[r_{t+1} + \gamma \max Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (\text{A.22})$$

where $\alpha \in [0, 1]$ is the *learning rate* (i.e., how much the new information overrides the new information) and γ denotes the discount factor. For fully stochastic events a learning rate of one is optimal, while for deterministic events the learning rate should decrease to zero. The optimal actions can then easily be deduced by selecting in each state the action with the highest expected Q-value

$$\arg \max_{a \in A_s} Q^\pi(s, a) \quad (\text{A.23})$$

where A_s are all actions possible in state s .

While originally the state-action value pairs were represented in a *Q-table*, this representation suffers from the same curse of dimensionality of traditional NNs. For use with large state spaces (e.g., image data), *Deep Q-Networks* (DQN) [125] can handle vast amounts of data using convolutions, while still performing Q-learning on tasks. They pass each state through a CNN to obtain the Q-values for that state, and the reward for the next state, and then update the Q-values as well as the CNN itself.

C Validation Data

This appendix shows the individual results for each data set-model pair.⁶ Table A.3 summarizes the data set-model pair performance, confidence, local fidelity, accuracy, fidelity and time per data point needed to form a contrastive explanation. Table A.4 details per data set-model pair the length of the contrastive explanation (C-Length) in terms of decision nodes, length of the non-contrastive explanation (NC-Length), and the improvement of the former over the latter. In addition, it shows the number of features for each data set as the upper bound of explanation length.

Table A.3 Data set-model pair performance, confidence, local fidelity, accuracy, fidelity and time results

Data set	Model	Perform.	Confidence	Loc. fid.	Accuracy	Fidelity	Time (s)
IRIS	RF	0.93	0.89	0.99	0.99	0.99	0.0262
	LR	0.93	1.00	1.00	0.96	1.00	0.0205
	SVM	0.93	0.77	0.99	0.98	0.97	0.0186
	NN	0.97	1.00	1.00	0.98	1.00	0.0238
DIABETES	RF	1.00	0.97	0.98	1.00	1.00	0.1667
	LR	1.00	1.00	1.00	0.99	0.99	0.0208
	SVM	1.00	0.97	0.98	1.00	1.00	0.0228
	NN	0.95	1.00	0.99	0.99	0.99	0.0230
HEART DISEASE	RF	0.94	1.00	0.99	0.92	0.92	0.0263
	LR	1.00	1.00	1.00	0.99	0.99	0.0196
	SVM	1.00	0.74	0.96	0.86	0.86	0.0200
	NN	1.00	1.00	0.99	0.92	0.92	0.0408
SPECT	RF	1.00	1.00	1.00	1.00	1.00	0.0471
	LR	1.00	1.00	1.00	1.00	1.00	0.0503
	SVM	1.00	1.00	1.00	1.00	1.00	0.0541
	NN	1.00	1.00	1.00	1.00	1.00	0.0524
CENSUS INCOME	RF	0.53	0.86	0.98	0.78	0.78	0.2312
	LR	0.62	0.99	0.99	0.70	0.86	0.0323
	SVM	0.64	0.96	0.96	0.67	0.8	0.0706
	NN	0.67	0.87	0.98	0.74	0.8	0.0842
WINE QUALITY	RF	0.36	0.96	0.99	0.69	0.79	0.1433
	SVM	0.26	0.96	0.99	0.57	0.85	0.0376
	NN	0.27	0.97	0.99	0.64	0.84	0.0151
PARKINSON	RF	1.00	1.00	1.00	1.00	1.00	0.0600
	SVM	0.96	0.99	1.00	0.61	0.96	0.0507
	NN	1.00	1.00	1.00	1.00	1.00	0.1259
STUDENT	RF	0.87	0.98	1.00	0.89	0.95	0.1254
	SVM	0.82	0.98	1.00	0.86	0.96	0.0491
	NN	0.73	0.99	0.99	0.92	0.93	0.0808
<i>Average</i>	–	–	0.96	0.99	0.88	0.94	0.0600

⁶ The full source code for the quantitative validation (benchmark) is available online at <https://github.com/MarcelRobeer/ContrastiveExplanation-experiment>

Table A.4 Data set-model pair length results

Data set	Features	Model	C-Length	NC-Length	Improvement
IRIS	4	RF	1.01	1.73	0.72
		LR	1.02	1.33	0.31
		SVM	1.04	2.00	0.96
		NN	1.57	2.38	0.81
DIABETES	7	RF	1.11	6.39	5.28
		LR	1.54	5.48	3.94
		SVM	1.54	5.00	3.46
		NN	1.66	6.87	5.21
HEART DISEASE	13	RF	1.42	6.46	5.04
		LR	1.38	5.46	4.08
		SVM	1.14	5.14	4.00
		NN	1.32	3.79	2.47
SPECT	45	RF	1.26	6.95	5.69
		LR	1.26	6.23	4.97
		SVM	1.26	4.84	3.58
		NN	1.26	6.95	5.69
CENSUS INCOME	108	RF	1.12	1.00	-0.12
		LR	1.01	1.00	-0.01
		SVM	0.98	1.00	0.02
		NN	1.13	1.00	-0.13
WINE QUALITY	11	RF	1.08	10.46	9.38
		SVM	0.96	9.98	9.02
		NN	1.00	10.25	9.25
PARKINSON	28	RF	1.00	1.00	0.00
		SVM	1.13	8.50	7.37
		NN	1.00	8.07	7.07
STUDENT	58	RF	1.1	9.06	7.96
		SVM	1.02	7.91	6.89
		NN	1.23	9.44	8.21
<i>Average</i>			1.19	5.37	4.18

D Experiment Materials

This appendix contains the questionnaire materials. First, we overview the agenda items used for the preference and agreement parts of the experiment. Next, we include the questionnaire pages shown to the participants.

Agenda items and explanations

Table A.5 summarizes the ten agenda items for the preference part, and the six agenda items for the agreement part. Table A.6 presents the corresponding predictions and explanations. Facts (predictions made by the model) highlighted in red indicate that it is a wrong prediction according to the actual label.

Table A.5 Experiment agenda items

Agenda item	Time	Food	Agenda	Health (past hr)
<i>Preference</i>				
0	11:52	chips (4g CHO)	-	hypo
1	14:47	apple (9g CHO)	14:13 14 minuts of bicycling (16kph, sport)	-
2	14:23	banana (13g CHO)	-	-
3	19:1	ham cheese tosti (4g CHO)	19:53 38 minutes of walking	hyper
4	20:7	muesli bar (5g CHO)	19:50 69 minutes of walking	hyper
5	11:51	spaghetti (7g CHO)	12:24 37 minutes of chores	-
6	11:52	grapes (16g CHO)	-	-
7	20:49	muesli bar (10g CHO)	20:11 25 minutes of dancing (sport)	-
8	21:38	lasagna (13g CHO)	19:39 67 minutes of skating (sport) & 22:15 107 minutes of walking	hyper
9	19:45	pizza (8g CHO)	19:29 16 minutes of chores	hyper
<i>Agreement</i>				
10	12:17	grapes (14g CHO)	11:30 16 minutes of dancing (sport)	hypo
11	14:37	chips (3g CHO)	-	-
12	20:35	chips (2g CHO)	19:48 19 minutes of skating (sport)	hyper
13	11:51	spaghetti (7g CHO)	12:24 37 minutes of chores	-
14	15:39	apple (14g CHO)	-	-
15	19:15	carrots & potatoes (16g CHO)	20:01 41 minutes of running (sport)	-

Table A.6 Prediction and explanation per experiment agenda item

Agenda item	Prediction			Explanation	
	Actual	Fact (pred.)	Foil	Contrastive	Non-Contrastive
<i>Preference</i>					
0	hypo	hypo	ok	had a hypo	had lunch (after 11.30) and spent less than 5 minutes on activities had lunch (after 11.30)
1	ok	ok	hypo	did not perform another activity and did not have a hypo	had lunch (after 11.30)
2	ok	ok	hyper	her food did not contain more than 21g CHO	had lunch (after 11.30) and spent less than 5 minutes on activities and did not have a hypo
3	hyper	hyper	ok	did not perform sports and had dinner (after 16.00)	had food after 15.00 and performed sports
4	hyper	hyper	ok	did not perform sports and ate a muesli bar (after 16.00)	had food after 15.00 and ate spaghetti
5	hypo	ok	hypo	did not perform another activity and did not have a hypo	had lunch (after 11.30)
6	hypo	hypo	ok	ate grapes	had lunch (after 11.30) and spent less than 5 minutes on activities and did not have a hypo
7	hyper	hyper	ok	performed dancing and had dinner (after 16.00)	had food after 15.00 and performed dancing
8	hyper	hyper	ok	performed two activities and ate lasagna	had food after 15.00 and did not eat spaghetti
9	ok	hyper	ok	did not perform sports and had dinner (after 16.00)	had food after 15.00 and performed chores
<i>Agreement</i>					
10	ok	ok	hypo	did not perform another activity	had food before 11.30
11	ok	ok	hypo	did not have a a hypo	had food before 11.30 and performed an activity more than 3 hours before eating and did not have a hypo
12	hyper	hyper	ok	did not perform another activity and had a hyper	had dinner (after 15.00) and ate spaghetti
13	hypo	ok	hypo	did not perform another activity and did not have a hypo	had food after 11.30
14	hyper	ok	hyper	did not have a hyper	had an apple after 15:00 and did not perform sports
15	ok	hyper	ok	did not perform another activity	ate carrots and potatoes and the food contained more than 12g CHO

Questionnaire pages

The pages below include the following pages of the questionnaire (the pages containing the agenda items are omitted for brevity reasons):

- ▶ Introduction
- ▶ Informed Consent (IC)
- ▶ Demographics (*5 questions*)
- ▶ Diabetes management familiarity (*3 questions*)
- ▶ Scenario & type I diabetes mellitus (T1DM) management explanation (as shown before and after the test quiz)
- ▶ Test quiz (*9 questions, filled in with correct answers*)
- ▶ Preference part:
 - Introduction
 - Example
 - Pairwise comparison (placeholder)
 - Reason for preference
- ▶ Agreement part:
 - Introduction
 - Example
 - Agreement cases (placeholder)
 - Reason for (dis)agreement
- ▶ Thank you, general remarks

Introduction

Thank you for your interest in our online experiment. The experiment will take approximately 45 minutes.

Diabetes management

Type 1 Diabetes Mellitus (T1DM) is a chronic condition where the body is unable to produce enough insulin, the hormone that helps move glucose (sugar) into your body's tissues. This glucose is used to fuel your body's cells. With insufficient insulin, the body is unable to get glucose into the cells where it is needed, which makes the blood glucose levels very high and causes health problems.

When the blood sugar in your body is too low we refer to it as a hypo, while a too high blood glucose level is called a hyper. Both are damaging to the body. Hypos are characterized by sweating, fatigue, difficulty to concentrate and hungriness. Hypers result in thirst, increased urination, nauseousness (feeling sick), and a bad mood.

Important elements to manage the disease are ones' nutrition and physical activity and exercise. Food and drink can increase the glucose levels, depending on the amount of carbohydrates (CHO) in the food. Exercise decreases glucose levels, depending on the intensity and length of the exercise.

About the experiment

Your task is to make judgments about suggested decisions by an intelligent app that supports a diabetes patient, for various agenda items of that patient. This study constitutes two parts. In the preference part you will be given 30 agenda items for a diabetes patient, and two decisions corresponding to this agenda item. Some decisions will include a rationale of why this decision was made. You will be asked which decision you find preferable in general, and preferable regarding four value judgments. In the agreement part you will be provided with 18 agenda items. For each agenda item, you will be asked whether you agree with the decision made for that agenda item.

Informed Consent

Before taking part in this study, please read the consent form below. Continue to the next page if you understand the statements and freely consent to participate in the study.

Consent Form

Your responses will be collected anonymously and are kept confidential. There are no known or anticipated risks associated with this study. You are free to withdraw and stop at any moment of the experiment, and will receive no penalty if you decide to do so.

Participation in this study typically takes 45 minutes.

If participants have further questions about this study or their rights, or if they wish to lodge a complaint or concern, they may contact the researchers at: **removed for anonymity**

By continuing to the next page you indicate that you are at least 18 years old, have read and understood this consent, and voluntarily participate in this study.

Demographics

We first ask you to fill in the questions below, so that we can collect some general demographics about the people performing our experiment.

* What is your sex?

- Male
- Female
- Prefer not to say*

* What is your age?

- 18 to 24
- 25 to 34
- 35 to 44
- 45 to 54
- 55 to 64
- 65 to 74
- 75 or older

* In what region do you currently reside?

- Africa
- Americas
- Asia
- Europe
- Oceania

*

What is the highest degree or level of school you have completed?
If currently enrolled, highest degree received.

- Less than high school degree
- High school degree or equivalent (e.g., GED)
- Some college but no degree
- Associate degree
- Bachelor degree
- Master's degree
- Other (please specify)

*

How well do you read and write English?

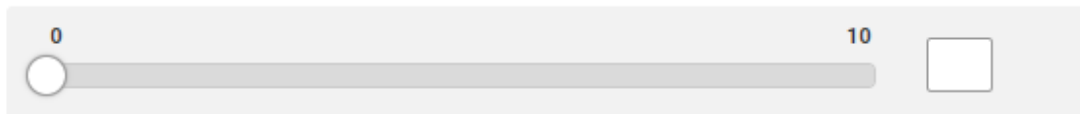
Not well at all	Not very well	Well	Pretty well	Very well
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Diabetes management familiarity

Before we start the experiment we would like to gather some insights about your familiarity with diabetes and managing diabetes.

* How familiar are you with diabetes management?

0 10



* Do you have a family history of diabetes?

- Yes
- No
- Prefer not to say*


* Is your occupation or study related to healthcare?
(for example nurse, studying medicine)

- No
- Yes, namely...

Experiment

Please read carefully and take notes if necessary.

Emily is a type 1 diabetes mellitus (T1DM) patient. Her profile is as follows:

NAME	Emily
PICTURE	
GENDER	Female
AGE	16
WEIGHT	68.7kg (151.5 lbs)
HEALTH	Type I diabetes mellitus (T1DM) since childhood
HOBBIES	Bicycling Dancing Skating
FAVORITE FOODS	Italian cuisine Fruit

Task

Emily has an intelligent app that helps her to predict *each time she eats* what her *health status will be in the next hour*. For the food item she chooses to eat, the app will give her a decision whether she will have a **hyper** (high glucose), **hypo** (low glucose), or remain **ok**. Imagine you are Emily, and the app gives the decision for the food item you choose.

You will be given 16 moments in the agenda of Emily. For each agenda item (see image below), you will be provided with:

1. The **current time** (when she wants the app to make a decision)
2. The **food item** she wants to eat at this time, and the number of grams of carbohydrate (CHO)
3. The **activities** she performed (within one hour before eating), and is planning to perform (within one hour after eating)
4. Whether she had a **hyper** (too high blood sugar) **or hypo** (too low blood sugar) during the last hour

5. A decision of whether her health status in the next hour results in a:

- *Hyper* (too high blood sugar)
- *Hypo* (too low blood sugar)
- *Ok* (neither hyper nor hypo, the ideal situation)

Your task is to make a judgment about the decision made.

The image below shows an **example agenda item (as you will be shown)**:

1	Time Emily asks for a decision		Current time	17:01
2	Food she wants to eat at current time		Food	Cereal & milk (40g CHO)
3	Activities performed and planned		Agenda	16:35 43 minutes of skating (sport)
4	Health status in hour before current time		Health	Had hyper (high blood sugar) in the past hour

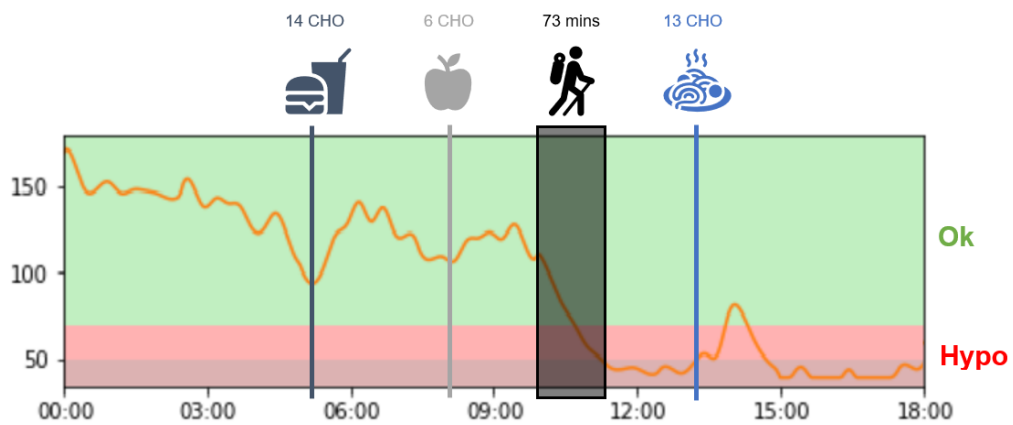
For this agenda item, the **prediction of the intelligent app is 'Hyper'**.

Expected Behavior

In general, you can expect the following effects on the prediction:

- *Food* will **increase glucose**, where more grams of Carbohydrate (CHO) increase the glucose level more
- *Activities* will **decrease glucose**, where more intense exercise (e.g., sports) and longer exercise will decrease glucose levels more
- Previous health status will result in similar outcomes:
 - a *hyper before food* increases the **chance of a hyper**
 - a *hypo before an activity* increases the **chance of a hypo**

Example Agenda



In the example agenda above, the orange trend line indicates the *blood glucose* (mg/dL) during the day for *Emily*. Her agenda has four items:

- 04:55 - She eats a hamburger meal (14g CHO)
- 08:12 - She eats an apple (6g CHO)
- 09:59 - She performs hiking for 73 minutes
- 13:17 - She eats spaghetti (13g CHO)

Observe that *hiking* results in a **hypo** after exercise (red area), while *eating spaghetti* results in **ok** (neither a hypo or hyper; green area) in the hour after eating, even though she had a hypo in the hour before eating.

TEST: Scenario

A small test of whether you were able to recall some details about the scenario of diabetes patient Emily and diabetes management. Performing well in this test allows you to gain a bonus reward.

Each question is worth 1 point (total of 9 points), where you can gain a bonus if you obtain 5 or more points.

- * How do carbohydrates (CHO) affect blood glucose levels?
- More CHO decreases blood glucose levels more
 - CHO has no effect on blood glucose levels
 - More CHO increases blood glucose levels more
- * Which factors determine the size of the effect of exercise on blood glucose?
- Duration (longer = larger effect)
 - Duration (shorter = larger effect)
 - Intensity (less intense = larger effect)
 - Intensity (more intense = larger effect)
- * What is the age of *Emily*?
- 16
- * What is the expected effect of a *hyper before eating*?
- It increases blood glucose
 - It decreases blood glucose
 - It increases the chance of a hyper after eating
 - It decreases the chance of a hyper after eating

*

What is a hyper?

- Too high blood glucose level
- Too low blood glucose level

*

What is the expected effect of exercising on your blood glucose level?

- Blood glucose will increase
- Blood glucose will decrease

*

This was **not** one of *Emily's* hobbies...

- Walking
- Skating
- Bicycling
- Dancing

*

When will the prediction of the intelligent app be 'ok'?

- When Emily will not have a hypo in the next hour
- When Emily will not have a hyper in the next hour
- When Emily will not have a hyper **and** will not have a hypo in the next hour

Part: Preference

In this part of the experiment, you are tasked with deciding your preference for Decision A or Decision B regarding five (5) statements. You can also have no preference when you judge both decisions to be equal. However, even with a slight preference we urge you to choose between Decision A and Decision B.

You will be shown 30 preference judgments.





Click next to see an example.

Part: Preference (example)

You are first given an agenda item about a *food item* Emily wants to consume at a certain *time*. In addition, this agenda item contains activities for the hour before and after eating, and the health status of Emily in the past hour.

The decision is a prediction of whether Emily will have a hypo or hyper in the next hour, or will be ok (neither hypo nor hyper). You are given two decisions, and asked to determine your preferences for one decision over the other regarding five (5) statements.

1. Agenda item (provided by Emily to the intelligent app)

	Current time	17:01
	Food	Cereal & milk (40g CHO)
	Agenda	16:35 43 minutes of skating (sport)
	Health	Had hyper (high blood sugar) in the past hour

2. Two decisions (both made by the intelligent app)

Decision A	Decision B
We predict Hyper	We predict Hyper instead of Ok, because she ate cereal & milk

3. Determine your preferences for five (5) statements





* Which decision do you prefer regarding the following statements?

	Decision A	No preference	Decision B
Preference: In general, I prefer this decision	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Transparency: I understand this decision more clearly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Contents: The considerations made in this decision are more similar to the ones I would use	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Alignment: The way the decision is presented aligns with how I would make this decision	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Persuasiveness: I am more likely to act upon this decision	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Click next if you understand the task, and want to start the experiment.

Part: Preference

Finally, for one preference we ask you **why** you prefer one decision over the other.

	Current time	17:01
	Food	Cereal & milk (40g CHO)
	Agenda	16:35 43 minutes of skating (sport)
	Health	Had hyper (high blood sugar) in the past hour

A 33.33%

Decision A	Decision B
We predict Hyper	We predict Hyper instead of Ok, because she ate cereal & milk

B 33.34%

Decision A	Decision B
We predict Hyper instead of Ok, because she ate cereal & milk	We predict Hyper, because she had food before 15.00 and ate cereal & milk

C 33.33%

Decision A	Decision B
We predict Hyper	We predict Hyper, because she had food before 15.00 and ate cereal & milk

*

Which decision do you prefer regarding the following statements?

	Decision A	No preference	Decision B
Preference: In general, I prefer this decision	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Transparency: I understand this decision more clearly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Contents: The considerations made in this decision are more similar to the ones I would use	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Alignment: The way the decision is presented aligns with how I would make this decision	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Persuasiveness: I am more likely to act upon this decision	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

*

What are the main reasons for your preference for one explanation over the other?

- It was longer in length.
- It was shorter in length.
- It contained more information.
- It contained less but more interesting information.
- It made a comparison between two predictions.
- I understood it more clearly.
- The contents were more similar to the elements I would use.
- It aligns more with the way I make decisions.
- I was more likely to act upon that decision.
- Other (please specify)

Part: Agreement

In this part of the experiment, you are tasked with deciding whether you **agree** or **disagree** with the decision made (i.e., prediction for the next hour of whether the patient will have a hyper, hypo or be ok) for the agenda item.

You will be shown 18 decisions.

Click next to see an example.

Part: Agreement





You are first given an agenda item about a food item consumed at a certain time. In addition, this agenda item contains activities for the hour before and after eating, and the health status in the past hour.

The decision is a prediction by the intelligent app of whether Emily will have a hypo or hyper in the next hour, or will be ok (neither hypo nor hyper). You are given a decision, and asked whether you **agree** or **disagree** with this decision.

Recall that this is the expected behavior:

- Food will increase glucose, where more grams of Carbohydrate (CHO) increase the glucose level more
- Activities will decrease glucose, where more intense exercise (e.g., sports) and longer exercise will decrease glucose levels more
- Previous health status will result in similar outcomes:
 - a hyper before food increases the chance of a hyper
 - a hypo before an activity increases the chance of a hypo

1. Agenda item (provided by Emily to the intelligent app)

	Current time	17:01
	Food	Cereal & milk (40g CHO)
	Agenda	16:35 43 minutes of skating (sport)
	Health	Had hyper (high blood sugar) in the past hour

2. Decision (made by the intelligent app)

Decision

We predict Hyper

3. Determine whether you agree or disagree with the decision

* Do you agree with this decision for the agenda item?

Yes (agree) No (disagree)

Click next if you understand the task, and want to start the experiment.

Part: Agreement

TNO innovation
for life

18 (dis)agreements (random order)

Part: Agreement

Finally, for one decision we ask you **why** you choose to agree/disagree with that decision.



Current time

17:01



Food

Cereal & milk (40g CHO)



Agenda

16:35 43 minutes of skating (sport)



Health

Had hyper (high blood sugar) in the past hour

A 33.34%

Decision

We predict Hyper instead of Ok, because she ate cereal & milk

B 33.33%

Decision

We predict Hyper, because she had food before 15.00 and ate cereal & milk

C 33.33%

Decision

We predict Hyper

*

Do you agree with this decision for the agenda item?

Yes (agree)

No (disagree)

*

What are your main reasons for agreeing/disagreeing? (e.g., agenda item contents, decision contents, expected different prediction)

Thank you!

Thank you for completing the experiment! The answers you provided are anonymous and kept confidential. If you have any comments regarding our study, please do not hesitate to fill in the form below. If you have any further questions, please contact the researchers at: **removed for anonymity**

Any other general remarks or feedback regarding this survey?

PUBLICATIONS

The first paper based on this paper was accepted for publication and presented at the *2018 Third Annual Workshop on Human Interpretability in Machine Learning (WHI 2018)* as part of the International Conference on Machine Learning (ICML) in Stockholm, Sweden.

- ▶ J. van der Waa, M. Robeer, J. van Diggelen, M. Brinkhuis, & M. Neerincx, “Contrastive Explanations with Local Foil Trees”, in *2018 Workshop on Human Interpretability in Machine Learning (WHI 2018)*, 2018, pp. 41-47. [Online]. Available: <http://arxiv.org/abs/1806.07470>

Contrastive Explanations with Local Foil Trees

Jasper van der Waa^{*1,2} Marcel Robeer^{*1,3} Jurriaan van Diggelen¹ Matthieu Brinkhuis³ Mark Neerincx^{1,2}

Abstract

Recent advances in interpretable Machine Learning (iML) and eXplainable AI (XAI) construct explanations based on the importance of features in classification tasks. However, in a high-dimensional feature space this approach may become unfeasible without restraining the set of important features. We propose to utilize the human tendency to ask questions like “Why this output (the fact) instead of that output (the foil)?” to reduce the number of features to those that play a main role in the asked contrast. Our proposed method utilizes locally trained one-versus-all decision trees to identify the disjoint set of rules that causes the tree to classify data points as the foil and not as the fact. In this study we illustrate this approach on three benchmark classification tasks.

1. Introduction

The research field of making Machine Learning (ML) models more interpretable is receiving much attention. One of the main reasons for this is the advance in such ML models and their applications to high-risk domains. Interpretability in ML can be applied for the following purposes: (i) transparency in the model to facilitate understanding by users (Herman, 2017); (ii) the detection of biased views in a model (Crawford, 2016; Caliskan et al., 2017); (iii) the identification of situations in which the model works adequately and safely (Barocas & Selbst, 2016; Coglianese & Lehr, 2016; Friedler et al., 2018); (iv) the construction of accurate explanations that explain the underlying causal phenomena (Lipton, 2016); and (v) to build tools that allow

^{*}Equal contribution ¹Perceptual and Cognitive Systems, Dutch Research Organization for Applied Research (TNO), Soesterberg, The Netherlands ²Interactive Intelligence group, Technical University of Delft, Delft, The Netherlands ³Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands. Correspondence to: Jasper van der Waa <jasper.vanderwaa@tno.nl>.

model engineers to build better models and debug existing models (Kulesza et al., 2011; 2015).

The existing methods in iML focus on different approaches of how the information for an explanation can be obtained and how the explanation itself can be constructed. See for example for an overview the review papers of Guidotti et al. (2018) and Chakraborty et al. (2017). A number of examples of common methods are: ordering the feature’s contribution to an output (Datta et al., 2016; Lei et al., 2016; Ribeiro et al., 2016), attention maps and saliency of the features (Selvaraju et al., 2016; Montavon et al., 2017; Sundararajan et al., 2017; Zhang et al., 2017), prototype selection, construction and presentation (Nguyen et al., 2016), word annotations (Hendricks et al., 2016; Ehsan et al., 2017), and summaries with decision trees (Krishnan et al., 1999; Thiagarajan et al., 2016; Zhou & Hooker, 2016) and decision rules (Hein et al., 2017; Malioutov et al., 2017; Puri et al., 2017; Wang et al., 2017). In this study we focus on feature-based explanations. Such explanations tend to be long when based on all features or use an arbitrary cutoff point. We propose a model-agnostic method to limit the explanation length with the help of contrastive explanations. The method also adds information of how that feature contributes to the output in the form of decision rules.

Throughout this paper, the main reason for explanations is to offer transparency in the model’s given output based on which features play a role and what that role is. A few methods that offer similar explanations are LIME (Ribeiro et al., 2016), QII (Datta et al., 2016), STREAK (Elenberg et al., 2017) and SHAP (Lundberg & Lee, 2016). Each of these approaches answers the question “Why this output?” in some way by providing a subset of features or an ordered list of all features, either visualized or structured in a text template. However, when humans answer such questions to each other they tend to limit their explanations to a few vital points (Pacer & Lombrozo, 2017). This human tendency for simplicity also shows in iML: when multiple explanations hold we should pick the simplest explanation that is consistent with the data (Huysmans et al., 2011). The mentioned approaches do this by either thresholding the contribution parameter to a fixed value, presenting the entire ordered list or by applying it only to low-dimensional data.

This study offers a more human-like way of limiting the list of contributing features by setting a contrast between two outputs. The proposed contrastive explanations present only the information that causes some data point to be classified as some class instead of another (Miller et al., 2017). Recently, Dhurandhar et al. (2018) have proposed constructing explanations by finding contrastive perturbations—minimal changes required to change the current classification to any arbitrary other class. Instead, our approach creates *contrastive targeted explanations* by first defining the output of interest. In other words, our contrastive explanations answer the question “Why this output instead of that output?”. The contrast is made between the *fact*, the given output, and the *foil*, the output of interest.

A relative straightforward way to construct contrastive explanations given a foil based on feature contributions, is to compare the two ordered feature lists and see how much some feature differs in their ranking. However, a feature may have the same rank in both ordered lists but can be used in entirely different ways for the fact and foil classes. To mitigate this problem we propose a more meaningful comparison based on how a feature is used to distinct the foil from the fact. We train an arbitrary model to distinguish between fact and foil that is more accessible. From that model we distill two sets of rules; one used to identify data points as a fact and the other to identify data points as a foil. Given these two sets, we subtract the factual rule set from the foil rule set. This relative complement of the fact rules in the foil rules is used to construct our contrastive explanation. See Figure 1 for an illustration.

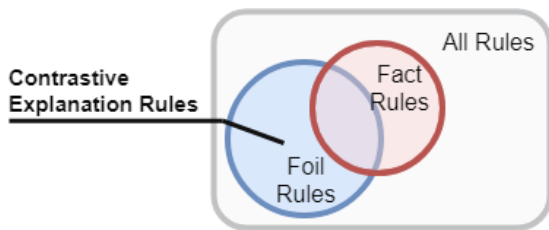


Figure 1. This figure shows the general idea of our approach to contrastive explanations. Given a set of rules that define data points as either the fact or foil, we take the relative complement of the fact rules in the foil rules to obtain a description how the foil differs from the fact in terms of features.

The method we propose in this study obtains this complement by training a one-versus-all decision tree to recognize the foil class. We refer to this decision tree as the Foil Tree. Next, we identify the fact-leaf—the leaf in which the current questioned data point resides. Followed by identifying the foil-leaf, which is obtained by searching the tree with some strategy. Currently our strategy is simply to choose the closest leaf to the fact-leaf that classifies data points as

the foil class. The complement is then the set of decision nodes (representing rules) that are a parent of the foil-leaf but not of the fact-leaf. Rules that overlap are merged to obtain a minimum coverage rule set. The rules are then used to construct our explanation. The method is discussed in more detail in section 2. An example of its usage is discussed in section 3 on three benchmark classification tasks. The validation on these three tasks shows that the proposed method constructs shorter explanations than the fully feature list, provide more information of how these features contribute and that this contribution matches the underlying model closely.

2. Foil Trees; a way for obtaining contrastive explanations

The method we propose learns a decision tree centered around any questioned data point. The decision tree is trained to locally distinguish the foil-class from any other class, including the fact class. Its training occurs on data points that can either be generated or sampled from an existing data set, each labeled with predictions from the model it aims to explain. As such, our method is model-agnostic. Similar to LIME (Ribeiro et al., 2016), the sample weights of each generated or sampled data point depend on its similarity to the data point in question. Samples in the vicinity of the questioned data point receive higher weights in training the tree, ensuring its local faithfulness.

Given this tree, the ‘foil-tree’, we search for the leaf in which the data point in question resides, the so called ‘fact-leaf’. This gives us the set of rules that defines that data point as the not-foil class according to the foil-tree. These rules respect the decision boundary of the underlying ML model as it is trained to mirror the foil class outputs. Next, we use an arbitrary strategy to locate the ‘foil-leaf’—for example the leaf that classifies data point as the foil class with the lowest number of nodes between itself and the fact-leaf. This results in two rule sets, whose relative complement define how the data point in question differs from the foil data points as classified by the foil-leaf. This explanation of the difference is done in terms of the input features themselves.

In summary, the proposed method goes through the following steps to obtain a contrastive explanation for an arbitrary ML model, the questioned data point and its output according to that ML model:

1. **Retrieve the fact;** the output class.
2. **Identify the foil;** explicitly given in the question or derived (e.g. second most likely class).
3. **Generate or sample a local data set;** either randomly sampled from an existing data set, generated

according to a normal distribution, generated based on marginal distributions of feature values or more complex methods.

4. **Train a decision tree**; with sample weights depending on the training point’s proximity or similarity to the data point in question.
5. **Locate the ‘fact-leaf’**; the leaf in which the data point in question resides.
6. **Locate a ‘foil-leaf’**; we select the leaf that classifies data points as part of the foil class with the lowest number of decision nodes between it and the fact-leaf.
7. **Compute differences**; to obtain the two set of rules that define the difference between fact- and foil-leaf, all common parent decision nodes are removed from each rule sets. From the decision nodes that remain, those that regard the same feature are combined to form a single literal.
8. **Construct explanation**; the actual presentation of the differences between the fact-leaf and foil-leaf.

Figure 2 illustrates the aforementioned steps. The search for the appropriate foil-leaf in step 6 can vary. In Section 2.1 we discuss this more in detail. Finally, note that the method is not symmetrical. There will be a different answer on the question “Why class A and not B?” then on “Why class B and not A?” as the foil-tree is trained in the first case to identify class B and in the second case to identify class A. This is because we treat the foil as the expected class or the class of interest to which we compare everything else. In addition, even if the trees are similar, the relative complements of their rule sets are reversed

2.1. Foil-leaf strategies

Up to now we mentioned one strategy to find a foil-leaf, however multiple strategies are possible—although not all strategies may result in a satisfactory explanation according to the user. The strategy used in this study is simply the first leaf that is closest to the fact-leaf in terms of number decision nodes, resulting in a minimal length explanation.

A disadvantage of this strategy is its ignorance towards the value of the foil-leaf compared to the rest of the tree. The nearest foil-leaf may be a leaf that classifies only a relatively few data points or classifies them with a relatively high error rate. To mitigate such issues the foil-leaf selection mechanism can be generalized to a graph-search from a specific (fact) vertex to a different (foil) vertex while minimizing edge weights. The foil-tree is treated as a graph whose decision node and leaf properties influence some weight function. This generalization allows for a number of strategies, and each may result in a different foil-leaf.

The strategy used in this preliminary study simply reduces to each edge having a weight of one, resulting in the nearest foil-leaf when minimizing the total weights.

As an example, an improved strategy may be where the edge weights are based on the relative accuracy of a node (based on its leaves) or leaf. Where a higher accuracy results in a lower weight, allowing the strategy to find more distant, but more accurate, foil-leaves. This may result in relatively more complex and longer explanations, which nonetheless hold in more general cases. For example the nearest foil-leaf may only classify a few data points accurately, whereas a slightly more distant leaf classifies significantly more data points accurately. Given the fact that an explanation should be both accurate and fairly general, this proposed strategy may be more beneficial (Craven & Shavlik, 1999).

Note that the proposed method assumes the knowledge of the used foil. In all cases we take the second most likely class as our foil. Although this may be an interesting foil it may not be the contrast the user actually wants to make. Either the user makes its foil explicit or we introduce a feedback loop in the interaction that allows our approach to learn which foil is asked for in which situations. We leave this for future work.

3. Validation

The proposed method is validated on three benchmark classification tasks from the UCI Machine Learning Repository (Dua & Karra Taniskidou, 2017); the Iris data set, the PIMA Indians Diabetes data set and the Cleveland Heart Disease data set. The first data set is a well-known classification task of plants based on four flower leaf characteristics with a size of 150 data points and three classes. The second data set is a binary classification task whose task is to correctly diagnose diabetes and contains 769 data points and has nine features. The third data set is aims at classifying the risk of heart disease from no presence (0) to presence (1–4), consisting of 297 instances with 13 features.

To show the model-agnostic nature of our proposed method we applied four distinct classification models to each data set: a random forest, logistic regression, support vector machine (SVM) and a neural network. Table 1 shows for each data set and classifier the F_1 score of the trained model. We validated our approach on four measures; explanation length, accuracy, fidelity and time. These measures for evaluating iML decision rules are adapted from Craven & Shavlik (1999), where the mean length serves as a proxy measure demonstrating the relative explanation comprehensibility (Doshi-Velez & Kim, 2017). The fidelity allows us to state how well the tree explains the underlying model,

Contrastive Explanations with Local Foil Trees

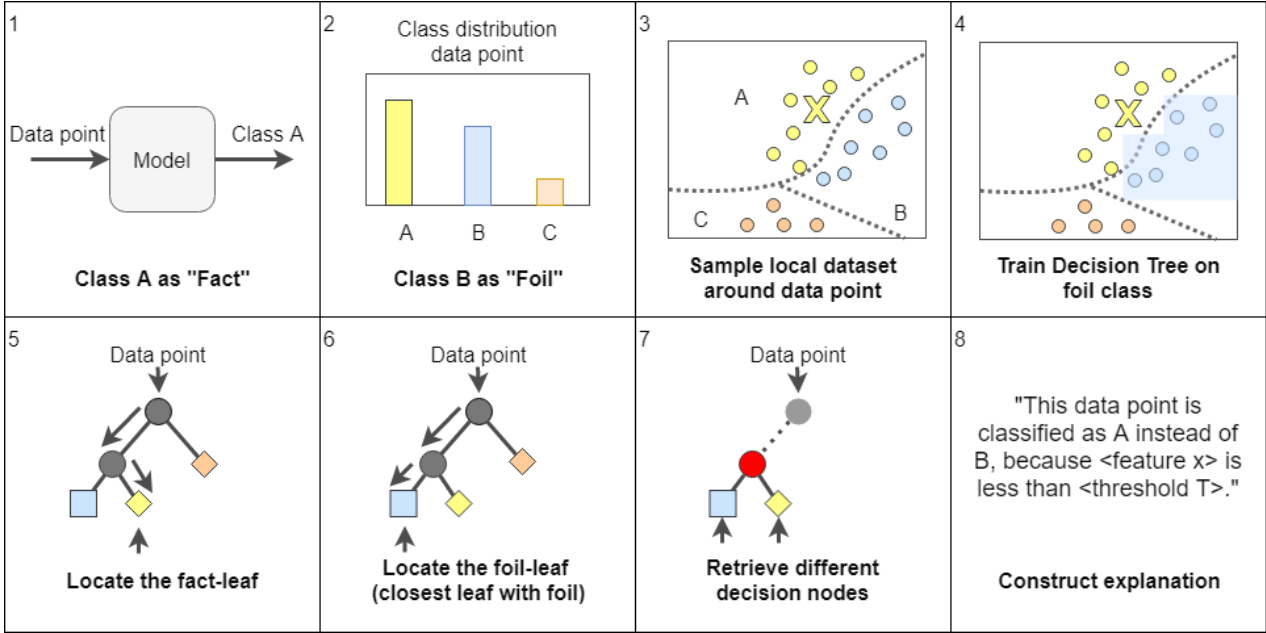


Figure 2. The steps needed to define and train a Foil Tree and to use it to construct a contrastive explanation. Each step corresponds with the listed steps in section 2.

and the accuracy tells us how well its explanations generalize to unseen data points. Below we describe each in detail:

1. **Mean length**; average length of the explanation in terms of decision nodes. The ideal value is in the range $[1.0, \text{Nr. features})$, since a length of 0 means that no explanation is found and a length near the number of features offers little gain compared to showing the entire ordered feature contribution list as in other iML methods.
2. **Accuracy**; F_1 score of the foil-tree for its binary classification task on the test set compared to the true labels. This measure indicates how general the explanations generated from the Foil Tree are on an unseen test set.
3. **Fidelity**; F_1 score of the foil-tree on the test set compared to the model output. This measure provides a quantitative value of how well the Foil Tree agrees with the underlying classification model it tries to explain.
4. **Time**; number of seconds needed on average to explain a test data point.

Each measure is cross-validated three times to account for randomness in foil-tree construction. These results are shown in their respective columns in Table 1. They show that on average the Foil Tree is able to provide concise explanations, with a mean length 1.33, while accurately mimicking the decision boundaries used by the model with a

mean fidelity of 0.93 and generalizes well to unseen data with a mean accuracy of 0.92. The foil-tree performs similar to the underlying ML model in terms of accuracy. Note that for the random forest, logistic regression and SVM models on the diabetes data set rules of length zero were found—i.e. no explanatory differences were found between facts and foils in a number of cases—, resulting in a mean length of less than one. For all other models our method was able to find a difference for every questioned data point.

To further illustrate the proposed method, below we present a single explanation of two classes of the Iris data set in a dialogue setting;

- System: The flowertype is ‘Setosa’.
- User: Why ‘Setosa’ and not ‘Versicolor’?
- System: Because for it to be ‘Versicolor’ the ‘petal width (cm)’ should be smaller and the ‘sepal width (cm)’ should be larger.
- User: How much smaller and larger?
- System: The ‘petal width (cm)’ should be smaller than or equal to 0.8 and the ‘sepal width (cm)’ should be larger than 3.3.

The fact is the ‘Setosa’ class, the foil is the ‘Versicolor’ class and the total length of the explanation contains two decision nodes or literals. The generation of this small dialogue is based on text templates and fixed interactions for the user.

Table 1. Performance of foil-tree explanations on the Iris, PIMA Indians Diabetes and Heart Disease classification tasks. The column ‘Mean length’ also contains the total number of features for that data set as the upper bound of the explanation length.

DATA SET	MODEL	F_1 SCORE	MEAN LENGTH	ACCURACY	FIDELITY	TIME
IRIS	RANDOM FOREST	0.93	1.94 (4)	0.96	0.97	0.014
	LOGISTIC REGRESSION	0.93	1.50 (4)	0.89	0.96	0.007
	SVM	0.93	1.37 (4)	0.89	0.92	0.010
	NEURAL NETWORK	0.97	1.32 (4)	0.87	0.87	0.005
DIABETES	RANDOM FOREST	1.00	0.98 (9)	0.94	0.94	0.041
	LOGISTIC REGRESSION	1.00	0.98 (9)	0.94	0.94	0.032
	SVM	1.00	0.98 (9)	0.94	0.94	0.034
	NEURAL NETWORK	1.00	1.66 (9)	0.99	0.99	0.009
HEART DISEASE	RANDOM FOREST	0.94	1.32 (13)	0.88	0.90	0.106
	LOGISTIC REGRESSION	1.00	1.21 (13)	0.99	0.99	0.006
	SVM	1.00	1.19 (13)	0.86	0.86	0.012
	NEURAL NETWORK	1.00	1.56 (13)	0.92	0.92	0.009

4. Conclusion

Current developments in Interpretable Machine Learning (iML) created new methods to answer “Why output A?” for Machine Learning (ML) models. A large set of such methods use the contributions of each feature used to classify A and then provides either a subset of feature whose contribution is above a threshold, the entire ordered feature list or simply apply it only to low-dimensional data.

This study proposes a novel method to reduce the number of contributing features for a class by answering a contrasting question of the form “Why output A (fact) instead of output B (foil)?” for an arbitrary data point. This allows us to construct an explanation in which only those features play a role that distinguish A from B. Our approach finds the contrastive explanation by taking the complement set of decision rules that cause the classification of A in the rule set of B. In this study we implemented this idea by training a decision tree to distinguish between B and not-B (one-versus-all approach). A fact-leaf is found in which the data point in question resides. Also, a foil-leaf is selected according to a strategy where all data points are classified as the foil (output B). We then form the contrasting rules by extracting the decision nodes in the sub-tree from the lowest common ancestor between the fact-leaf and foil-leaf, that hold for the foil-leaf but not for the fact-leaf. Overlapping rules are merged and eventually used to construct an explanation.

We introduced a simple and naive strategy of finding an appropriate foil-leaf. We also provided an idea to extend this method with more complex and accurate strategies, which is part of our future work. We plan a user validation of our explanations with non-experts in Machine Learning to test the satisfaction of our explanations. In this study we tested if the proposed method is viable on three different benchmark tasks as well as to test its fidelity on different

underlying ML models to show its model-agnostic capacity.

The results showed that for different classifiers our method is able to offer concise explanations that accurately describe the decision boundaries of the model it explains.

As mentioned, our future work will consist out of extending this preliminary method with more foil-leaf search strategies as well as applying the method to more complex tasks and validating its explanations with users. Furthermore, we plan to extend the method with an adaptive foil-leaf search to adapt explanations towards a specific user based on user feedback.

References

- Barocas, S. and Selbst, A. D. Big Data’s Disparate Impact. *Cal. L. Rev.*, 104:671, 2016.
- Caliskan, A., Bryson, J. J., and Narayanan, A. Semantics Derived Automatically from Language Corpora Contain Human-Like Biases. *Science*, 356(6334):183–186, 2017.
- Chakraborty, S., Tomsett, R., Raghavendra, R., Harborne, D., Alzantot, M., Cerutti, F., Srivastava, M., Preece, A., Julier, S., Rao, R. M., Kelley, Troy D., Braines, D., Sensoy, M., Willis, C. J., and Gurrum, P. Interpretability of Deep Learning Models: A Survey of Results. In *IEEE Smart World Congr. DAIS - Work. Distrib. Anal. Infrastruct. Algorithms Multi-Organization Fed.* IEEE, 2017.
- Coglianesi, C. and Lehr, D. Regulating by Robot: Administrative Decision Making in the Machine-Learning Era. *Geo. LJ*, 105:1147, 2016.
- Craven, M. W. and Shavlik, J. W. Rule Extraction: Where Do We Go from Here? Technical report, University of Wisconsin Machine Learning Research Group, 1999.

- Crawford, K. Artificial Intelligence’s White Guy Problem. *The New York Times*, 2016.
- Datta, A., Sen, S., and Zick, Y. Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems. In *Proc. 2016 IEEE Symp. Secur. Priv. (SP 2016)*, pp. 598–617. IEEE, 2016. ISBN 9781509008247. doi: 10.1109/SP.2016.42.
- Dhurandhar, Amit, Chen, Pin-Yu, Luss, Ronny, Tu, Chun-Chen, Ting, Paishun, Shanmugam, Karthikeyan, and Das, Payel. Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives. *arXiv preprint arXiv:1802.07623*, 2018.
- Doshi-Velez, F and Kim, B. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv preprint arXiv:1702.08608*, 2017.
- Dua, D. and Karra Taniskidou, E. UCI Machine Learning Repository, 2017. URL <http://archive.ics.uci.edu/ml>.
- Ehsan, U., Harrison, B., Chan, L., and Riedl, M. O. Rationalization: A Neural Machine Translation Approach to Generating Natural Language Explanations. *arXiv preprint arXiv:1702.07826*, 2017.
- Elenberg, E. R., Dimakis, A. G., Feldman, M., and Karbasi, A. Streaming Weak Submodularity: Interpreting Neural Networks on the Fly. *arXiv preprint arXiv:1703.02647*, 2017.
- Friedler, S. A., Scheidegger, C., Venkatasubramanian, S., Choudhary, S., Hamilton, E. P., and Roth, D. A Comparative Study of Fairness-Enhancing Interventions in Machine Learning. *arXiv preprint arXiv:1802.04422*, 2018.
- Guidotti, R., Monreale, A., Turini, F., Pedreschi, D., and Giannotti, F. A Survey Of Methods For Explaining Black Box Models. *arXiv preprint arXiv:1802.01933*, 2018.
- Hein, D, Udluft, S, and Runkler, T. A. Interpretable Policies for Reinforcement Learning by Genetic Programming. *arXiv preprint arXiv:1712.04170*, 2017.
- Hendricks, L. A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., and Darrell, T. Generating Visual Explanations. In *Eur. Conf. Comput. Vis.*, pp. 3–19, 2016. ISBN 9783319464923. doi: 10.1007/978-3-319-46493-0_1.
- Herman, B. The Promise and Peril of Human Evaluation for Model Interpretability. In *Conf. Neural Inf. Process. Syst.*, 2017.
- Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., and Baesens, B. An Empirical Evaluation of the Comprehensibility of Decision Table, Tree and Rule Based Predictive Models. *Decis. Support Syst.*, 51(1):141–154, 2011. ISSN 01679236. doi: 10.1016/j.dss.2010.12.003.
- Krishnan, R., Sivakumar, G., and Bhattacharya, P. Extracting Decision Trees From Trained Neural Networks. *Pattern Recognit.*, 32:1999–2009, 1999. doi: 10.1145/775047.775113.
- Kulesza, T., Stumpf, S., Wong, W.-K., Burnett, M. M., Perona, S., Ko, A., and Oberst, I. Why-Oriented End-User Debugging of Naive Bayes Text Classification. *ACM Trans. Interact. Intell. Syst. (TiiS)*, 1(1):2, 2011.
- Kulesza, T., Burnett, M., Wong, W.-K., and Stumpf, S. Principles of Explanatory Debugging to Personalize Interactive Machine Learning. In *Proc. 20th Intl. Conf. on Intell. User Interfaces*, pp. 126–137. ACM, 2015.
- Lei, T., Barzilay, R., and Jaakkola, T. Rationalizing Neural Predictions. *arXiv preprint arXiv:1606.04155*, 2016. ISSN 9781450321389. doi: 10.1145/2939672.2939778.
- Lipton, Z. C. The Mythos of Model Interpretability. In *2016 ICML Work. Hum. Interpret. Mach. Learn.*, 2016.
- Lundberg, S. and Lee, S.-I. An Unexpected Unity Among Methods for Interpreting Model Predictions. In *29th Conf. Neural Inf. Process. Syst. (NIPS 2016)*, 2016.
- Malioutov, D. M., Varshney, K. R., Emad, A., and Dash, S. Learning Interpretable Classification Rules with Boolean Compressed Sensing. *Transparent Data Min. Big Small Data. Stud. Big Data*, 32, 2017. doi: 10.1007/978-3-319-54024-5.
- Miller, T., Howe, P., and Sonenberg, L. Explainable AI: Beware of Inmates Running the Asylum. In *Proc. Int. Jt. Conf. Artif. Intell. (IJCAI)*, pp. 36–41, 2017.
- Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K. R. Explaining Nonlinear Classification Decisions with Deep Taylor Decomposition. *Pattern Recognit.*, 65(C):211–222, 2017. ISSN 00313203. doi: 10.1016/j.patcog.2016.11.008.
- Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., and Clune, J. Synthesizing the Preferred Inputs for Neurons in Neural Networks via Deep Generator Networks. *Adv. Neural Inf. Process. Syst.*, 29, 2016.
- Pacer, M. and Lombrozo, T. Ockham’s Razor Cuts to the Root: Simplicity in Causal Explanation. *J. Exp. Psychol. Gen.*, 146(12):1761–1780, 2017. ISSN 1556-5068. doi: 10.1037/xge0000318.

- Puri, N., Gupta, P., Agarwal, P., Verma, S., and Krishnamurthy, B. MAGIX: Model Agnostic Globally Interpretable Explanations. *arXiv preprint arXiv:1702.07160*, 2017.
- Ribeiro, M. T., Singh, S., and Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD'16)*, pp. 1135–1144, 2016. ISBN 9781450321389. doi: 10.1145/2939672.2939778.
- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. In *NIPS 2016 Work. Interpret. Mach. Learn. Complex Syst.*, 2016. ISBN 9781538610329. doi: 10.1109/ICCV.2017.74.
- Sundararajan, M., Taly, A., and Yan, Q. Axiomatic Attribution for Deep Networks. In *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017.
- Thiagarajan, J. J., Kailkhura, B., Sattigeri, P., and Ramamurthy, K. N. TreeView: Peeking into Deep Neural Networks Via Feature-Space Partitioning. In *NIPS 2016 Work. Interpret. Mach. Learn. Complex Syst.*, 2016.
- Wang, T., Rudin, C., Velez-Doshi, F., Liu, Y., Klampfl, E., and Macneille, P. Bayesian Rule Sets for Interpretable Classification. In *Proc. IEEE Int. Conf. Data Min. (ICDM)*, pp. 1269–1274. IEEE, 2017. ISBN 9781509054725. doi: 10.1109/ICDM.2016.130.
- Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S. Top-Down Neural Attention by Excitation Backprop. *Int. J. Comput. Vis.*, pp. 1–19, 2017. ISSN 15731405. doi: 10.1007/s11263-017-1059-x.
- Zhou, Y. and Hooker, G. Interpreting Models via Single Tree Approximation. *arXiv preprint arXiv:1610.09036*, 2016.