Journal of Artificial Intelligence Research 70 (2021) 245-317 Submitted 06/2020; published 01/2021

A Survey on the Explainability of Supervised Machine Learning

Nadia Burkart NADIA.BURKART@IOSB.FRAUNHOFER.DE
Fraunhofer Center of Machine Learning,
Fraunhofer Institute for Optronics, System Technologies, and Image Exploitation IOSB,
Interactive Analysis and Diagnosis
Fraunhoferstrasse 1, 76131 Karlsruhe, Germany

Marco F. Huber MARCO.HUBER@IEEE.ORG
University of Stuttgart, Institute of Industrial Manufacturing and Management IFF,
Fraunhofer Institute for Manufacturing Engineering and Automation IPA,
Center for Cyber Cognitive Intelligence (CCI),
Nobelstrasse 12, 70569 Stuttgart, Germany

Abstract

Predictions obtained by, *e.g.*, artificial neural networks have a high accuracy but humans often perceive the models as black boxes. Insights about the decision making are mostly opaque for humans. Particularly understanding the decision making in highly sensitive areas such as healthcare or finance, is of paramount importance. The decision-making behind the black boxes requires it to be more transparent, accountable, and understandable for humans. This survey paper provides essential definitions, an overview of the different principles and methodologies of explainable Supervised Machine Learning (SML). We conduct a state-of-the-art survey that reviews past and recent explainable SML approaches and classifies them according to the introduced definitions. Finally, we illustrate principles by means of an explanatory case study and discuss important future directions.

1. Introduction

The accuracy of current Artificial Intelligence (AI) models is remarkable but accuracy is not the only aspect that is of utmost importance. For sensitive domains, a detailed understanding of the model and the outputs is important as well. The underlying machine learning and deep learning algorithms construct complex models that are opaque for humans. Holzinger et al. (2019b) state that the medical domain is among the greatest challenges for AI. For areas such as health care, where a deep understanding of the AI application is crucial, the need for Explainable Artificial Intelligence (XAI) is obvious.

Explainability is important in many domains but not in all domains. We already mentioned areas in which explainability is important such as health care. In other domains such as *aircraft collision avoidance*, algorithms have been operating without human interaction without giving explanations for years. Explainability is required when there is some degree of *incompleteness*. Incompleteness, to be sure, is not to be confused with uncertainty. Un-

©2021 AI Access Foundation. All rights reserved.

certainty refers to something that can be formalized and handled by mathematical models. Incompleteness, on the other hand, means that there is something about the problem that cannot be sufficiently encoded into the model (Doshi-Velez & Kim, 2017). For instance, a criminal risk assessment tool should be unbiased and it also should conform to human notions of fairness and ethics. But ethics is a broad field that is subjective and hard to formalize. In contrast, airplane collision avoidance is a problem that is well understood and that can be described precisely. If a system avoids collisions sufficiently well, there are no further concerns about it. No explanation is required.

Incompleteness can stem from various sources. Another example are safety reasons. For a system that cannot be tested in a full deployment environment, there is a certain incompleteness with regards to whether the defined test environment is actually a suitable model for the real world. The human desire for scientific understanding also adds incompleteness to the task. The models can only learn to optimize their objective by means of correlation. Yet, humans strive to discover causal dependencies. All of the examples given contribute to a lack of understanding but sometimes, as these examples were supposed to illustrate as well, this may not bother us.

Automated decision-making can determine whether someone qualifies for certain insurance policies, it can determine which advertisement one sees, which job interviews one is being invited to, which university position one is being offered or what kind of medical treatment one will receive. If a loan is approved and everything accords with one's expectations, probably no one will ever ask for a detailed explanation. However, in case of a rejection, the reasons would be quiet interesting and helpful.

In most cases, the models are complicated because the problem is complex and it is almost impossible to explain what exactly the models are doing and why they are doing it. Yoshua Bengio (Bengio & Pearson, 2016), a pioneer in the field of deep learning research, said "As soon as you have a complicated enough machine, it becomes almost impossible to completely explain what it does." Jason Yosinski from Uber states (Voosen, 2017) "We build amazing models. But we don't quite understand them. Every year this gap is going to get a little bit larger." Voosen (2017) gets to the heart of the issue and questions "why, model, why?."

According to Lipton (2018), explainability is demanded whenever the goal the prediction model was constructed for differs from the actual usage when the model is being deployed. In other words, the need for explainability arises due to the mismatch between what a model can explain and what a decision maker wants to know. Explainability issues also arises for well-functioning models that fail on few data instances. Here, we also demand explanations for why the model did not perform well on these few feature combinations (Kabra et al., 2015). According to Martens et al. (2009), explainability is essential whenever a model needs to be validated before it can be implemented and deployed. Domains that demand explainability are characterized by making critical decisions that involve, for example, a human life, *e.g.*, in health care.

The renewed EU General Data Protection Regulation (GDPR) could require AI providers to provide users with explanations of the results of automated decision-making based on their personal data. Personal data is defined as information relating to an identified or identifiable natural person (Europa.eu, 2017). The GDPR replaced the Data Protection Directive from 1995. This new requirement affects large parts of the industry. The Euro-

pean Parliament has revised regulations that concern the collection, storage, and usage of personal information. The GDPR may make complicated or even lead to the prohibition of the use of opaque models that are used for certain applications, e.g., for recommender systems that work based on personal data. Goodman and Flaxman (2016) call this the right of explanation for each subject (person). This will most likely affect financial institutions, social networks and the health care industry. Automated decision-making used by financial institutions for monitoring credit risk or money laundering needs to be transparent, interpretable and accountable.

In January 2017, the Association for Computing Machinery (ACM, 2017) released a statement on algorithmic transparency and accountability. In this statement, the ACM points out that the usage of algorithms for automated decision-making can result in harmful discrimination. To avoid those problems, the ACM also published a list of certain principles to follow. In May 2017, the DARPA launched the program XAI (Gunning, 2017) which aims at providing explainable and highly accurate models. XAI is an umbrella term for any research trying to solve the black-box problem for AI. Since there are a lot of different approaches for solving this problem, each with their own individual needs and goals, there is no single common definition of the term XAI. The key idea, however, is to enable users to understand the decision-making of any model.

The words understanding, interpreting and explaining are often used interchangeably when used in the context of explainable AI (Doran et al., 2017). Usually, *interpretability* is used in terms of comprehending how the prediction model works as a whole. *Explainability*, in contrast, is often used when explanations are given by prediction models that are incomprehensible themselves.

Explainability and interpretability are also important aspects for deep learning models, where a decision depends on an enormous amount of weights and parameters. Here, the parameters are often abstract and disconnected from the real world, which makes it difficult to interpret and explain the results of deep models (Angelov & Soares, 2019). Samek et al. (2017) describe different methods for visualizing and explaining deep learning models like the Sensitivity Analysis (SA) or the Layer-wise relevance propagation (LRP). Biswas et al. (2017) or Zilke et al. (2016) also mention different techniques for decompositional decision rules from ANNs. Since the focus of this paper lies on explanations for SML mainly for tabular data, only a few explanation methods for deep learning models are mentioned.

As the research field of XAI grows rapidly, there already exist a few survey papers that gather existing approaches. Guidotti et al. (2018b) provide a survey on approaches for explaining black box models. The authors provide a taxonomy for the approaches that is not directly linked to one certain learning problem. Molnar (2018) describes several approaches for generating explanations and interpretable models. He introduces different data sets and thereby describes some approaches in the main field of interpretable models, model agnostic methods and example-based explanations. Samek et al. (2019) introduce the topic of explainable AI and describe corresponding approaches. Further, they provide directions of future development. Adadi and Berrada (2018) provide an overview of general XAI research contributions by addressing different perspectives in a non-technical overview of the key aspects in XAI and various approaches that are related to XAI. Biran and Cotton (2017) provide an overview of explanations and justifications for different machine learning models like Bayesian networks, recommender systems and other adjacent areas. Explaining

a machine learning model means to render the output of a model understandable to a human being. Justification can be generated for a black box model and describes why the generated decision is a meaningful one (Biran & Cotton, 2017). Montavon et al. (2018) offers various techniques for interpreting individual outputs from deep neural networks, focusing on the conceptual aspects that make these interpretations useful in practice. Gilpin et al. (2018) defines the terms *interpretability* and explainability and reveals their differences. A new taxonomy which offers different explanation possibilities for machine learning models is introduced. This taxonomy can be used to explain the treatment of features by a network, the representation of features within a network or the architecture of a network. Unlike many other papers on explainable machine learning, the paper by Dosilović et al. (2018) focuses on explanations and interpretations of supervised machine learning methods. The paper describes the *integrated (transparency-based)* and the *post hoc* methods and offers a discussion about the topic of explainable machine learning. Tjoa and Guan (2019) give an broad overview of interpretation approaches and classify them. The authors focus on machine interpretation in the medical field and disclose the complexity of interpreting the decision of a black box model.

This survey paper provides a detailed introduction to the topic of explainable SML regarding the definitions and a foundation for classifying the various approaches in the field. We distinguish between the various problem definitions to categorize the field of explainable supervised learning into interpretable models, surrogate model fitting and explanation generation. The definition of interpretable models focuses on the entire model understanding that is achieved either naturally or by using design principles to force it. The surrogate model fitting approach approximates local or global interpretable models based on a black box. The explanation generation process that directly produces a kind of explanation distinguishes between local and global explainability.

In summary, the paper offers the following contributions:

- formalization of five different explanation approaches and a review of the corresponding literature (classification and regression) for the entire explanation chain
- reasons for explainability, review of important domains and the assessment of explainability
- a chapter that solely highlights various aspects around the topic of data and explainability such as data quality and ontologies
- a continuous use case that supports the understanding of the different explanation approaches
- a review of important future directions and a discussion.

2. Reasons for Explainability and Demanding Domains

In this chapter, we will describe reasons for explainability and introduce example domains where XAI is needed.

2.1 Reasons for Explainability

Automated decision-making systems are not widely accepted. Humans want to understand a decision or at least they want to get an explanation for certain decisions. This is due to the fact that humans do not trust blindly. Trust, then, is one of the motivating aspects of explainability. Other motivating aspects are causality, transferability, informativeness, fair and ethical decision-making (Lipton, 2018), accountability, making adjustments and proxy functionality.

Trust: Trust and acceptance of the prediction model are needed for the prediction model's deployment. Understanding and knowing the prediction model's strengths and weaknesses is a prerequisite for human trust and, hence, for model deployment.

Causality: Explainability, e.g. in the form of attribute importance, conveys a sense of causality to the system's target group. This concept of causality can only be grasped when the system points out the underlying input-output relationship.

Transferability: The prediction model needs to convey an understanding of future behavior for a human decision-maker in order to use the prediction model with unseen data. Only when the decision-maker knows that the model generalizes well or when he knows in which context it generalizes well, the prediction model will be put in charge of making decisions.

Informativeness: In order to be deployed as a system, it is necessary to know whether the system actually serves the real world purposes it is designed for instead of merely serving the purposes it was trained for. If this information is given, the system can be deployed.

Fair and Ethical Decision Making: Knowing the reasons for a certain decision is a societal need and most likely it will be an official right for EU-citizens (Goodman & Flaxman, 2016). This right to explanation requires decision-makers to present their results in a comprehensible way in order to perceive conformity to ethical standards. Each person that is affected by an automated decision can make use of this right to explanation.

Accountability: One goal of incorporating explainability into the decision-making process is to make an algorithm accountable for its actions. In order for a system to be accountable, it has to be able to explain and justify its decisions. Furthermore, the data-shift problem can be targeted with interpretable systems, making these more accountable for their actions (Freitas, 2014).

Making Adjustments: Understanding the prediction model and the underlying factors enables domain experts to compare the prediction model to the existing domain knowledge. Explainability is a prerequisite for the ability to adjust the prediction model by incorporating domain knowledge. According to Selvaraju et al. (2016), explainability of prediction models can teach humans, especially domain experts using these prediction models, how to make better decisions. Furthermore, when looked at from an algorithmic point of view, explainability enables system designers to make changes to the prediction model by, e.g., adjusting parameters. Explainability is also useful for developers since it can be used to identify failure modes.

Proxy Functionality: When explainability is provided by a system, it can also be examined based on other criteria that cannot be easily quantified such as safety, non-discrimination, privacy, robustness, reliability, usability, fairness, verification and causality (Doshi-Velez & Kim, 2017). In this case explainability serves as a proxy.

2.2 Domains Demanding Explainability

As already mentioned at the outset, explainability is not required for every domain. There are domains that use black box prediction models since these domains are either well studied and users trust the existing models or because no direct consequences threaten in case the system makes mistakes (Doshi-Velez & Kim, 2017) , *e.g.*, in recommendation systems for marketing . According to Lipton (2018), explainability is demanded whenever the goal the prediction model was constructed for differs from the actual usage the model is being deployed for. The need for explainability, then, arises due to a mismatch between what a model can explain and what a decision-maker wants to know. According to Martens et al. (2009), explainability is important whenever a model needs to be validated before it can be implemented and deployed. Domains that demand explainability are characterized by making critical decisions that involve, e.g., human lives (medicine/healthcare) or a lot of money (banking/finance) (Strumbelj et al., 2010). In what follows, we take a closer look at relevant domains and provide a motivating example for the necessity of explainability in these specific domains.

Medical Domain/Health-Care: When a medical researcher uses an intelligible system for screening patients with a high risk for cancer, it is not sufficient to identify patients with a high risk in an accurate manner; but he also understand causes of cancer (Henelius et al., 2014).

Judicial System: In order to defend an automated decision in court, it is necessary to understand the reasons for a specific prediction (Freitas, 2014).

Banking/Financial Domain: It is a legal obligation to be able to explain why a customer was denied a credit (Freitas, 2014). Furthermore, it is of great interest for banks and insurance companies to predict and understand customer churn to be able to develop a reasoned counteracting plan (due to high costs of seeking new customers) (Verbeke et al., 2011).

Bio-informatics: If trust can be established in a system, more time and money will be invested in experiments regarding the system's domain according to Freitas (2014) and Subianto and Siebes (2007).

Automobile Industry: If there is an autonomously driving car involved in an accident, it is of great interest to the developer, to the people involved and to the legal system to understand the reasons why the accident happened in order to fix the system and to sue the person responsible for the accident.

Marketing: Marketing is mainly concerned about distributing the products of a company. A company is in a better position than another company if it can explain why a customer preferred one product over another one since this information can be used, e.g., to equip other products with purchase-relevant attributes.

Election campaigns: Similar to customer churn, votes in an election can be influenced if the reasons for voting decisions are better understood. In an election campaign, voters can be targeted with coordinated advertising based on their personal interests.

Precision Agriculture: Due to the use of remote sensors, satellites, and UAVs, information regarding a particular area can be gathered. Through the gathered data and machine learning models, farmers can develop a better understanding of what they need to do in order to increase the harvest benefits (Byrum, 2017).

Expert Systems for the Military: The military can make use of expert systems, e.g., in the context of training soldiers. In a military simulation environment, a user has to accomplish a certain goal. With the help of explainable machine learning, the user receives meaningful information on how to accomplish the goal more efficiently (Lent et al., 2004).

Recommender systems: Explainable recommendations help system designers to understand why a recommender system offers a particular product to a particular user group. It helps to improve the effectiveness of a recommender system and the clarity of a decision (Zhang & Chen, 2018).

3. Concepts of Explainability

A variety of different approaches for explaining learned decisions have been proposed (Molnar, 2018). Some try to explain the model as a whole or completely replace it with an inherently understandable model such as a decision tree (Freitas, 2014). Other approaches try to steer the model in the learning process to a more explainable state (Schaaf & Huber, 2019; Burkart et al., 2019) or focus on just explaining single predictions for example by highlighting important features (Ribeiro et al., 2016b) or contrasting it to another decision (Wachter et al., 2018). In the following sections, we structure the area of explainable supervised machine learning. First, we describe the problem definitions and dimensions. Next, we introduce interpretable model types and techniques. Finally, the explanation itself is described.

3.1 Problem Definition and Dimensions

In this subsection, we will give an overview of the problem definitions and the according dimensions in SML. This creates the basis for categorizing the procedures.

SML aims at learning a so-called *model* $h(\mathbf{x}) = y$ that maps a feature vector $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ to a target $y \in \mathcal{Y} \subseteq \mathbb{R}$. For this purpose, a set of training data $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ is used for the learning process of the model. SML can be divided into the tasks of *classification* and *regression*. For classification, the target y is a discrete value often called a *label*. For instance, if $y \in \{0, 1\}$ or $y \in \{-1, 1\}$ one speaks about binary classification. The task of regression is to predict a continuous target value $y \in \mathbb{R}$.

In this paper, we differentiate between two different types of models. A model can either be a black box $b : \mathcal{X} \to \mathcal{Y}$, $b(\mathbf{x}) = y$ with $b \in \mathcal{B}$, where $\mathcal{B} \subset \mathcal{H}$ is the hypothesis space of black box models, e.g., $\mathcal{B} = \{\text{neural networks with one hidden layer}\}$. Or the model can be an interpretable one $w : \mathcal{X} \to \mathcal{Y}$, $w(\mathbf{x}) = y$ with $w \in \mathcal{I}$, where $\mathcal{I} \subset \mathcal{H}$ is the hypothesis space of interpretable models, e.g., $\mathcal{I} = \{\text{decision trees of depth } 3\}$.

To evaluate the prediction performance of the trained model, we use the *error measure* $S : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. A common example of an error measure in binary classification with $y \in \{-1, 1\}$ is the *hinge loss* $S(h(\mathbf{x}), y) = \max\{0, 1 - h(\mathbf{x}) \cdot y\}$, which is zero when the true label y and the prediction $h(\mathbf{x})$ are identical. In regression problems, the *squared deviation* $S(h(\mathbf{x}), y) = (h(\mathbf{x}) - y)^2$ is a common error measure. Given an error measure, SML can be formulated as an optimization problem.

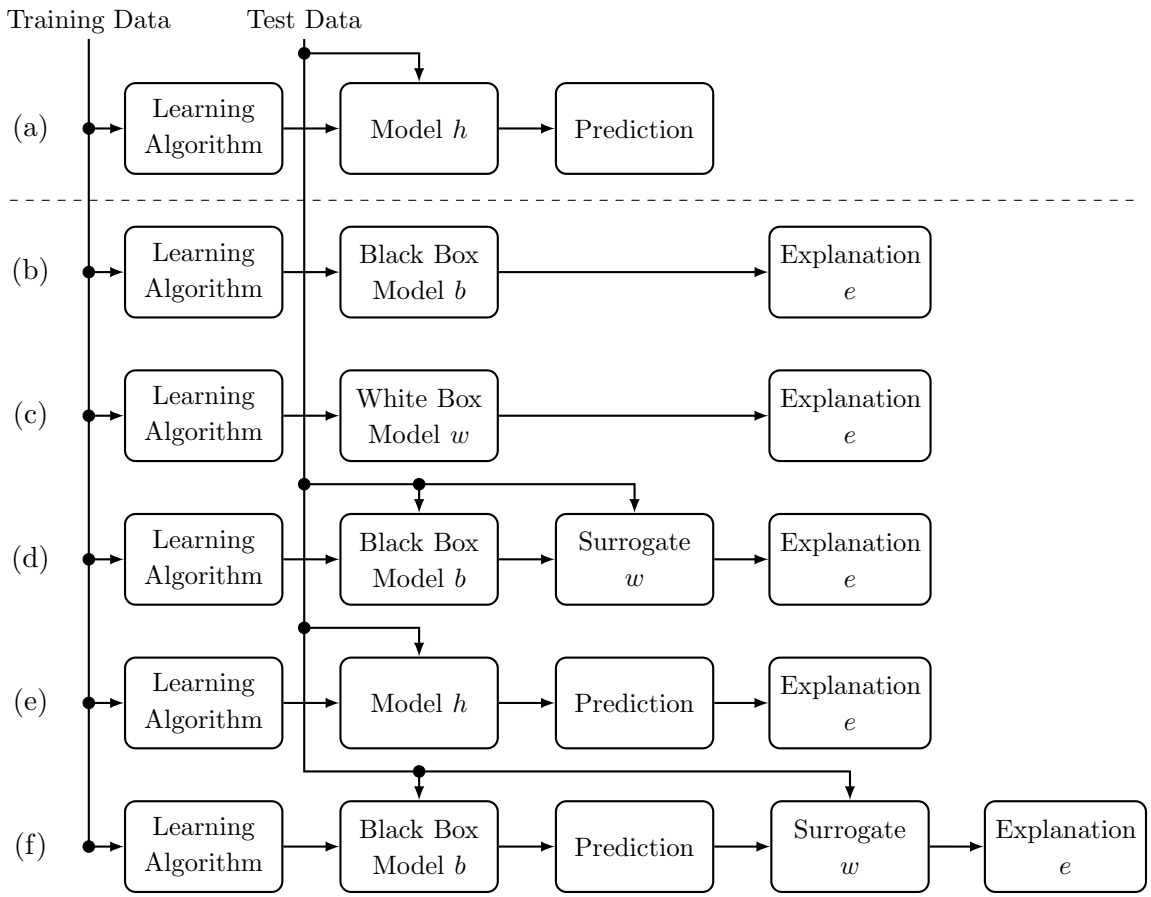


Figure 1: From an opaque supervised model to an explanation. (a) Standard supervised machine learning without explanation. (b)-(d) Model/global explanations: (b) post-hoc black box model explanation, (c) interpretable by nature, i.e., white box model explanation, and (d) explaining a black box model by means of a global surrogate model. (e)-(f) Instance/local explanations: (e) directly or (f) with a local surrogate.

Problem 1 (Supervised Machine Learning) *Given training data $\mathcal{D}$, the SML aims for solving the optimization problem*

$$h^* = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} S\big(h(\mathbf{x}_i), y_i\big) \ , \tag{1}$$

where the averaged error over all training instances is minimized and where h^* is the resulting model.

In case of a parametric model $h(\mathbf{x}; \theta)$ with parameter vector θ as in a neural network, (1) is equivalently formulated as

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} S\big(h(\mathbf{x}_i; \theta), y_i\big) \ . \tag{2}$$

In many cases, the optimization problem (1) or (2) cannot be solved analytically exactly. One of the rare exceptions where this is possible is linear regression. Thus, it is common to end up with a sub-optimal solution that is found numerically, as is the case when deploying deep neural networks where the parameter vector θ is determined by means of gradient descent.

It is important to differentiate between the *learning algorithm* and the SML model: The learning algorithm tries to solve the optimization problem (1), either directly or implicitly. The result of the optimization problem and, thus, the output of the learning algorithm, is the actual model, which then can be applied to an unseen data instance $\mathbf{x}$ in order to obtain a prediction $y = h^*(\mathbf{x})$. Figure 1(a) illustrates this SML learning pipeline.

3.2 Five Ways to Gain Interpretability

Solving Problem 1 usually leads to black box models requiring further processing in order to obtain the desired explanations. In what follows, we define five interpretable SML problems by modifying or extending (1). In general, these problems can be grouped into *model explanation* approaches (cf. Figure 1(b)–(d)) and *instance explanation* approaches (cf. Figure 1(e)–(f)). Model explanation approaches generate the explanations on the basis of an SML model trained on $\mathcal{D}$ and aim at providing insights about the inner workings of the entire model. In contrast, instance explanation approaches merely try to explain a model prediction y for a single data instance $\mathbf{x}$. Thus, generated explanations are only valid for $\mathbf{x}$ and its close vicinity. Model and instance explanation approaches are also referred to as *global* and *local* explanation approaches, respectively.

In what follows, the different ways to gain interpretability are defined formally. In addition, we will briefly mention some well-known examples.

3.2.1 Explanation Generation

We first need to define the explanation generation process itself. While learning algorithms provide the SML model and the model, in turn, provides predictions given unseen data instances, the so-called *explanator* is concerned with deriving a human-comprehensible explanation for the model and/or the prediction. Thus, in any case some explanator is mandatory to gain understanding of the model used or the prediction obtained. As indicated in

Figure 1(c), even interpretable models[1] need an additional component that extracts an explanation from the model. Here, an explanator provides additional means for the model's comprehensibility such as feature statistics, feature importance or visualizations. In turn, explanators are no predictors. They always rely on a learned model.

Problem 2 (Explanation Generation) *An explanator function e is defined as*

$$e : (\mathcal{X} \to \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathcal{E} \tag{3}$$

which takes an SML model (black box or interpretable) and a specific data set as input and provides an explanation belonging to the set $\mathcal{E}$ of all possible explanations as an output. There are two possible explanation generation problems:

Global *Extracting a global explanation from a model that is representative for some specific data set $\mathcal{D}'$, i.e., $e(b, \mathcal{D}')$ in case of a black box model (see Figure 1(b)) or $e(w, \mathcal{D}')$ for interpretable models (see Figure 1(c) and (d)).*

Local *Instance explanators extract an explanation for a single test input $\mathbf{x}$ and the corresponding prediction y, i.e., $e(b, (\mathbf{x}, y))$ or $e(w, (\mathbf{x}, y))$.*

Given this definition, we can introduce the first type of explanation of an SML model, namely the direct interpretation of a given black box model in a *post-hoc* fashion as depicted in Figure 1(b). This is achieved by means of global explanators, often being model-agnostic. A well-known example are partial dependency plots (cf. Section 6.1.1 and Goldstein et al. (2015)).

Global explanators in general require no predictions and solely rely on the learned model (black box or interpretable) and some set of feature vectors—often the training data itself. Thus, they are sometimes also referred to as *static* explanators. In some cases, not even the data set $\mathcal{D}'$ is required to generate an explanation from a learned model, i.e., it follows $e(b, \varnothing)$ and $e(w, \varnothing)$. A corresponding example are the attribute weights extracted from linear regression models.

Local explanators allow for the second way of generating explanations as depicted in Figure 1(e). Given a prediction of a model (black box or interpretable), the local explanator provides insights that are only valid for the particular instance and cannot be generalized for the entire model. For this reason, they are often also named *dynamic* or *output* explanation generation. Examples are counterfactual explanations (see Section 6.2.2), Shapley values (cf. Section 6.2.1 and Strumbelj & Kononenko, 2014), or decision paths when classifying $\mathbf{x}$ with a decision tree.

3.2.2 Learning Interpretable Models

Next, we consider models that are interpretable on their own, i.e., models that are easily comprehensible for humans.

Problem 3 (White Box Model Learning) *Modifying Problem 1 leads to the optimization problem*

$$w^* = \arg \min_{w \in \mathcal{I}} \frac{1}{n} \sum_{i=1}^{n} S\big(w(\mathbf{x}_i), y_i\big) \ , \tag{4}$$

1. The terms "white box model" and "interpretable model" are used synonymously throughout this paper.

with $(\mathbf{x}_i, y_i) \in \mathcal{D}$.

That is, by solving Problem 3, one aims for learning a white box model from the hypothesis space of interpretable models $\mathcal{I}$, which is the third way of gaining explainability. This corresponds to the pipeline depicted in Figure 1(c) and is often also called *ante-hoc* interpretability. Typical examples are learning small decision trees or using linear models for regression problems.

3.2.3 Surrogate Model Learning

While using interpretable models might be appropriate for some learning problems, they come at the cost of flexibility, accuracy, and usability according to Ribeiro et al. (2016a). Here, so-called *post-hoc* interpretability can be of help, where the black box model is still used for predictions and, thus, one can rely on the potentially high accuracy of this model. In addition, a white box surrogate model of the black box model is generated to gain interpretability. In what follows, the remaining two ways for obtaining a surrogate model are introduced.

Problem 4 (Surrogate Model Fitting) *Surrogate Model Fitting is the process of translating a black box model into an approximate interpretable model by solving*

$$w^* = \arg\min_{w \in \mathcal{I}} \frac{1}{|\mathcal{X}|} \sum\nolimits_{\mathbf{x} \in \mathcal{X}} S\big(w(\mathbf{x}), b(\mathbf{x})\big) \ . \tag{5}$$

Here, we differentiate between two different kinds of surrogate models.

Global *The surrogate model w approximates the black box model b on the whole training data, i.e., $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is taken from the training data set $\mathcal{D}$. Alternatively, $\mathcal{X}$ is a sample data set which represents the input data distribution of the model b sufficiently well.*

Local *The surrogate model w approximates the black box model around a test input $\mathbf{x}$ defined as $\mathcal{X} = \{\mathbf{x}' | \mathbf{x}' \in N(\mathbf{x})\}$, where N is some neighborhood of $\mathbf{x}$.*

It is worth mentioning that in (5), the measure S acts as a *fidelity* score, i.e., it quantifies how well the surrogate w approximates or agrees with the black box model b.

Global and local surrogates are reflected in Figure 1(d) and (f), respectively. The global surrogate tries to simulate all predictions of the black box model with high accuracy, which makes it possible to understand the black box. A simple example of a global surrogate model is to learn a decision tree on a data set $\mathcal{D}' = \{(\mathbf{x}_1, b(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, b(\mathbf{x}_n))\}$, i.e., the surrogate is trained on the predictions $b(\mathbf{x}_i)$ of the black box model, where $\mathbf{x}_i$ are the feature vectors from the training data.

A local surrogate is only valid near the current prediction. Therefore, only a local understanding of the black box model can be gained. Well-known representatives of the class of local surrogates are LIME (Ribeiro et al., 2016c) or SHAP (Lundberg & Lee, 2017).

3.2.4 Summary

Table 1 summarizes various dimensions regarding the problem definitions in the field of explainable SML:

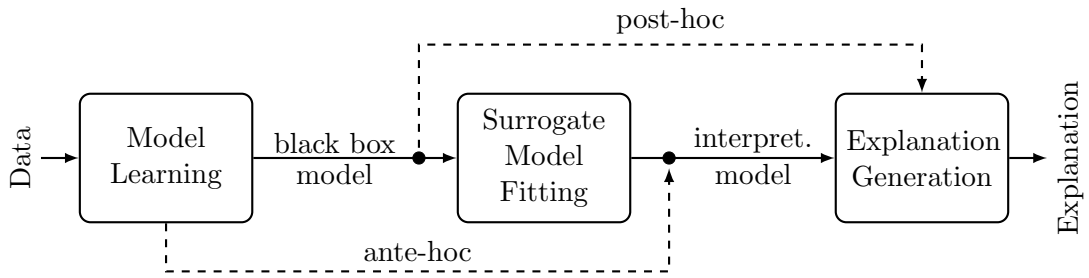


Figure 2: Summary of model/global explanation approaches. Strictly following the solid arrows corresponds to global surrogate fitting as depicted in Figure 1(d). The ante-hoc path corresponds to learning an interpretable/white box model (cf. Figure 1(c)), while the post-hoc path aims for black box model explanation (cf. Figure 1(b)).

ante-hoc Interpretability is built-in from the beginning of the model creation

post-hoc Interpretability is created after model creation

instance/local Interpretability only holds locally for a single data instance and its close vicinity

model/global Interpretability holds globally for the entire model

specific The mechanism for gaining interpretability works only for a specific model class

agnostic The mechanism for gaining interpretability is generally applicable for many or even all model classes

data independent The mechanism for gaining interpretability works without additional data

data dependent The mechanism for gaining interpretability requires data

Interpretable/white box models are inherently ante-hoc and specific. There is no agnostic interpretable model learning approach. One aim behind using an interpretable model is to have build-in (by nature) model explainability. Local interpretability is merely possible when combined with a local explanation approach. The data independence/dependence dimensions are not appropriate for white box models since they alone cannot provide an explanation.

Surrogate models are created in a post-hoc fashion from a given black box model and describe a certain instance or the entire model (see Problem 4). The surrogate model can be applied either to a specific model class or it can be agnostic for many classes of models. In order to create a surrogate model, it is inevitable to rely on data, at least to measure and optimize the fidelity between surrogate and black box model or to create a local surrogate for a single data instance.

We can see that interpretable model learning and surrogate model fitting are mutually exclusive when it comes to ante-hoc or post-hoc explainability. This is emphasized in Figure 2.

Table 1: Dimensions of explainable supervised learning. ✓ indicates fulfillment of the dimension, while ✗ states that the dimension is not applicable. Further, – indicates that the dimension is not appropriate.

Dimensions	**Interpretable Models**	**Surrogate Models**	**Explanation Generation**
ante-hoc	✓	✗	✗
post-hoc	✗	✓	✓
instance/local	✗	✓	✓
model/global	✓	✓	✓
specific	✓	✓	✓
agnostic	✗	✓	✓
data independent	–	✗	✓
data dependent	–	✓	✓

As is the case with fitting surrogate models, the generation of an explanation is strictly post-hoc as well. Besides that, for both it is possible to find approaches that are used in instance or model explanations that are model specific or model agnostic and that are data independent—and thus global—or data dependent.

3.3 Interpretable Model Types

According to Section 3.2, some models belong to the hypothesis space $\mathcal{I}$. These interpretable models provide interpretability by themselves and, thus, can be considered "interpretable by nature". The models are understood in their entirety by the model's target group (see Section 3.5.3). The following incomplete list names some interpretable models:

Linear models: Linear models consist of input features and weights for each of them. The models are built by, *e.g.*, logistic regression in such a way as to assign a weight to every feature used by the model. The assigned weight indicates the feature's contribution to the prediction. Linear models are also easily adopted to yield a continuous value instead of a discrete class label; hence, they are useful for regression. A special group of linear applications are scoring systems. Scoring systems assign each feature or feature interval a specific weight. A final score is evaluated in the same way as the prediction of a linear model is. However, an instance is classified after comparing the final score with a defined threshold.

Decision trees: Decision trees are trained on the labeled input features. The constructed trees consist of tree nodes and leaf nodes. For an exemplary decision tree see Figure 4 and 5. Tree nodes are assigned a splitting feature and a splitting value. Leaf nodes are assigned a class label in case of classification and the averaged value in case of regression. The process for a test instance $\mathbf{x}$ starts at the top tree node—the root node—and traverses downwards until it reaches a lead node. At every intermediate node, a particular feature value is compared to the splitting value. Depending on the outcome of this comparison, traversing continues with the left or the right path. Decision trees are usually constructed in a top-down and greedy manner such that once a feature and a feature's value are selected as the splitting criterion, they cannot be switched by another splitting value or splitting feature (Letham et al., 2012). Every decision tree can be transformed into a rule-based model but not vice versa.

Rule-based models: Decision rules have the structure of IF *condition* THEN *label* ELSE *other label*. There are different specific rule-based approaches such as simple decision rules, decision sets, decision tables and m-of-n rules. To add a rule to an already existing one, one can either build a list by adding a rule with *else if* or create a set by adding another rule without any additional syntax. A condition in a rule consists either of a single feature, operator, value triple—in the literature sometimes called literal (Wang et al., 2015b) or predicate (Lakkaraju et al., 2016, 2017)—or it consists of a conjunction or disjunction of predicates. The most common case is to use conjunctions of predicates as the condition (Huysmans et al., 2011).

Naive Bayes: The Bayesian classification is a statistical classification method that predicts the probability of an object belonging to a particular group. Naive Bayesian classifiers simplify this problem by assuming the validity of the independence assumption. This assumption states that the effect of a characteristic in the classification is independent of the expressions of other feature values. This assumption is naive insofar as it rarely applies to reality. However, it significantly reduces the complexity of the problem.

k-nearest neighbours: Nearest neighbor models do not explicitly build a model from the training data but instead use a similarity measure to compute the nearest neighbors of the test data instance. In case of classification, the label y of the test instance is determined by means of majority voting. The predicted value y for regression problems is obtained via averaging over the neighbor's values.

Interactive models: Interactive learning combines human feedback with the machine learning concepts of active learning and online learning (Holzinger et al., 2019c). Thus, interactive learning is considered a subset of the so-called "human-in-the-loop" algorithms. Learning algorithms are built interactively through end-user input. End-users can review the model output, make corrections in the learning algorithm, and provide feedback about the model output to the learning algorithm. Holzinger et al. (2017) summarized the aim of this by concluding to make use of the human cognitive abilities when machines fail.

Bayesian Networks: Bayesian networks are acyclic graphs in which the variables are represented as nodes and the relationship between the variables as directed edges. These links between variables express a conditional probability. They are designed to model causal relations in the real world. Bayesian networks do not provide a logical but a probabilistic output (Charniak, 1991).

3.4 General-Purpose Techniques for Interpretability by Design

Interpretability can be forced into a model. This means that we can clearly choose between a more interpretable model or a more accurate model. However, this does not necessarily imply a trade-off between interpretability and accuracy. Both goals can be achieved if they are specifically examined during the generation procedure. It is believed that linear models are better interpretable than deep models, where most popular notions of interpretability depend on the comprehensibility of the features (Lipton et al., 2016). However, if we compare linear models to deep models, deep neural networks tend to operate on raw or lightly processed features while linear models often have to operate on heavily hand-engineered features in order to reach a similar performance level(Lipton, 2018). Those raw features used by deep models are intuitively meaningful, whereas the hand-engineered features used

in linear models are not easy decomposable (Lipton, 2018). Moreover, linear models which rely on the application of specific features tend to be less likely to also be useful when deployed for other tasks. Conversely, deep neural networks seem to generalize better on different tasks (Lipton et al., 2016).

Rudin (2018) claims that this is often not true especially when considering structured data with meaningful features. Interpretability can be enhanced into linear models or Bayesian networks via constraints by demanding, *e.g.*, sparsity or monotonicity; it can be incorporated into rule- and tree-based prediction models by restricting the number of rules or by limiting the tree size. Other approaches use a specific penalizing procedure that make use of a combination of the methods just described.

Sparsity: Sparsity needs to be introduced into the model due to the limited cognitive capacity of humans (Ustun & Rudin, 2014). However, one has to be careful with prediction models that are too simple (Freitas, 2014). Such models tend to oversimplify the underlying relationship between input features and output. This category supports the generation of local or global explainability.

Monotonicity: Monotonicity constraints are concerned with the relationship between input features and output. If this relationship is monotone, then the connection between input features and output can be more easily grasped by the user. A monotone relationship between an input and output is characterized by an increase in the input value leading to either an increase or a decrease in the output value. The monotonicity constraint can be incorporated into the system as either a hard or a soft constraint (Freitas, 2014). Furthermore, Martens et al. (2011) distinguish between the phases in the data mining process that incorporate the monotonicity constraint into the system. This category supports the generation of both local and global explainability.

Ontologies: An ontology is defined as a specification of a conceptualization in the context of knowledge sharing (Gruber et al., 1993). Ontologies can be applied to any type of interpretable model (surrogate or white box) and the corresponding data set. The ontology provides an extension or simplification for each explanation belonging to a set of all possible explanations as an output (either local or global). Ontologies are described in detail in Section 7.3.

3.5 The Explanation

Miller states that an explanation is the answer to a why question (Miller, 2019). Gkatzia et al. (2016) found out that human decision-making can be improved by *Natural Language Generation (NLG)*, especially in the area of uncertain data. According to Kass and Finin (1988), the quality of an explanation depends on three different criteria. These criteria are relevance, persuasiveness and comprehensibility. An explanation is relevant if it responds to the current goals and needs of the user. On the one hand, the explanation should provide as much information as possible to achieve these goals. On the other hand, it should be as short as possible in order to avoid providing additional information that is not necessary and could distract the user. An explanation is said to be convincing if the user accepts it. The user is convinced by an explanation if it is based on facts that the user believes to be true. Comprehensibility of an explanation is achieved through different facets. The explanation system should use a specific explanation type that the recipient is able to understand.

Beside the facets that an explanation should be short and highlight interesting aspects to the user, the explanation should also be simple so that the recipient does not have to look up too many unfamiliar terms (Kass & Finin, 1988). For a system that creates explanations, it is important to have a certain degree of flexibility and responsiveness. If the user fails to understand an explanation, the system should provide further knowledge in order to satisfy the user's needs (Swartout et al., 1991). Fischer et al. (1990) confirm the aspect of Kass and Finin (1988) that an explanation should be as brief as possible. They introduce an explanation system which generates minimal explanations. When the system receives feedback from the user that the explanation was not sufficient, it adds further details to the given explanation (Fischer et al., 1990). This approach is capable of satisfying the needs of the user without overburdening him. Furthermore, this approach tries to avoid complex explanations. Such explanations are too detailed or not well structured which makes it difficult for the user to understand them (Weiner, 1980).

The nature of data is also important for a convincing explanation, as some of the proposed dimensions can be influenced by different types of data. This means that, depending on the available data type, different explanatory approaches must be considered and the type of communication may change. For example, using a heat map to visualize the gradient of a model is better suited for image data than for tabular or text data. Biran and McKeown (2017) believe that people will only trust a prediction made by machine learning models if the system can justify its decision. Lipton (2018) describes that explanations should focus on abnormal reasons (Tversky & Kahneman, 1974). Lipton (2018) states that an interpretable model should be human-simulatable. Human-simulatable means that a user is able to "take in input data together with the parameters of the model and in reasonable time step through every calculation required to produce a prediction". To build an explanation, the following building blocks can be considered:

What *to explain (content type),* ***How*** *to explain (communication), to* ***Whom*** *is the explanation addressed (target group)*

3.5.1 Content Type

Depending on the model, different explanations can be generated. In order to explain a decision, we need to choose a certain type of explanation or stylistic element. We differentiate between the following types:

Local explanation: Local explainability is only concerned with an individual's decision (Phillips et al., 2017) and provides the reason (or reasons) behind a specific decision (Doshi-Velez & Kim, 2017). Local area explanations only regard the neighborhood of a data instance to explain the prediction.

Counterfactual (local) explanation: Counterfactual explanations (Wachter et al., 2018) provide data subjects, *e.g.*, a customer with meaningful explanations to understand a given decision, grounds to contest it, and advice how to change the decision to possibly receive a preferred one (e.g. loan approval).

Prototype (local) explanation: Prototype explainability is provided by reporting similar examples to explain the initial decision. Examples are prototypical instances that are similar to the unseen instance. According to Rüping (2006), providing examples helps to equip a model with explainability.

Criticism (local) explanation: Criticism (Rüping, 2006) supports prototypes since it detects what the prototypical instance did not capture.

Global explanation: Global explainability covers global dependencies to describe what a model focuses on in general. The global scope is concerned with the overall actions (Phillips et al., 2017) and provides a pattern that the prediction model discovered in general. The system can convey the behavior of a classifier as a whole without regarding predictions of individual instances (Lakkaraju et al., 2017).

3.5.2 Communication

The communication type determines how the explanation is communicated to the user.

Textual Description: Explainability is provided in text form. This mimics humans insofar as humans normally justify their decisions verbally. Examples of textual descriptions are generated captions of an image or explanation sentences that justify why a specific class was predicted.

Graphics: Explainability is provided in visual form. Visualization approaches try to illustrate what a model has learned by, e.g., depicting the parameters of the prediction model.

Multimedia: Explainability through multimedia combines different types of content: text, graphics, animation, audio and video.

3.5.3 Target Group

There are different groups who seek explainability. They pose different requirements for an explainable system due to the difference in experience and due to the different underlying goals (Weller, 2017). Therefore, explanations serve different purposes and can have different degrees of complexity. The user might be a non-expert and completely inexperienced with both machine learning and the domain. This user needs very simple and easily comprehensible explanations. Alternatively, the user might be a domain expert and familiar with the peculiarities of the data, although not familiar with machine learning. Such a user can be presented with more complex explanations and more subtle data. A domain expert could even feel offended if the explanations presented to him do not have a certain complexity (e.g., a doctor who prefers precise diagnoses over vague descriptions). Typically, a machine learning engineer will not be familiar with the domain but has a lot of experience with being exposed to complex topics. An explanation for an engineer can be more technical and may even contain internals of a model. Also related to the experience of the user is the time frame available to comprehend the explanation. An explanation that must be read in moments has to have a different look than an explanation that is to be fully understood within days. We consider the following groups:

Non-Expert: The non-expert uses the system. He neither has technical nor domain knowledge but receives the decision. His goal is to understand why a certain prediction was made, *e.g.*, why his credit was denied. Furthermore, he could be interested in understanding whether the explanation domain is composed of simple or complex input features.

Domain-Expert: A domain expert also uses the system but he is an expert regarding the domain of application. His goal is to more deeply understand the system and the factors and features that are used by the system in order to incorporate his domain knowledge into

the system and finally trust the system and its predictions. The domain expert considers trust an important requirement for system deployment.

System-Developer: A developer or system-designer builds the system and is a technical expert that usually does not have any domain knowledge. His goal is to really understand how the algorithm works and what features it exploits. He is concerned with the overall functionality of the system.

AI-Developer: The AI expert trains the model. He is a technical expert in the field of AI without any or only little domain knowledge. His goal is to understand how the algorithm works and what features it exploits so that he can debug and improve the model from a technical standpoint , *e.g.*, the accuracy.

3.6 Assessment of Explainability

After we elaborated on the building blocks of an explanation, we now want to discuss some suitable metrics for explainability. Miller (2019) did a survey of psychological studies to find out what humans actually consider a *good* explanation. His major findings are that an explanation should be crucially contrastive, concise, selected, social and that probabilities are less important. Furthermore, explanations should go beyond statistics and imply causality. They need to make certain assumptions about the target's beliefs.

Contrastive explanations should clarify why an input converted into a specific output and not into some counterfactual contrast output (Hilton, 1990). Saying that explanations should be concise means that explanations that are too long are not considered interpretable by humans. Explanations are also considered better if they confirm the beliefs of the addressee. This is generally known as *confirmation bias*. Explanations should fit the social context of the person that judges the interpretability. This means that the explanation should not only fit the knowledge of this person but also how she sees herself and her environment. Furthermore, explanations should not contain probability measures since most humans struggle to deal with uncertainty (Miller, 2019). Tversky and Kahneman (1981) note that explanations should also focus on abnormal reasons. Humans prefer rare events as explanations over frequent ones. All of these attributes are hard to measure. In order to asses explainability, Doshi-Velez and Kim (2017) propose three levels for tests: First, experiments on real-world tasks with humans; second, simple, experimental tasks with humans; and third, proxy tasks and metrics where other studies with the above assessment methods validate that they are good surrogates. When a system is evaluated for interpretability based on a proxy, usually complexity and sparsity measures are used. Especially for rule- and tree-based models, there are different measures. There are measures for sparsity that are concerned with the total number of features used (Su et al., 2015). For decision trees, sparsity measures the total number of features used as splitting features (Craven & Shavlik, 1996). For rule-based models, the size measures the total number of rules in a decision set (Lakkaraju et al., 2016), (Lakkaraju et al., 2017) or decision list (Bertsimas et al., 2011), (Letham et al., 2015). The length of a rule measures the total number of predicates used in the condition of the decision set (Lakkaraju et al., 2016) or decision list (Bertsimas et al., 2011). The length of each single rule can be accumulated to measure the total number of predicates used (Lakkaraju et al., 2017). Furthermore, measures for the total number of data instances that satisfy a rule (called *cover* in Lakkaraju et al. (2016)) and measures for

the total number of data instances that satisfy multiple rules (called *overlap* in Lakkaraju et al. (2016)) can be used to measure complexity. Freitas (2014) suggests a different measure for the complexity of a rule-based model. He argues that the average number of rule conditions which were considered for making predictions is a better complexity measure. For tree-based models, *size* measures the number of tree nodes (Craven & Shavlik, 1996). Another measure is their *depth* (Rüping, 2005). Furthermore, feature importance can be extracted from rule- and tree-based models. The total number of instances that use a feature for classification can be used as the feature's importance. Samek et al. (2017) uses perturbation analysis for measuring explainability. This method is based on three simple ideas. First, the predictive value of a model suffers more from the disturbance of important input features than from unimportant features. Second, approaches such as Sensitivity Analysis or Layer Wise Relevance Propagation provide a feature score that makes it possible to sort them. Third, it is possible to change the input values iteratively and document the predicted value. The averaged prediction score can be used to measure the explanation quality. If the averaged prediction score fluctuates, it can be an indication for important or unimportant parameters of an explanation.

Other measurement methods are introduced for recommender systems (Abdollahi & Nasraoui, 2017). The similar based approach analyzes the user's neighborhood. The more users from the neighborhood recommend a product, the better this majority vote can be used as a basis for an explanation. The approach from Abdollahi and Nasraoui (2016) also uses neighborhood style explanations. An explainability score with a range of zero to one is used to determine whether the user rated a product in a similar way that his neighborhood users did. If the value is zero, the user is not contributing to the user-based neighbour-style explanation. Another important aspect that is needed during the testing of an explanatory process is the ground truth. Therefore, an exploration environment such as the Kandinsky Patterns can be used (Holzinger et al., 2019a). The work of Murdoch et al. (2019) introduces the *Predicitve, Descriptive, Relevant (PDR) framework*. This framework uses three components for the interpretation generation and evaluation: *predictive accuracy*, *descriptive accuracy*, and *relevancy*. *Predictive accuracy* is the error measure for given interpretations. *Descriptive accuracy* describes the relationships that models learn. *Relevancy* is given if the explanation provides meaningful information for a specific audience (Murdoch et al., 2019).

However, measuring explainability is still a complicated task due to its subjectivity. Whether an explanation is considered satisfactory depends on the knowledge, needs and objectives of the addressees. Adadi and Berrada (2018) mention another challenge that makes measuring explainability difficult. They describe that ML models often have a complex structure and that this can lead to the fact that for the same input variables with the same target values, the algorithm generates different models because the algorithm passes through different paths during execution. This, in turn, leads to different explanations for the same data, making it difficult to accurately measure the explainability.

Users may have different types of questions they want to ask. In the work of Hoffman et al. (2018), each domain expert had the possibility to rate explanations by a scoring system. The result was summed up and then divided by the total amount of participants. The *Content Validity Ratio (CVR)* score was then transformed to a range between -1 and +1. A score above 0 indicates that the item is considered meaningful according to the explanation

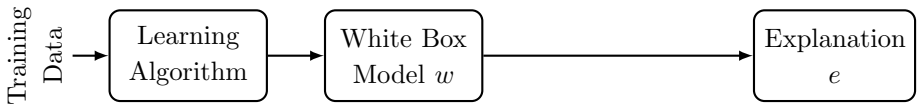


Figure 3: Learning white box models according to Figure 1(c).

satisfaction. Furthermore, Hoffman et al. (2018) present the *Explanation Goodness Checklist* for measuring the goodness of explanations. Here, the application context differs from the one used for the *Explanation Satisfaction Scale*. While the Explanation Satisfaction Scale collects opinions of participants after they have worked with the XAI system, the Explanation Goodness Checklist is used as an independent evaluation of explanations by other researches.

In what follows, we will survey past and recent explainability approaches in SML, mainly in classification. We will conclude each chapter that describes a category according to the definitions in section 3.1 with an illustrative example on the well-known IRIS data set for the task of classification (Dua & Graff, 2017). The instance we use for the local procedures is taken from the class *virginica* with the feature values $[5.8, 2.8, 5.1, 2.4]$.

4. Interpretable Model Learning

While in Section 3.3, we introduced various kinds of interpretable model types, this section focuses on the actual *learning algorithms* that produce these interpretable models. We differentiate between algorithms plainly providing interpretable models—named *interpretable by nature* in the following—and algorithms putting special emphasize on enhancing interpretability—named *interpretable by design*. This section refers to Problem 2 in Section 3.1 and the explanation generation process is outlined in Figure 3.

4.1 Interpretable by Nature

A closer look at some classic SML algorithms shows that they provide models that can already be interpreted without forcing interpretability on them. This means that the training process for the learning algorithms is not explicitly optimized to gain interpretability. Usually, these algorithms are optimized for high accuracy; interpretability is a by-product. It is naturally given and therefore inherent. The resulting models can be explained directly, e.g. through visualizations, in order to gain an understanding of the model's function and behavior. However, in order to be easily understood by humans, the models must be sparse, small and simple. The overview of the different approaches is listed in Table 2.

4.1.1 Decision Trees

The *Classification and Regression Trees (CART)* (Breiman, 2017) algorithm is a divide and conquer algorithm that builds a decision tree from the features of the training data. Splitting features and their values are selected using the Gini index as the splitting criterion. The *ID2of3* (Craven & Shavlik, 1996) algorithm is based on M-of-N splits learned for each node using a hill climbing search process for the construction of the decision tree (Craven

& Shavlik, 1996). The *C4.5* algorithm (Quinlan, 1996) is a divide and conquer algorithm that is based on concepts from information theory. A gain ratio criterion based on entropy is used as the node splitting criterion. The *C5.0T* algorithm is an extension of the C4.5 algorithm (Ustun & Rudin, 2016). The *ID3* algorithm chooses attributes with the highest information gain.

4.1.2 Decision Rules

1R (Holte, 1993) uses as input a set of training examples and produces as output a 1-rule. 1R generates rules that classify an object on the basis of a single attribute. The *AntMiner+* algorithm uses ant-based induction that creates one rule at a time (Martens et al., 2007b). The $cAntMiner_{PB}$ ordered algorithm uses ant-based induction to create an ordered list of rules that consider interactions between individual rules whereas its unordered version creates an unordered set of rules (Otero & Freitas, 2016). Others are the *CN2* algorithm (Martens et al., 2009), *RIPPER* algorithm (Cohen, 1995), the *Re-RX* algorithm (Setiono et al., 2008) and the *C5.0R* algorithm (Ustun & Rudin, 2016).

4.1.3 Generalized Additive Models (GAMs)

GAMs learn a linear combination of shape functions in order to visualize each feature's relationship with the target (Lou et al., 2012). A shape function relates a single feature to the target. Hence, a shape function's visualization illustrates the feature's contribution to the prediction. Shape functions can be regression splines, trees or ensembles of trees. However, interactions between features are not considered by GAMs. Therefore, Lou et al. (2013) introduce interaction terms to GAMs. The *Generalized Additive Models plus Interactions* GA^2Ms considers pairwise interactions of features. These interaction terms add more accuracy to the prediction model while maintaining the prediction model's interpretability. Caruana et al. (2015) proofed the explainability of the model while remaining the state-of-the-art accuracy on a pneumonia case study.

4.1.4 The Use Case

The decision tree presents the correlations in the data set and provides a simple overview over the contributions that each feature is likely going to have on the data. Decision trees thus allow for ante-hoc model explanations. To train the tree, we need a data set with labeled outputs and the corresponding class labels *virginica, setosa* and *versicolor*. Figure 4 illustrates a decision tree with the depth of two. The nodes illustrate which feature is considered or, more precisely, which decision is made to split the data set into the branches. Illustratively, the tree first splits all instances at *petal width 0.8*. Thereby, all instances with petal width smaller than 0.8 are classified as *setosa*, while everything else is further divided by petal length. The user is able to grasp the concept of the decision-making of the decision tree. Figure 5 illustrates a decision tree with the depth of three. The tree first splits all instances at *(petal width* $\leqq$ *2.45)*, whereas the next split is at *(petal width* $\leqq$ *1.75)*. The two last splits are either at *(petal length* $\leqq$ *4.95)* or at *(petal length* $\leqq$ *4.85)*. Those two examples clearly illustrate that a decision tree with the depth of two or three is still comprehensible.

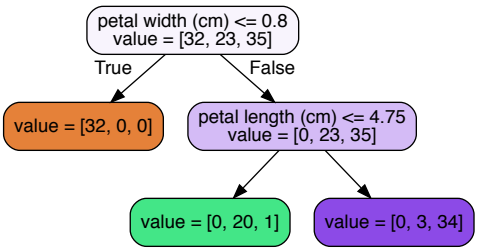


Figure 4: Tree with depth of 2

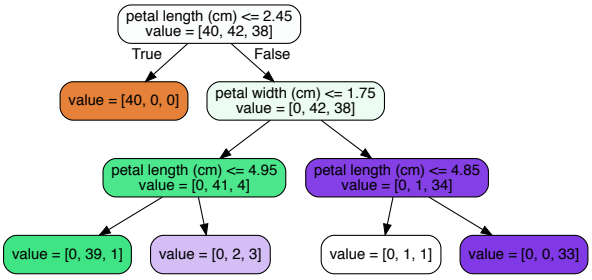


Figure 5: Tree with depth of 3

4.2 Interpretable by Design

An alternative approach to directly train interpretable models is to include interpretability in the design of the training. We call this learning approach "interpretable by design". As was the case with the "by nature" approaches discussed above, the result usually is a white box model but in contrast, interpretable by design approaches allow the degree of interpretability to be controlled or increased. This means that the resulting models are created with the intention of further improving the interpretability for humans. Section 3.4 lists some general-purpose techniques that can be used to achieve interpretability by design.

4.2.1 Decision Trees

Oblique Treed Sparse Additive Models (OT-Spam) belong to the category of region-specific predictive models (Wang et al., 2015a). Oblique Trees are used to divide the feature space into smaller regions. Sparse Additive Models called experts are trained for each local region. These experts are assigned as leaf nodes to the oblique tree to make the predictions. For this specific approach, *Factorized Asymptotic Bayesian* (FAB) inference was used to build the prediction model. The size of the tree is also regularized by the FAB inference. Another work presents *Neural Decision Tree (NDT)*, an interplay of decision trees with neural networks. This model has the structure of a decision tree where the splitting nodes consist of independent perceptrons. NDT can be used for supervised and unsupervised problems and takes the advantages of both methods: the clear model architecture of decision trees and the high-performance capacities of neural networks (Balestriero, 2017). Yang et al. (2018b) presents *Deep Neural Decision Tree (DNDT)* which are similar to Balestriero (2017)'s model. This approach is specifically designed for tabular data that learns through backpropagation. DNDTs differ from Balestriero (2017)'s method in that each input feature has its own neural network, making it easier to interpret.

4.2.2 Decision Rules

The *Bayesian Or's of And's* (BOA) (Wang et al., 2015b) model is based on a Bayesian approach that allows the incorporation of priors. These priors are used to regulate the size and shape of the underlying rules. Wang introduces the Beta-Binomial prior and the Poisson prior. The Beta-Binomial prior rules the average size of the conditions, whereas

the Poisson prior rules the condition length and the overall model size. To learn such rules, association rule mining and simulated annealing or literal-based stochastic local search is performed.

Two-Level Boolean Rules are either of the form AND-of-ORs or of the form OR-of-ANDs (Su et al., 2015). Since a rule consists of two levels, a rule of the form OR-of-ANDs consists of two conditions connecting predicates with ANDs that are connected with an OR. The proposed approach allows for the incorporation of accuracy and sparsity in form of a trade-off parameter. Two optimization based formulations are considered to learn two-level Boolean rules. One is based on the 0-1 loss, the other is based on the Hamming distance. Furthermore, when the 0-1 loss is used, they use a redundancy-aware binarization method for two-level LP-relaxation. When the Hamming distance is used, they use block coordinate descent or alternating minimization. The same idea as in *SLIM* (Ustun & Rudin, 2016) can be used to construct M-of-N rule tables. *TILM* uses thresholds of feature values and a feature selection process to find binary decision rules (Ustun & Rudin, 2014).

The *Ordered Rules for Classification* model introduces a decision list classifier that is based on mixed integer optimization (MIO) and association rules (Bertsimas et al., 2011). The set of general association rules is found using MIO. MIO is also used for ordering these rules and incorporating other desiderata such as the number of features used in a condition and the total number of rules.

The *Bayesian List Machine* (BLM) model is based on a Bayesian framework that learns a decision list. Sparsity as the trade-off between interpretability and accuracy can be introduced into the Bayesian framework. The learning process consists of first finding the rules and then ordering them (Letham et al., 2012).

The *Bayesian Rule List* (BRL) method differs from BLM just by a different prior (Letham et al., 2015). They use a Bayesian framework and incorporate sparsity into the model. As output, BRL provides class probabilities. *Scalable BRL* (Yang et al., 2016) is an extended version of BRL that provides a posterior distribution of the rule list.

Falling Rule Lists (FRL) are decision lists that are ordered based on class probabilities. With each rule going down the list, the probability of belonging to the class of interest decreases. Furthermore, once an unseen instance matches a rule it is assigned the probability of the rule it matches instantaneously. Since Falling Rule Lists are based on a Bayesian framework, a prior for the size of a decision list can be integrated into the learning process (Wang & Rudin, 2014).

Malioutov et al. (2017) propose a rule-based classifier by applying a linear programming (LP) relaxation for interpretable sparse classification rules. *Classification based on Predictive Association Rules* (CPAR) uses dynamic programming to generate a small set of high quality and lower redundancy (Yin & Han, 2003). *Two-Level Boolean Rules* (TLBR) create classification predictions by connecting features with logical statements in rules (Su et al., 2016).

4.2.3 Decision Sets

The goal of *Interpretable Decision Sets* (IDS) is to learn a set of short, accurate, non-overlapping rules. These learned rules can be used independently of each other. (Lakkaraju et al., 2016) use a pre-mined rule space on top of which they apply association rule mining.

Only frequent item-sets are considered and mined. In the process of selecting a set of rules, different objectives that include accuracy and interpretability are optimized.

Bayesian Rule Set (BRS) is another approach that introduces interpretable sets of rules (Wang et al., 2016). This approach is based on a Bayesian framework that incorporates the total number of rules as a sparsity prior. Furthermore, for each rule in the set, the minimum support requirement has to be met.

4.2.4 Linear Models

Ustun and Rudin (2014) introduces an *integer programming framework*. In addition to the creation of various interpretable models, scoring systems can also be created by this framework. Scoring systems assign each feature or feature interval a weight. A final score is evaluated in the same way as the prediction of a linear model. However, an instance is classified only after comparing the final score with a defined threshold.

Supersparse linear integer models (SLIM) is a method used for making scoring systems more interpretable (Ustun & Rudin, 2014). The authors introduce a pareto-optimal trade-off between accuracy and sparsity. Here, the model is considered to be interpretable when it is sparse. In order to achieve this trade-off, an integer program that encodes 0-1 loss for accuracy and L_0 seminorm for sparsity together with a regularization parameter for the trade-off is used to learn SLIM. PILM is a generalization of SLIM.

4.2.5 Surrogate Fitting

The concept of interpretable by design can be extended so as to also include approaches that learn a black box model but that are optimized towards improving surrogate model fitting. That is, the learning algorithm of the black box model already is designed in such a way that fitting the surrogate to the black box model is simplified or results in a higher fidelity. Wu et al. (2018) for instance achieve this by using regularization during training a neural network that forces the network to allow a small decision tree to be fitted as a global surrogate model. A similar approach is considered in Schaaf and Huber 2019, where the used L_1-orthogonal regularization allows significantly faster training. They state that this preserves the accuracy of the black box model while it can still be approximated by small decision trees. Burkart et al. (2019) use a rule-based regularization technique to enforce interpretability for neural networks.

4.2.6 The Use Case

Bayesian Rule List (BRL) construct rules to be accurate but still interpretable for the users. The approach resembles simple logical decision-making. In this case, the rules are model-specific and support the model interpretability. The algorithm requires the labeled data to mine a decision list. Listings 1, 2, 3 illustrate the results for classification discriminating each class separately. The rules can be read fairly comfortably and they can tell the user the direct relationship of a feature value with the classification probability that results from that value. These rules apply to all instances alike. The user can quickly learn the knowledge the model has deducted from the data set and thereby grasp the decision-making process. For example, the *setosa* classification rules apply to *setosa* and *not setosa* classification.

```
IF petal width: 0.8 to inf THEN probab. of setosa: 1.2%
```

Table 2: Overview of interpretable by nature approaches.

Approach	**Learning Task**	**Model**	**References**
GAMs	Classification	Linear Model	Lou et al. (2012)
CN2	Classification	Rule-based	Clark and Niblett (1989)
RIPPER	Classification	Rule-based	Cohen (1995)
Re-RX	Classification	Rule-based	Setiono et al. (2008)
C5.0R	Classification	Rule-based	Ustun and Rudin (2016)
AntMiner+	Classification	Rule-based	Martens et al. (2007a)
cAntMinerPB	Classification	Rule-based	Otero and Freitas (2016)
CART	Classification & Regression	Tree-based	Breiman (2017)
ID2of3	Classification & Regression	Tree-based	Craven and Shavlik (1996)
ID3	Classification & Regression	Tree-based	Quinlan (1986)
C4.5	Classification & Regression	Tree-based	Quinlan (2014)
C5.0T	Classification & Regression	Tree-based	Ustun and Rudin (2016)
Bayesian Network	Classification	Bayesian Network	Friedman et al. (1997)
Linear regression (Lasso)	Regression	Linear model	Tibshirani (1996)
Linear regression (LARS)	Regression	Linear model	Efron et al. (2004)
Logistic regression	Classification	Linear model	Berkson (1953)
KNN	Classification & Regression	Nearest Neighbor	Freitas (2014)
MGM	Classification	Clustering	Kim et al. (2015)
AOT	Classification	Tree-based	Si and Zhu (2013)

```
ELSE IF petal length: -inf to 2.45 THEN probab. of setosa: 97.4%
ELSE probab. of setosa: 50.0%
```

Listing 1: Setosa Classification

```
IF petal length: 2.45 to 4.75 THEN probab. of versicolor:97.3%
ELSE IF petal width: 0.8 to 1.7 THEN probab. of versicolor:42.9%
ELSE probab. of versicolor: 2.4%
```

Listing 2: Versicolor Classification

```
IF petal length: 5.15 to inf THEN probab. of virginica: 96.6%
ELSE IF petal length:-inf to 4.75 THEN probab. of virginica:2.6%
ELSE IF petal width:-inf to 1.75 THEN probab. of virginica:25.0%
```

Listing 3: Virginica Classification

5. Surrogate Models

This section is concerned with discussing various approaches on fitting global or local surrogates to a black box model or a model prediction, respectively. A surrogate model translates the model into an approximate model (Henelius et al., 2014) either local or global. This technique is applied whenever the model is not interpretable by itself, i.e., whenever it is a black box. An interpretable model is build on top of the black box. Separating the prediction model from its explanation introduces flexibility, accuracy and usability (Ribeiro et al., 2016a).

Table 3: Overview of interpretable by design approaches.

Approach	Learning Task	Model	References
SLIM	Classification	Rule-based	Ustun and Rudin (2014)
TILM	Classification	Rule-based	Ustun and Rudin (2016)
PILM	Classification	Linear model	Ustun and Rudin (2016)
RiskSLIM	Classification	Rule-based	Ustun and Rudin (2017)
Two-level Boolean Rules	Classification	Rule-based	Su et al. (2015)
BOA	Classification	Rule-based	Wang et al. (2015b)
ORC	Classification	Rule-based	Bertsimas et al. (2011)
BLM	Classification	Rule-based	Letham et al. (2012)
BRL	Classification	Rule-based	Letham et al. (2015)
(S)BRL	Classification	Rule-based	Yang et al. (2016)
FRL	Classification	Rule-based	Wang and Rudin (2015)
IDS	Classification	Rule-based	Lakkaraju et al. (2017)
BRS	Classification	Rule-based	Wang et al. (2016)
OT-SpAM	Classification	Tree-based	Wang et al. (2015a)
Tree Regularization	Classification	Tree-based	Wu et al. (2018)
NDT	Classification	Tree-based	Balestriero (2017)
DNDT	Classification	Tree-based	Yang et al. (2018b)
LP relaxation	Classification	Rule-based	Malioutov et al. (2017)
1R	Classification	Rule-based	Holte (1993)
TLBR	Classification	Rule-based	Su et al. (2016)
CPAR	Classification	Rule-based	Yin and Han (2003)

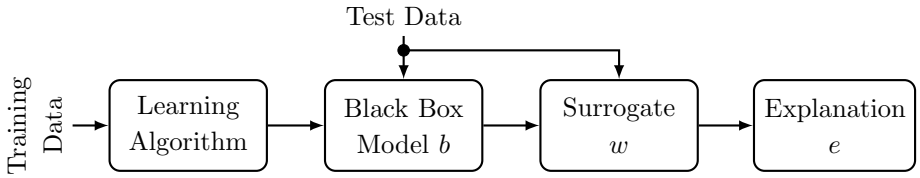


Figure 6: Fitting a global surrogate according to Figure 1(d).

5.1 Global Surrogates

This section describes global surrogate models that learn interpretable models to mimic the predictions of a black box model. The overview of the different approaches are listed in Table 4 and 5. The global surrogate fitting process is depicted in Figure 6.

5.1.1 Linear Models

The *sub-modular pick algorithm* (sp-Lime) provides global explanations by using sub-modular picks to find representative instances of the underlying prediction model (Ribeiro et al., 2016c). The explanations offered by Lime for these representative instances are combined and considered as the global explanation of the black box. Another variation is k-Lime, where a clustering algorithm is applied to the black box model to find k clusters (Hall et al., 2017a). These clusters represent local regions. For each local region, a *Generalized Linear Model* (GLM) is learned. A global surrogate GLM is used as an explanation when the unseen data instance does not fall into a local region.

Table 4: Overview of global surrogate approaches.

Approach	**Learning Task**	**Input Model**	**Model**	**References**
SP-Lime	Classificiation	Agnostic	Linear Model	Ribeiro et al. (2016c)
k-Lime	Classificiation	Agnostic	Linear Model	Hall et al. (2017a)
Tree Merging	Classificiation	Specific (DT)	Tree-based	Andrzejak et al. (2013)
Decision Tree extract	Classificiation	Specific (DT)	Tree-based	Bastani et al. (2017)
Soft Decision Tree	Classificiation	Agnostic	Tree-based	Hinton and Frosst (2017)
Binary Decision Tree	Classificiation	Agnostic	Tree-based	Yang et al. (2018a)
Probabilistic interpretation	Classificiation	Specific (DT)	Tree-based	Schetinin et al. (2007)
EM min. Kullback	Classificiation	Specific (DT)	Tree-based	Hara and Hayashi (2016)

5.1.2 Decision Trees

Andrzejak et al. (2013) introduce an approach to merge two decision trees into a single one based on distributed data. The procedure utilizes an efficient pruning strategy that is based on predefined criteria.

The *Decision Tree Extraction* (Bastani et al., 2017) approach extracts a new model from a given black box model. The resulting surrogate model is then used as an approximation of the original black box model in the form of a decision tree. First, the algorithm generates a Gaussian Mixture Model (GMM) to cluster the data points in the training data set. The second step comprises the computation of class labels for the clustered data points by utilizing the black box model to approximate.

Hinton and Frosst (2017) illustrate a soft decision tree which is trained by stochastic gradient descent using the predictions of the neural net. The tree uses learned filters to make hierarchical decisions based on an input example.

Yang et al. (2018a) propose a binary decision tree that represents the most important decision rules. The tree is constructed by recursively partitioning the input variable space by maximizing the difference in the average contribution of the split variable.

Schetinin et al. (2007) describe the probabilistic interpretation of Bayesian decision tree ensembles. Their approach consists of the quantitative evaluation of uncertainty of the decision trees and allows experts to find a suitable decision tree.

Hara and Hayashi (2016) approximate a simple model from tree ensembles by deriving the expectation maximization algorithm minimizing the Kullback-Leibler divergence.

5.1.3 Rule-Based

Rule extraction approaches learn decision rules or decision trees from the predictions made by black boxes. They are divided into pedagogical, decompositional (see Figure 7) and eclectic approaches (Andrews et al., 1995; Martens et al., 2011). The pedagogical approach (model-agnostic) perceives the underlying prediction model as a black box and uses the provided relation between input and output. The decompositional approach (model-specific) makes use of the internal structure of the underlying prediction model. The eclectic or hybrid approach combines the decompositional and the pedagogical. The aforementioned

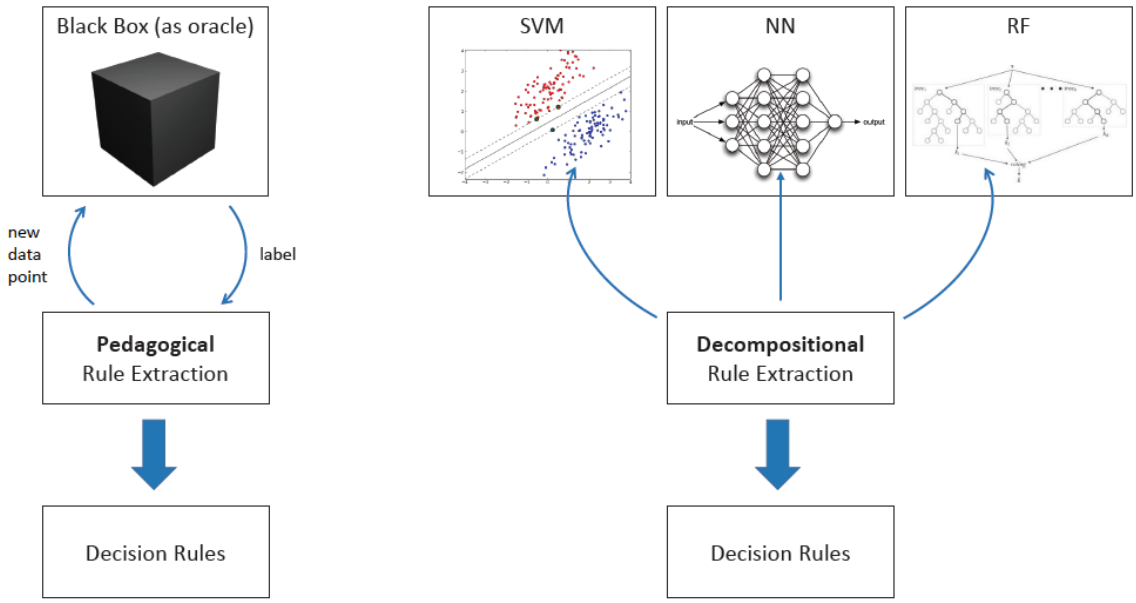


Figure 7: Difference between pedagogical and decompositional rule extraction approaches.

approaches regarding interpretable decision rules and decision trees can also be applied to black boxes. The approaches are listed in Table 5.

The rule extraction techniques have an advantage over interpretable decision rules or trees. Although a combination of features is not present in the training data, the rule extraction technique can still work since its underlying prediction model labels any feature combination. Decision rules depend on the available training data and cannot augment their training set (Craven & Shavlik, 1996).

5.1.4 Pedagogical Decision Rules

The *Building Representations for Artificial Intelligence using Neural Networks (BRIANNE)* algorithm was originally proposed for Artificial Neural Network (ANN). The algorithm measures the relevance of a specific feature for some output. This measure is then used to build rule conditions. The measure determines the features and the features' values of a condition (Biswas et al., 2017).

The *Validity Interval Analysis (VIA)* (Zilke et al., 2016) algorithm aims at finding provably correct rules. This is done by applying an approach similar to sensitivity analysis. The *Binarized Input-Output Rule Extraction (BIO-RE)* (Biswas et al., 2017) algorithm applies a sampling based approach that generates all possible input combinations and asks the black box for their predictions. Based on that, a truth table is build on top of which an arbitrary rule-based algorithm can be applied to extract rules.

There is a class of rule extraction approaches utilizing genetic programming, which is motivated by Darwin's theory on survival of the fittest. The *Genetic-Rule Extraction (G-Rex)* algorithm for instance chooses the best rules from a pool of rules generated by

Table 5: Overview of surrogate rule extraction approaches and their properties of explainability.

Approach	Extraction Approach	Learning Task	Model Scope	Model	References
BRAINNE	Feature Selection	Classification	Pedagogical	Rule-based	Sestito and Dilan (1992)
VIA	Sensitivity analysis	Classification	Pedagogical	Rule-based	Zilke et al. (2016)
BIO-RE	Feature Selection	Classification	Pedagogical	Rule-based	Taha and Ghosh (1996)
G-Rex	Genetic algorithm	Classification Regression	Pedagogical	Rule-based	Johansson et al. (2004)
STARE	Feature Selection	Classification	Pedagogical	Rule-based	Zhou et al. (2000)
REFNE	Ensemble concept	Classification	Pedagogical	Rule-based	Zhou et al. (2003)
BUR	Ensemble concept	Classification	Pedagogical	Rule-based	Ninama (2013)
Iter	Sequential Covering	Classification Regression	Pedagogical	Rule-based	Huysmans et al. (2006)
OSRE	Orthogonal search-based	Classification	Pedagogical	Rule-based	Etchells and Lisboa (2006)
Minerva	Sequential Covering	Classification Regression	Pedagogical	Rule-based	Martens et al. (2008)
RxREN	Reverse engineering	Classification	Pedagogical	Rule-based	Augasta and Kathirvalavakumar (2012)
RxNCM	Correctly and misclassified data ranges	Classification	Pedagogical	Rule-based	Biswas et al. (2017)
BETA	Reverse engineering	Classification	Pedagogical	Rule-based	Lakkaraju et al. (2017)
KDRuleEX	Genetic algorithm	Classification	Pedagogical	Rule-based	Sethi et al. (2012)
TREPAN	Symbolic Learning	Classification	Pedagogical	Tree-based	Craven and Shavlik (1996)
ANN-DT	Interpolated sample outputs	Classification Regression	Pedagogical	Tree-based	Schmitz et al. (1999)
DecText	Improved splitting (SetZero) and discretization	Classification	Pedagogical	Tree-based	Boz (2002)
REx	Genetic algorithm	Classification	Pedagogical	Rule-based	Kamruzzaman (2010)
GEX	Genetic algorithm	Classification	Pedagogical	Rule-based	Markowska-Kaczmar and Chumieja (2004)
DeepRED	Tree induction	Classification	Decompositional [ANN]	Rule-based	Zilke et al. (2016)
SUBSET	Feature Importance	Classification	Decompositional [SVM]	Rule-based	Biswas et al. (2017)
MofN	Feature Importance	Classification	Decompositional	Rule-based	Setiono et al. (2014)
Knowledgetron	Feature Selection (heuristic search)	Classification	Decompositional [ANN]	Rule-based	Fu (1994)
NeuroRule	Feature Importance	Classification	Decompositional [ANN]	Rule-based	Lu et al. (1995)
RX	Feature Importance	Classification	Decompositional [ANN]	Rule-based	Biswas et al. (2017)
NeuroLinear	Discretization of hidden unit activation values	Classification	Decompositional [ANN]	Rule-based	Setiono and Liu (1997)
full-RE	Feature selection	Classification	Decompositional [ANN]	Rule-based	Biswas et al. (2017)
FERNN	Feature Importance	Classification	Decompositional [ANN]	Tree & Rule-based	Biswas et al. (2017)
CRED	Tree induction	Classification	Electic [ANN]	Tree & Rule-based	Zilke et al. (2016)
ANNT	Tree induction	Classification	Decompositional [ANN]	Rule-based	Biswas et al. (2017)
GRG	Clustering	Classification	Model-specific	Rule-based	Odajima et al. (2008)
E-Re-RX	Ensemble concept	Classification	Decompositional [ANN]	Rule-based	Hayashi (2013)
X-TREPAN	Tree induction	Classification Regression	Model-specific (Decompositional [ANN])	Tree-based	Biswas et al. (2017)
RuleFit	Predictive Learning	Classification Regression	Decompositional [RF]	Linear Model & Rule-based	Friedman et al. (2008)
Node harvest	Feature Selection	Classification Regression	Decompositional [RF]	Rule-based	Meinshausen (2010)
DHC	Scoring Function	Classification Regression	Decompositional [RF]	Rule-based	Mashayekhi and Gras (2017)
SGL	Feature Importance	Classification Regression	Decompositional [RF]	Rule-based	Mashayekhi and Gras (2017)
multiclass SGL	Scoring Function	Classification	Decompositional [RF]	Rule-based	Mashayekhi and Gras (2017)
SVM+Prototypes	Clustering	Classification	Decompositional [SVM]	Rule-based	Martens et al. (2007b)
Fung	Clustering	Classification	Decompositional [SVM]	Rule-based	Fung et al. (2008)
SQRex-SVM	Sequential Covering	Classification	Decompositional [SVM]	Rule-based	Barakat and Bradley (2007)
ALBA	Active Learning	Classification	Decompositional [SVM]	Rule-based	Martens et al. (2009)

the black box and combines them with a genetic operator (Martens et al., 2009, 2007b). The *REX* algorithm was originally proposed to extract rules from ANNs. However, it can be applied to any black box. It uses genetic programming to extract fuzzy rules (Ninama, 2013). The Genetic Rule Extraction (GEX) algorithm is a genetic programming algorithm that uses sub-populations based on the number of different classes that are present in the training data. The algorithm extracts propositional rules from the underlying black box (Martens et al., 2009).

The *Statistics based Rule Extraction (STARE)* algorithm is based on breadth first search. Furthermore, the input data is permuted in order to produce a truth table that is used for rule extraction. The *Rule Extraction From Neural Networks Ensemble (REFNE)* algorithm uses ANN ensembles to generate instances that are then used to build decision rules.

The *Bust Unordered Rule (BUR)* algorithm is based on the gradient boosting machine. The algorithm first learns and then prunes the decision rules (Ninama, 2013). *Iter* (Martens et al., 2009) is a sequential covering algorithm that learns one rule at a time. It randomly generates extra data instances and uses the black box as an oracle to label these data instances. The *Orthogonal Search based Rule Extraction (OSRE)* algorithm was originally proposed for ANNs and Support Vector Machines (SVMs). It converts the given input to a desired format and uses activation responses to extract rules (Etchells & Lisboa, 2006). *Minerva* (Martens et al., 2008) is a sequential covering algorithm that uses iterative growing to extract decision rules from a black box.

The *Rule Extraction by Reverse Engeneering (RxREN)* (Biswas et al., 2017) algorithm uses reverse engineering for input feature pruning. Input features are pruned when their temporary omission does not change the classification output significantly. The *Rule Extraction from Neural Networks using Classified and Miss-classified data (RxNCM)* algorithm is based on a modification applied to RxREN. RxNCM uses classified and misclassified data to figure out per class ranges for significant features. It also differs from RxREN in the black box pruning step. Input features are only pruned when their absence increases prediction accuracy. Based on that, rules are extracted from the black box (Biswas et al., 2017).

The *Black box Explanations through Transparent Approximations (BETA)* algorithm applies two level decision sets on top of black box predictions. The first level specifies the neighborhood of a rule (subspace descriptor) and the second level introduces decision rules that are specific for each region (Lakkaraju et al., 2017).

5.1.5 Decompositional Decision Rules from SVMs

There are techniques that use a SVM as the underlying prediction model and that extract rules from the SVM. The *SVM+Prototypes* (Martens et al., 2007b) approach first separates the two classes using a SVM. Per subset, the algorithm uses clustering and finds prototypes for each cluster. It further uses support vectors to create rule defining regions.

The rule extraction technique proposed by Fung is limited to linear SVMs and extracts propositional rules (Martens et al., 2009). *SQRex-SVM* uses a sequential covering algorithm that is only interested in correctly classified support vectors. The approach is limited to binary classification since only rules for the desired class are extracted (Martens et al., 2009). The *Active Learning-Based Approach (ALBA)* (Martens et al., 2009) extracts rules from SVMs by using support vectors. The algorithm re-labels the input data based on the

predictions made by the SVM and generates extra data close to the support vectors that are also labeled by the underlying SVM.

5.1.6 Decompositional Decision Rules from ANNs

There are techniques that use ANNs or Deep Neural Networks (DNNs) as the underlying prediction model and extract rules from the ANN. The *SUBSET* method finds subsets of features that fully activate hidden and output layers. These subsets are then used for rule extraction from ANNs (Biswas et al., 2017). The *MofN* algorithm is an extension of the SUBSET algorithm. The goal is to find connections with similar weights. This can be done using clustering. The extracted rules label an unseen data instance with the rule label when M of the N conditions of the rule are met (Biswas et al., 2017) (Zilke et al., 2016). The *Knowledgetron (KT)* algorithm extracts rules for each neuron based on the input neurons that are responsible for another neuron's activation. This is the basis for the final rule extraction step (Biswas et al., 2017) (Zilke et al., 2016). The *NeuroRule* algorithm first applies pruning to the ANN in order to remove irrelevant connections between neurons. An automated rule generation method is then used to generate rules that cover the maximum number of samples with the minimum number of features in the condition of the rule. However, continuous data needs to be discretized first (Biswas et al., 2017). The *RX* algorithm can only be applied to ANNs with just one hidden layer. First, the ANN is pruned to remove irrelevant connections between neurons. Then, neuron activations are used to generate rules. Another limitation of this approach is the restriction to discrete data. The *NeuroLinear* algorithm extracts oblique decision rules from the underlying ANN (Biswas et al., 2017). The *full-RE* algorithm first learns intermediate rules for hidden neurons. These intermediate rules are linear combinations of the input neurons that activate a specific hidden neuron. Then a linear programming solver is used to learn the final rules (Biswas et al., 2017). The *Fast Extraction of Rules from Neural Networks (FERNN)* algorithm aims at identifying relevant hidden and input neurons. It uses C4.5 to construct a decision tree on top of the ANN. This decision tree is the basis for the rule extraction step(Biswas et al., 2017). The *Continuous Rule Extractor via Decision tree induction (CRED)* algorithm first builds a decision tree for each output neuron by just using the hidden neurons. Then, input neurons are also incorporated into these decision trees. Finally, rules are extracted from these decision trees (Zilke et al., 2016). The *ANNT* algorithm aims at reducing the number of rules that are extracted (Biswas et al., 2017). The *Greedy Rule Generation (GRG)* algorithm is restricted to discrete input features and ANNs consists of only one hidden unit (Biswas et al., 2017). The *Ensemble-Recursive-Rule Extraction (E-Re-RX)* algorithm uses multiple, *e.g.*, two ANNs to extract decision rules (Biswas et al., 2017).

5.1.7 Decompositional Decision Rules from Random Forests (RFs)

There are approaches that use a RF as the underlying prediction model and that solely extract rules from the RF. The *RuleFit* algorithm uses predictive learning to build a linear function on top of the rules extracted from the RF. The linear function consists of rule conditions and input features. The algorithm uses lasso penalty for the coefficients and the final linear combination (Mashayekhi & Gras, 2017).

The *Node harvest algorithm* starts with an initial set of rules and adds additional rules to this set based on two criteria. In order for a rule to be added to the set, a rule has to satisfy a threshold set for the number of features used in the condition of the rule and a threshold set for the number of samples it covers. The algorithm further finds weights for all the selected rules while minimizing a loss function (Mashayekhi & Gras, 2017).

The *hill climbing with downhill moves (DHC)* algorithm removes rules from the set of rules generated by the RF based on a scoring function. The scoring function assigns each rule a score based on specific accuracy measures and complexity measures for interpretability that were defined beforehand (Mashayekhi & Gras, 2017). The *Sparse Group Lasso (SGL)* method first groups rules based on their underlying tree in the RF. It further finds a weight vector over the generated rules from the RF. This weight vector is sparse. Groups are assigned a weight as well. Based on the weights, single rules or whole groups are eliminated from the final decision rules (Mashayekhi & Gras, 2017). The *Multi-class Sparse Group Lasso (MSGL)* method functions in the same way as the SGL method but uses multi-class sparse group lasso instead of sparse group lasso. In contrast to all the previously mentioned decompositional RF methods, this method can only be used for classification tasks but not for regression tasks (Mashayekhi & Gras, 2017).

5.1.8 Pedagogical Decision Trees

TREPAN uses a hill climbing search process for its tree construction. It further uses a gain ratio criterion to find the best M-of-N splits for each node. Whenever there are not enough training instances available for a certain split, additional data instances for training are generated. However, TREPAN is limited to binary classification (Craven & Shavlik, 1996).

The *ANN-DT* algorithm was initially proposed for tree induction from ANN. However, binary decision trees can be extracted from any block box using ANN-DT, which is an algorithm similar to CART. It uses a sampling strategy to create artificial training data that receives its labels from the underlying black box (Ninama, 2013).

DecText was originally proposed for ANNs but it can be applied to any black box model. The method uses a novel decision tree splitting criterion for building the decision tree (Ninama, 2013).

5.1.9 Decompositional Decision Trees

Barakat and Diedrich (2004) use the C4.5 algorithm and apply it to SVMs in order to extract decision trees from SVMs. *X-TREPAN* (Biswas et al., 2017) is an extension of the TREPAN algorithm in order to extract decision trees from ANNs.

5.1.10 Pedagogical Decision Tables

The *KDRuleEX* (Biswas et al., 2017) method uses a genetic algorithm to create new training data when not enough data instances are given for a certain split. The final output is a decision table.

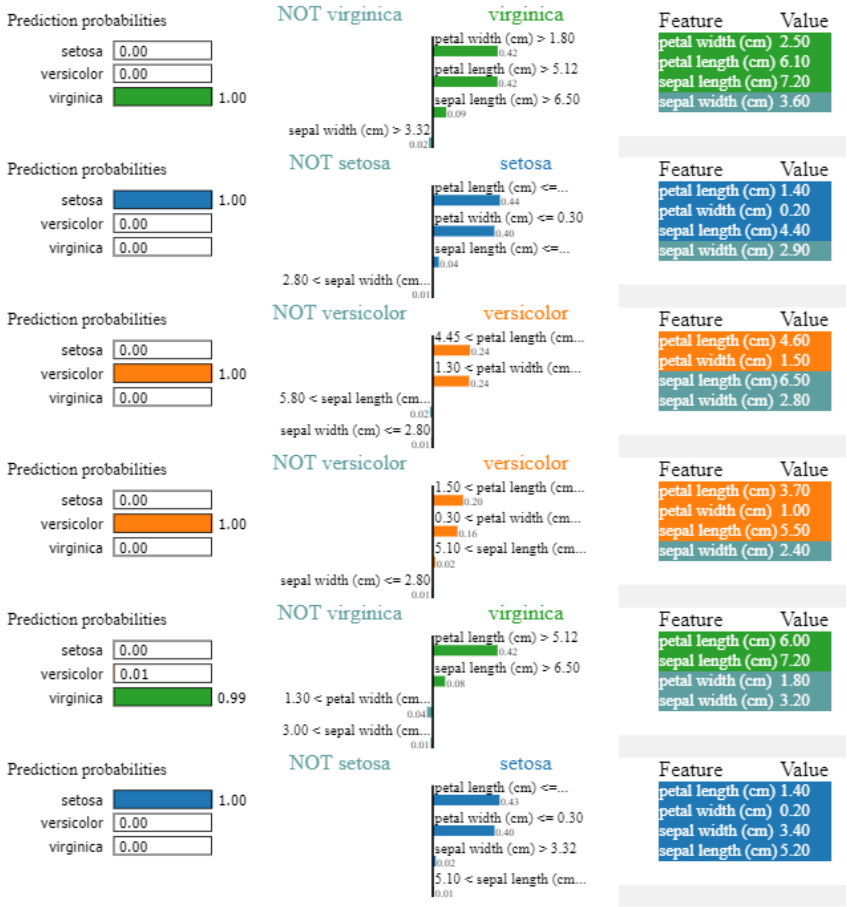


Figure 8: Six explanations for the three classes (virginica, setosa and versicolor) that were picked by sp-LIME. On the right side, the features and the feature's values are displayed.

5.1.11 The Use Case

Submodular-pick LIME is model-agnostic, post-hoc and a global surrogate which expands the LIME method by choosing a few instances that explain the global decision boundaries of the model best. Sp-Lime needs the black box model and a sample of the input data set that represents the entire data space. The approach outputs a single feature contribution overview for each of the chosen instances to be representative for the feature space. Thereby, the user can evaluate each of them separately and deduct from the set of explanations the understanding of the model globally. From the small number of local explanations, the user gets an impression of the decision process of the model for different feature distributions. We chose six explanations of the three classes (see Figure 8). From the instances explained, one can see which features affect which class in which direction (negative or positive), an assertion that holds globally since the task at hand is simple. This explanation seems to be kind of global since it provides only a small sample of the global behaviour depending on how many explanations are chosen. The more explanations are chosen, the more difficult it may be for the user to understand the model.

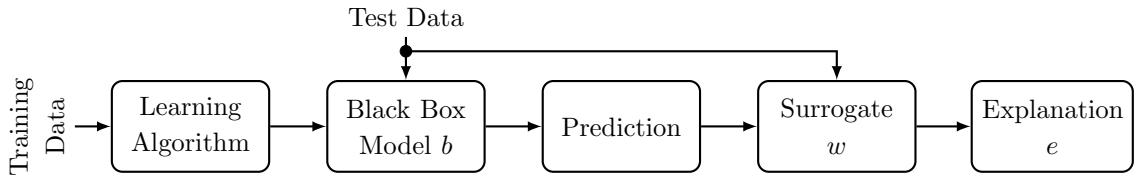


Figure 9: Fitting a local surrogate according to Figure 1(f).

5.2 Local Surrogates

In this section, we focus on approaches that provide a local surrogate model being extracted or being created from a given SML model. In contrast to the global surrogates discussed above, the local surrogate is valid only for a specific data instance and its close vicinity. LIME—one of the currently most popular explainability approaches—belongs to the class local surrogates. The overview of local surrogate models is listed in Table 6.

5.2.1 Decision Rules

Ribeiro et al. (2018) introduce anchorLIME. The approach generates explanations in the form of independent IF-THEN rules like IDS (Phillips et al., 2017). Model Understanding through Subspace Explanations (MUSE) (Lakkaraju et al., 2019) is a model-agnostic framework that provides understanding of a black box model by explaining how it behaves in the sub-spaces defined by the features of interest. The framework learns decision sets for a specific region in the feature space. LOcal Rule-based Explanations (LORE) (Guidotti et al., 2018a) learns a local interpretable model on a synthetic neighborhood generated by a genetic algorithm. It derives a meaningful explanation from the local interpretable predictor which consists of a decision rule set of counterfactual rules.

5.2.2 Linear Models

The explanation frameworks *Model Explanation System* (MES) and *Local Interpretable Model-Agnostic Explanations* (as well as its variations) are local model approximation approaches. MES (Turner, 2016) is a general framework which explains the predictions of a binary black box classifier. MES explains the outcome (predictions) made by a binary classifier. Model Agnostic Supervised Local Explanations (MAPLE) (Plumb et al., 2018) uses a local linear modeling approach with a dual interpretation strategy of random forests. LIME (Ribeiro et al., 2016c) describes a particular prediction made by any black box classifier by taking samples from the locality (neighborhood) of a single prediction. The explanation is valid only on a local level. The samples are presented in the form of an explanation model, *e.g.*, linear model, decision tree, or rule list. *explainVis* interprets the individual predictions as local gradients which are used to identify the contribution of each feature. With *explainVis*, it is possible to compare different prediction methods (Robnik-$\boldsymbol{v}$Sikonja & Kononenko, 2008). Another approach for interpreting predictions is SHAP (SHapley Additive exPlanation). SHAP explains a particular output by computing an interpretation contribution to each input of a prediction. SHAP fulfils the three characteristics *Local accuracy*, *Missingness* and *Consistency*. *Local accuracy* determines the same output for an

Table 6: Overview of local surrogate approaches.

Approach	**Learning Task**	**Model**	**References**
LIME	Classification/Regression	Linear Model	Ribeiro et al. (2016c)
aLime	Classification	Linear Model	Ribeiro et al. (2018)
MES	Classification	Linear Model	Turner (2016)
MUSE	Classification	Linear Model	Lakkaraju et al. (2019)
LORE	Classification	Linear Model	Guidotti et al. (2018a)
MAPLE	Classification	Linear Model	Plumb et al. (2018)
Kernel SHAP	Classification	Linear Model	Lundberg and Lee (2017)
Linear SHAP	Classification/Regression	Linear Model	Lundberg and Lee (2017)

approximated model and the original model if they receive the same input. Missingness relates to the fact of missing features. If a feature is absent, it should not have any impact on the output. *Consistency* ensures that if a feature increases or stays the same, the impact of this feature should not decrease. With the help of SHAP, there is a unique solution in the class of meaningful properties (Lundberg & Lee, 2017).

5.2.3 Use Case

LIME approximates the decision of the black box linearly in a local neighborhood of a given instance. As input LIME requires a black-box model and the instance of interest. Figure 10 illustrates the explanation for a *setosa* instance that is generated by LIME. This overview illustrates for the user which features influenced the likelihood of each of the three possible classes in this specific case. The user can see clearly why this instance was classified the way it was based on the specific feature values.

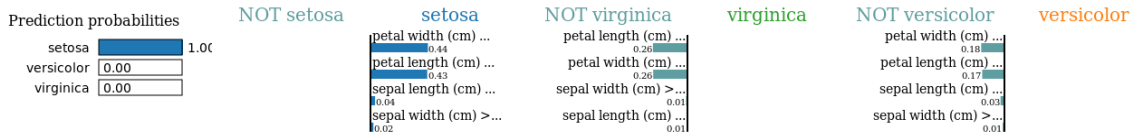


Figure 10: LIME explanation for the *virginica* instance. The instance was classified as *virginica* and the most influential features therefore is that *petal width* is smaller than 0.30 cm

Another approach is SHAP (Kernel SHAP or Linear SHAP) (Lundberg & Lee, 2017). It is a local and post-hoc approach which extracts the feature importance for a given prediction. As input we take the trained model and the desired instance for which we want an explanation. The SHAP approach relies on the fact that a prediction can be written as the sum of bias and single feature contributions. The feature contributions (shapley values) are extracted by marginalizing over every feature to analyze how the model behaves in its absence. This yields the result of an explanation in an overview (see Figure 11).

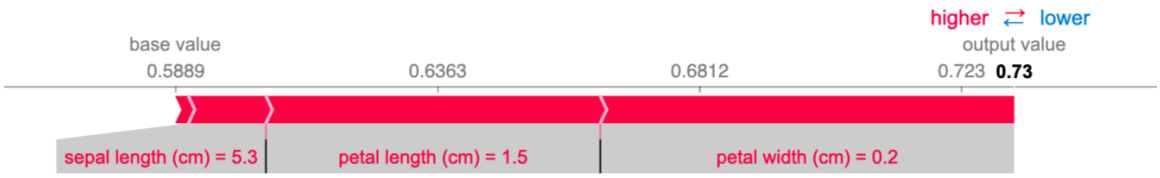


Figure 11: Shapley values plotted for setosa output: The plots illustrate that, starting from a base value of 0.5889 for the probability of class *setosa*, the features *petal length* and *petal width* contributed the most to the final probability of 0.73 (output value). A blue arrow in the other direction would tell us that a feature contributed negatively to the classification probability.

6. Explanation Generation

In this section, we describe approaches that directly can generate an explanation (either local or global). The difference between the surrogate models is that the explanation is directly inferred from the black box model.

Table 7: Overview of global explanation generation methods.

Approach	**Learning Task**	**Input Model**	**Output Model**	**References**
RF Feature Importance	Classification	Model-specific	Tree-based	Hall et al. (2017a)
Sparsity Constraints	-	-	-	Ustun and Rudin (2014)
Correlation Graph	-	Model-agnostic	-	Hall et al. (2017b)
Residual Analysis	-	Model-agnostic	-	Hall et al. (2017b)
Autoencoder	-	Model-agnostic	-	Hall et al. (2017b)
PCA	-	Model-agnostic	-	Hall et al. (2017b)
MDS	-	Model-agnostic	-	Hall et al. (2017b)
t-SNE	-	Model-agnostic	-	Hall et al. (2017b)
Nomograms	Classification	Model-agnostic	Linear Model	Robnik and Kononenko (2008)
SOM	-	Model-agnostic	-	Martens et al. (2008)
Quasi Regression	Classification	Model-agnostic	-	Jiang and Owen (2002)
EXPLAINER global	Classification	Model-agnostic	-	Subianto and Siebes (2007)
GSA	Classification	Model-agnostic	-	Cortez and Embrechts (2011)
GOLDEN EYE	Classification	Model-agnostic	-	Henelius et al. (2014)
GFA	Classification	Model-agnostic	-	Adler et al. (2016)
ASTRID	Classification	Model-agnostic	-	Henelius et al. (2017)
PDP	Classification	Model-agnostic	-	Goldstein et al. (2015)
IME	Classification Regression	Model-agnostic	-	Bohanec et al. (2017)
Monotonicity Constraints	-	-	-	Freitas (2014)
Prospector	Classification	Model-agnostic	-	Krause et al. (2016)
Leave-One-Out	Classification	Model-agnostic	-	Strumbelj et al. (2010)

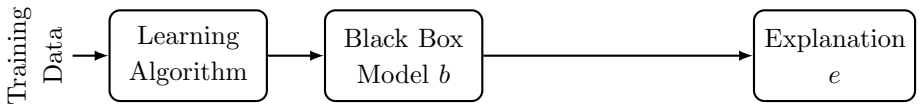


Figure 12: Generating global explanations according to Figure 1(b).

6.1 Global Explanation Generation

In this section, global explanators are reviewed. As depicted in Figure 12, they are independent of certain model predictions and try to reveal certain properties of the black box model. The global explanation generation approaches are listed in Table 7.

6.1.1 Feature Importance

RF Feature Importance (Hall et al., 2017a) is limited to decision trees and ensembles of decision trees. Per split in the decision tree, the information gain is assigned to the splitting feature as its importance measure. This importance measure can be accumulated per feature over all trees. Other feature importance measures for tree-based methods are the feature's depth in the tree or the total number of instances that are used for classification (Freitas, 2014).

Quasi Regression (Jiang & Owen, 2002) can be used to visualize the contributions to a black box function of different subsets of input features. The black box function is expanded in an orthonormal basis with an infinite number of coefficients. These coefficients can be estimated using the Monte Carlo method.

Explainer (global) (Subianto & Siebes, 2007) is an approach that provides global insights for the importance of a feature. Each feature is assigned a weight that reflects the feature's overall influence. However, this approach is restricted to discrete data.

Global Sensitivity Analysis (GSA) (Cortez & Embrechts, 2011) introduces a visualization approach to explain decisions made by black box models that is based on a sensitivity analysis method. It measures the effects on the output when the input features are varied through their range of values. GSA can consider multiple features at a time. Sensitivity measures used are range, gradient, and variance. Feature importances are plotted in a bar plot and the contribution of a given input feature or a pair of input features is plotted in a Variable Effect Characteristic (VEC) Plot or a VEC Surface and Contour Plot.

The *Golden Eye* (Henelius et al., 2014) method first finds groupings of features that indicate important feature interactions and associations between features that are exploited by the underlying classifier. This is done without considering the internal structure of the classifier or the distribution of the input data. The classifier is considered a black box. Further Golden Eye uses randomization of input features to figure out feature importance.

The *Gradient Feature Auditing (GFA)* approach uses an obscuring technique to find indirect influence of the input features without retraining the underlying black box model. It outputs a plot of features and the feature's influences as well as a feature ranking based on their influence on the target (Adler et al., 2016).

Automatic Structure Identification (ASTRID) finds groupings of feature interactions that describe the underlying data distribution. It is a top-down greedy algorithm based

on statistical significance testing to find the maximum cardinality grouping. These groupings reveal associations between features and feature interactions used by the underlying classifier. Features in the same group interact; features in different groups do not interact (Henelius et al., 2017).

Partial Dependence Plot (PDP)s display the average prediction of the black box when an individual feature is varied over its range. The underlying idea of partial dependence aims at showing how an individual feature affects the prediction of the global model. An individual feature's relationship with the target is visualized in PDPs (Goldstein et al., 2015).

Prospector uses an extension of PDP to visualize how features affect a prediction (Krause et al., 2016). PDP are extended by a partial dependence bar that shows a colored representation of the prediction value over the range of input values that a certain feature can take. Prospector also uses a novel feature importance metric that outputs a feature's importance number on an instance basis. Furthermore, Prospector provides actionable insight by supporting tweaking of feature values. However, it is restricted to single class predictions and also limited insofar as it does not take interactions between features into account.

The *Interaction-based Method for Explanations (IME)* method aims to find the feature importance while also considering feature interactions. Interactions are considered by randomly permuting some feature values and measuring the prediction difference. IME further uses the Shapley value from coalitional game theory to assign each feature a contribution value (Bohanec et al., 2017).

The *Leave-One-Out* approach learns a feature's contribution by omitting the feature from the prediction process. It considers all different subsets of features which enables the consideration of feature interactions. Each feature is assigned a local contribution value that can be aggregated and averaged to also assign each feature a global contribution value. However, the approach only works with low dimensional data or when a feature selection process is applied beforehand (Strumbelj et al., 2010).

Strumbelj and Kononenko (2014) describe a general approach for explaining how features contribute to classification and regression models' predictions by computing the situational importance of features (local or global).

iForest (Zhao et al., 2019) is an interactive visualization system that helps users interpret random forests model by revealing relations between input features and output predictions, hence enabling users to flexibly tweak feature values to monitor prediction changes. *RuleMatrix* is an interactive visualization technique that supports users with restricted expertise in machine learning to understand and validate black box classifiers. *Aggregate Valuation of Antecedents (AVA)* (Bhatt et al., 2019) is a value attribution technique that provides local explanations but also detects global patterns.

6.1.2 The Use Case

PDP is a global, model-agnostic and post-hoc explanation generation approach. PDPs can be implemented on all trained models, more or less efficiently. Additionally, we need the data set to marginalize over the features we are not interested in. PDPs are presented as plots in which we either plot a classification against its partial dependence on one feature (see Figure 13) or two features against each other with a map plot (see Figure 14). From a

Table 8: Global Feature Importance.

Feature	Importance
sepal length (cm)	0.47174951
sepal width (cm)	0.40272703
petal length (cm)	0.10099771
petal width (cm)	0.02452574

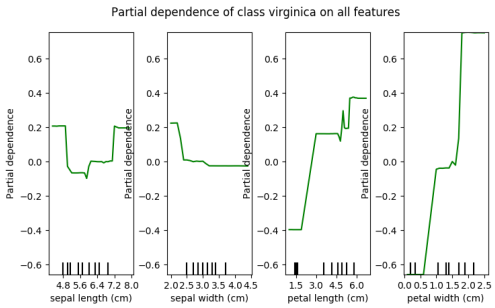


Figure 13: Explanations per feature.

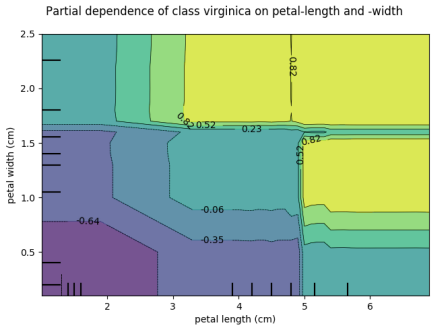


Figure 14: Explanation against two features.

PDP, the correlation a feature has to the classification of a certain class can be extracted. The map plot illustrates the virginica probability and the interaction of petal width and petal length. The plot shows the increase in virginica probability if petal length is greater than 3 cm and petal length is greater than 1.8 cm.

Another approach in this category is the Global Feature Importance. It is a model-agnostic, post-hoc explanation generation approach similar to the treeinterpreter (Saabas, 2015), but global. Global Feature Importance simply returns a list assigning an importance score to each feature that, when taken together, sum up to 1. This importance does not tell the user much about the effect of the features but merely whether or not the model considers them strongly. This is a first good entry for explainability but it needs to be combined with other explainability approaches to gain more detailed insights.

Table 8 illustrates that the feature importance provides that *sepal-length* and *-width* are the most decisive features which the model considers more often than the other two.

6.2 Local Explanation Generation

Local explanations are only valid in the vicinity of a certain prediction as indicated in Figure 15. This section reviews commonly used local explanators. The local explanation generation approaches are listed in Table 9.

6.2.1 Saliency Methods

Saliency methods in general relate the model prediction to the feature vector by ranking the explanatory power, i.e. the salience of the individual features.

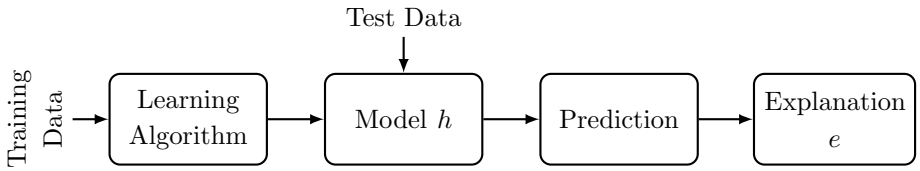


Figure 15: Generating local explanations according to Figure 1(e).

Feature Attribution: An important class of saliency methods are feature attribution approaches that explicitly try to quantify the contribution of each individual feature to the prediction.

Explainer (local) provides local insight by finding a set of features that is minimal in the sense that changing the values of those features changes the prediction outcome (Subianto & Siebes, 2007). This concept can be formalized and used as the basis for finding minimal sets of features. However, this approach is restricted to discrete data.

The *Local Gradients* approach learns a local explanation vector consisting of local gradients of the probability function. The local explanation vector can be mapped onto the input feature vector such that the sign of the values in the local explanation vector indicates the direction of influence of the underlying feature towards the instance and the absolute value indicates the amount of influence. When this gradient information cannot be obtained from the used prediction model, a probabilistic approximate such as parzen window, logistic regression or Gaussian Process Classification is employed (Baehrens et al., 2010).

The *Leave-One-Covariate-Out (LOCO)* approach trains two different models, one with all input features and one with the feature of interest left out. These two models can then be compared with each other using their computed residuals. If their difference is above a certain threshold for some values of the feature, the feature is considered important in that range. This approach can be applied to the entire data (global) or just one instance (local) (Lei et al., 2018; Hall et al., 2017a).

The *Quantitative Input Influence (QII)* (Datta et al., 2016) method introduces different measures for reporting the influence of input features on the prediction outcome. In order to do so, feature values are randomly perturbed and their change in the prediction outcome is measured. In order to consider correlations between input features, the causal QII measure called Unary QII is used.

Model Class Reliance (MCR) derives connections between permutation importance estimates for a single prediction model, U-statistics, conditional causal effects, and linear model coefficients. Furthermore, they give probabilistic bounds for MCR by using a generalizable technique (Fisher et al., 2018).

Individual Conditional Expectation (ICE) plots are an extension of PDPs. They visualize the relationship between the target and an individual feature on an instance basis and not for the whole model. When ICE plots and PDPs are plotted in the same graph, the difference between the individual behavior of a feature and its average behavior can be revealed. Two further extensions of normal ICE plots, centered and derivative ICE plots, can discover heterogeneity and explore the presence of interacting effects, respectively (Goldstein et al., 2015). The treeinterpreter (Hall et al., 2017b) can only be applied to decision trees. It

breaks down the final prediction for a specific instance into bias and feature contribution and augments the decision tree's nodes as well as the paths with this additional information.

Shapley values (Shapley, 1951) are created by means of a method from coalitional game theory assuming that each feature value of the instance is a player in a game where the prediction is the payoff. Shapley values illustrate how to dispense the payoff among the features.

Self Explaining Neural Networks (SENN) contain the interpretation functionality in their architecture. These models fulfill three characteristics for interpretability: explicitness, faithfulness, and stability. *Explicitness* deals with the question of how comprehensible the explanations are. *Faithfulness* checks whether the meaningful features predicted by the model are really meaningful. *Stability* ensures the coherency of explanations for similar input data (Melis & Jaakkola, 2018). The *Visual Explanation Model (VEM)* from Hendricks et al. (2016) describes an image and provides information why this image got classified in this way. Therefore, the model includes a *relevance loss* and a *discriminative loss*. The *relevance loss* is used to generate a description of an image based on visual components. The *discriminative loss* uses reinforcement paradigms to focus on category specific properties. The *Image Captioning Model (ICM)* is an approach which generates a discriminative description for a target image. To generate the description, the model explains why the visual target belongs to a certain category. Afterwards, the model compares the target image with a similar distractor image and attempts to describe the differences. The differences are included in the description to make it more precise (Vedantam et al., 2017).

In case of image data being processed by neural networks, a common approach is to highlight the contribution of individual pixels for a prediction by means of a heat map. For instance, *Layerwise Relevance Propagation* computes scores for pixels and regions in images by backpropagation. Another approach is *Grad-CAM*. This approach can be applied to any CNN model to produce a heat map that highlights the parts of the image that are important for predicting the class of interest. The algorithm further focuses on being class-discriminative and having a high resolution (Selvaraju et al., 2016).

DeepLIFT (Deep Learning Important FeaTures) is an approach where the prediction of a black box model is backpropagated to the input feature in order to generate a reference activation. By comparing the reference activation with the predicted activation of the neural network, DeepLIFT assigns scores according to their difference (Shrikumar et al., 2016). *SmoothGrad* uses copies of a target image, adds noise to those copies and creates sensitivity maps. The average of those sensitivity maps is used to explain the classification process for the target image (Smilkov et al., 2017). *Integrated Gradients* (IG) uses interior gradients of deep networks to capture the importance of the different input values (Sundararajan et al., 2016). Montavon et al. (2017) introduces *Deep Taylor*, an explanation approach for non-linear models. In this approach, each feature is visually analyzed in heat maps which clearly show which input features led to the classification of the output. *LRP* uses the prediction of a model and applies redistribution rules to assign a relevance score to each feature (Samek et al., 2017). In his article, Samek et al. (2017) also explains the output of black box models with the Sensitivity Analysis (SA) approach. Here, the output of a model is explained with the model's gradients. The constructed heat map visualizes which features need to be changed in order to increase the classification score.

Table 9: Overview of local explanation generation methods.

Approach	Learning Task	Input Model	Model	References
ICE Plots	Classification	Model-agnostic	-	Goldstein et al. (2015)
EXPLAINER local	Classification	Model-agnostic	-	Subianto and Siebes (2007)
Border Classification	Classification	Model-specific	Non-linear Model	Barbella et al. (2009)
Cloaking	Classification	Model-agnostic	-	Chen et al. (2015)
Tweaking Recommendation	Classification	Model-specific	Non-linear Model	Tolomei et al. (2017)
Nearest Neighbor	Classification	Model-agnostic	-	Freitas (2014)
SVM Recommendations	Classification	Model-specific	Non-linear Model	Barbella et al. (2009)
PVM	Classification	Model-agnostic	-	Bien and Tibshirani (2011)
Bayesian Case Model	Classification	Model-agnostic	-	Kim et al. (2014)
MMD-critic	Classification	Model-agnostic	-	Kim et al. (2016)
Influence Functions	Classification	Model-agnostic	-	Koh and Liang (2017)
Local Gradients	Classification	Model-agnostic	-	Baehrens et al. (2010)
LOCO	Classification	Model-agnostic	-	Lei et al. (2018)
QII	Classification	Model-agnostic	-	Datta et al. (2016)
SENN	Classification	-	Linear Model	Melis and Jaakkola (2018)
VEM	Classification	Model-specific	Non-linear Model	Hendricks et al. (2016)
Treeinterpreter	Classification Regression	Model-specific	Tree-based	Hall et al. (2017b)
ICM	Classification	Model-specific	Non-linear Model	Vedantam et al. (2017)
DeepLIFT	Classification	Model-specific	Non-linear Model	Shrikumar et al. (2016)
SmoothGrad	Classification	Model-specific	Non-linear Model	Smilkov et al. (2017)
Interior Gradients	Classification	Model-specific	Non-linear Model	Sundararajan et al. (2016)
explainVis	Classification	Model-specific	Linear Model	Robnik-vSikonja and Kononenko (2008)
Deep Taylor	Classification	Model-specific	Non-linear Model	Montavon et al. (2017)
LRP	Classification	Model-specific	Non-linear Model	Samek et al. (2017)
SA	Classification	Model-specific	Non-linear Model	Samek et al. (2017)
Grad-CAM	Classification	Model-specific	Non-linear Model	Selvaraju et al. (2016)

Attention Based Models: Attention based models are used to highlight the most promising parts of input features that lead to a certain output for a given task. Therefore, they make use of context vectors. A context vector is generated by mapping input values with an annotation sequence that contains promising information about preceding and following input features (Bahdanau et al., 2014). However, Jain and Wallace (2019) believe that attention based models do not provide meaningful explanations for model prediction. For a given output, they tried to find out whether the input features with high attention weights were responsible for the predicted outcome. Their work shows that there exists only a weak correlation between feature importance measures and learned attention weights. Wiegreffe and Pinter (2019) question the assumptions in Jain and Wallace (2019)'s work and believe that it depends on how you define an explanation. In various experiments, they show that attention approaches can be used to make model predictions interpretable and explainable.

The *Attention Based Summarization (ABS)* system summarizes a certain input text and captures the meaningful paragraphs. This data-driven approach combines a neural language model with an encoder (Rush et al., 2015). *Image Attention Based Models (IABMs)* describe the content of an image. Furthermore, these networks use additional layers for image interpretation. These interpretations illustrate how the description of the image was created (Xu et al., 2015a). The Encoder-Generator Framework extracts justifications for a specific decision from an input text. The generator selects possible justifications and the encoder maps these justifications to the target values (Lei et al., 2016). *Pointing and Justification-based Explanation (PJ-X)* is a multi-modal explanation approach for visual decision tasks.

It provides convincing explanations for an image by simultaneously highlighting the relevant parts and generating a textual justification (Park et al., 2016). Luong et al. (2015) introduces two attention based models for translation tasks: The *global* and the *local* approach. The *global* attention mechanism takes all available words into account whereas the *local* attention approach works with a small window of words for its next prediction.

Table 10: Overview of attention based models.

Approach	**Learning Task**	**Input Model**	**Model**	**References**
ABS	-	Model-specific	-	Rush et al. (2015)
iABM	-	Model-specific	-	Xu et al. (2015a)
Encoder-Generator	-	Model-agnostic	-	Lei et al. (2016)
PJ-X	Classification	Model-specific	Non-linear Model	Park et al. (2016)
Global Attention Approach	Classification	Model-specific	Non-linear Model	Luong et al. (2015)
Local Attention Approach	Classification	Model-specific	Non-linear Model	Luong et al. (2015)

6.2.2 Counterfactual Methods

Counterfactuals are defined by the Cambridge Dictionary as: "thinking about what did not happen but could have happened" (Cambridge, 2020). They can be expressed in a more formal way as follows: If $\mathbf{x}$ had been $\mathbf{x}'$, Y would have been y'. For example: If the weather had been sunny instead of rainy, I would have gone for a walk instead of staying at home. A counterfactual describes an altered reality in which other facts would have lead to different results. The factual $\mathbf{x}$ has the consequence y. However, if $\mathbf{x}$ changes to the counterfactual $\mathbf{x}'$, the consequence changes to y'. Counterfactuals can be used as a special form of explanation for machine learning systems. In this case, $\mathbf{x}$ and $\mathbf{x}'$ are inputs for the machine learning model and y or y' are outputs or predictions made by the model. Thus the problem of finding a counterfactual explanation turns into a search problem in the feature space with the goal of finding, *e.g.*, a nearby instance that leads to the prediction of a different class. A found counterfactual can be presented either by itself or as the difference from its factual to highlight the changes responsible for the difference in classification. Wachter et al. (2018) describe counterfactual methods as meaningful data subjects that lead to a specific decision. Furthermore, they provide reasons to challenge this output and also advices as to how to receive the desired decision (Wachter et al., 2018). Counterfactual methods are very similar to inverse classification methods being discussed in Section 6.2.3.

The simplest method for counterfactual explanations is the search by trial and error. This approach randomly changes the feature values of data instances and stops when the desired output gets classified (Molnar, 2018). Wachter et al. (2018) generates counterfactual explanations by searching counterfactual data points as close as possible to the original data points so that a new target is chosen. The distance can be measured with the Manhattan-distance which is weighted by the inverse median absolute deviation. Counterfactual instances described by Looveren and Klaise (2019) use a simple loss function and sparsely follow the explanation method of Wachter et al. (2018). A list of various counterfactual methods is to be found separately in Table 11.

Table 11: Overview of counterfactual methods.

Approach	Learning Task	Input Model	Model	References
Trial and Error	Classificiation	Model-agnostic	-	Molnar (2018)
Counterfactual Generation	Classificiation	Model-agnostic	-	Wachter et al. (2018)
Counterfactual Instances	Classificiation	Model-agnostic	-	Looveren and Klaise (2019)
Class Prototypes	Classificiation	Model-agnostic	-	Looveren and Klaise (2019)
LORE	Classification	Model-agnostic	-	Guidotti et al. (2018a)
FACE	Classification	Model-agnostic	-	Poyiadzi et al. (2020)
Coherent Counterfactuals	Classification/Regression	Model-agnostic	-	Russell (2019)
SEDC	Classificiation	Model-agnostic	Linear/Non-linear Model	Martens and Provost (2014)
Growing Sphere	Classificiation	Model-agnostic	-	Laugel et al. (2017, 2018)
Feature Tweeking	Classificiation	Model-specific	Linear Model	Tolomei et al. (2017)
OAE	Classificiation	Model-specific	Linear Model	Cui et al. (2015)

The problem of counterfactual instances is that they suffer from a low degree of interpretability. That is why the *Counterfactual Prototype* approach adds a loss term to the objective function which results in a less sparse but more interpretable output (Looveren & Klaise, 2019).

SEDC is a document-classification method which analyzes the data quality and the deficiencies of a model. It provides an improved understanding of the inner workings of a classifier which leads to better model performance and decision making (Martens & Provost, 2014).

The *Growing Sphere* provides post-hoc explanations for a data instance through the comparison of an output with its closest enemy. Therefore, Growing Sphere gains useful information about the relevant features and illustrates which concepts the classifier has learned so far (Laugel et al., 2017).

Local Surrogate (Laugel et al., 2018) consists of selecting an instance x_{border} which is close to the nearest black box decision border. The instance x_{border} gives important information about the spatial location of the black box decision border for the instance x.

The *Feature Tweeking Algorithm* takes the trained tree-based ensembled model, a true negative feature vector, a cost function, and a threshold as key input components. The cost function measures the transformation process from a truly negative instance to a positive one. The positive threshold is responsible for the fine-tuning so that every single feature follows the correct path of each tree (Tolomei et al., 2017).

The *optimal action extraction* (OAE) can be used for random forest classifiers, adaboost and gradient boosted trees. This approach attempts to find a feature vector so that the desired output is achieved at a minimum cost (Cui et al., 2015).

The *Feasible and Actionable Counterfactual Explanations* (FACE) approach aims to build coherent and feasible counterfactuals by using the shortest path distances defined via density-weighted metrics (Poyiadzi et al., 2020).

Russell (2019) proposes coherent counterfactual explanations for mixed data sets and proposes a concrete method for generating diverse counterfactuals based upon mixed integer programming.

6.2.3 Inverse Classification

Inverse classification looks at an individual instance and determines the values of that instance that need to be adjusted in order to change the instance's class (Barbella et al., 2009). This category supports the generation of counterfactual explanations.

Cloaking is an approach that finds features that can be hidden from the prediction model in order to decrease the probability of belonging to a certain class. These features are called evidence counterfactual. When removed they significantly reduce the probability of an instance belonging to a certain class (Chen et al., 2015).

The *inverse classification framework* presented by Lash et al. (2017) can be applied to different classification models. This framework provides meaningful information about adapting the input values in order to change the actual output class. In addition, the framework ensures that these proposed data modifications are realistic.

Border Classification reports an instance's features that need to be changed in order to place that instance on the border (separating surface) between two classes. This approach is restricted to SVMs since it makes use of the support vectors (Barbella et al., 2009). Based on Barbella et al. (2009)'s SVM classification, these recommendations report an instance's most influential support vector. The method utilizes a pull measure that uses the kernel function similarity to measure the contribution of a support vector towards the prediction of an unseen instance (Barbella et al., 2009). This approach can also be categorized as a *Prototype* which will be presented in the next section.

6.2.4 Prototypes and Criticism

Nearest neighbor classifiers can be regarded as a method for providing prototypes. They do not explicitly build a model from the training data but instead use a similarity measure to figure out the nearest neighbors of the unseen data instance. These nearest neighbors vote with their own label for the label of the unseen instance. Hence, an explanation for one unseen instance differs from the explanation of another unseen instance. To avoid different explanations depending on the instance to be classified, typical instances can be found and used as prototypes. Another disadvantage of nearest neighbor classifier arises with high dimensional data. The concept of neighborhood has to be reconsidered when the data has a large number of features. One can either only regard certain feature-neighborhoods instead of the whole feature space or incorporate feature weights into the classifier (Freitas, 2014).

Other approaches improve on the drawbacks of simple nearest neighbor classifiers. The *Prototype Vector Machines (PVMs)* (Bien & Tibshirani, 2011) finds a small set of prototypes that well represent the underlying data. The prototypes are found using an integer program that is approximated. These prototypes are selected to capture the full variance of the corresponding class while also discriminating from other classes. These prototypes can further be used for classification. An unseen data instance can be labeled based on the closest prototypes (Bien & Tibshirani, 2009).

Another approach involves selecting prototypes that maximize the coverage within the class, but minimize the coverage across them. The *Bayesian Case Model* (Kim et al., 2014) is an unsupervised clustering technique that learns prototypes and subspaces per cluster. Subspaces contain those features that characterize a cluster and are important to the corresponding prototype.

Table 12: Feature Importance for a certain instance.

Feature	Importance
bias	0.35685
petal length (cm)	0.28069664
petal width (cm)	0.26613121
sepal length (cm)	0.08566027
sepal width (cm)	0.00684522

The *MMD-critic* approach uses prototypes together with criticism in order to put the focus on the aspects that are not captured by the model. The approach uses Bayesian model criticism together with maximum mean discrepancy (MMD) to select prototypes. Criticism samples are scored with a regularized witness function and then selected (Kim et al., 2016).

ProtoDash (Gurumoorthy et al., 2017) is an approach for selecting prototypical examples from complex data sets that extends the work from Kim et al. (2016) by non-negative weights for the importance of each prototype.

The *Influence Function* (Koh & Liang, 2017) method approximates leaving out one training data instance and retraining the prediction model with influence functions. The method reports those training instances that were most influential for a specific instance and its prediction.

6.2.5 The Use Case

The treeinterpreter approach is model-specific to decision trees and random forests. It is a local, post-hoc inspection approach that retrieves feature contributions to a final prediction in a manner very similar to the Shapley values. Here too, as input we need the trained model and the desired instance for which we want an explanation.

In case of a *setosa* sample, this method yields the following contributions which sum up to 1 because the random forest used predicts the setosa class with complete certainty. In this particular case, the result of the treeinterpreter highlights that both *petal length* and *petal width* had a positive impact on the decision, while *sepal length* and *sepal width* were almost not considered. The bias of a little more than one-third illustrates that the model is slightly biased towards the *setosa class* compared to the other classes (see Table 12).

7. Data and Explainability

This chapter focuses on the topic of data and explainability. First, we highlight the topic of data quality which is an essential factor for explainability. Furthermore, we describe the topic of ontologies in detail. Ontologies can improve the explainability of any given model by incorporating knowledge either before the model training or after the explanation generation to further improve them.

7.1 Data Quality

A survey conducted by Kaggle (Kaggle, 2017) revealed that the most significant barrier data scientists face is poor data. The quality of the underlying data is essential. If a huge

amount of incomplete and noisy data is used to train a model, the results will be poor. Data in the real world is always incomplete, noisy and inconsistent. For example, if you need to prevent data quality issues in a Hospital Information System (HIS) where the data is added manually by the hospital personnel, it is mandatory to have the uncleaned data to learn from mistakes the personnel made in order to prevent these mistakes from happening in the future. In fields in which persons are being classified to a certain class, *e.g.*, credit scoring or medical treatment, it is mandatory to have an accurate model that hasn't been trained by inconsistent data. It is a commonly held belief that the more data is available, the better the model is. For example, Google's Research Director Peter Norvig (Cleland, 2011) claimed: "We don't have better algorithms than anyone else; we just have more data." While this isn't wrong altogether, it obscures the crucial point—which is that it isn't enough to have more data but to have more good data (Amatriain, 2017). Several experiments were conducted where more data was included in a training step and the results showed that this did not improve the performance of the model (Amatriain, 2017).

Therefore, Gudivada et al. (2017) quite rightly states: "The consequences of bad data may range from significant to catastrophic". In 1996, Wang and Strong (1996) introduced a typical definition for data quality. They defined data quality as the degree to which the available data meets the needs of the user. As Helfert and Ge (2016) have derived from this definition, the concept of data quality is context-dependent and subjective. There is a variety of data quality dimensions, the six core dimensions of which are Completeness, Uniqueness, Timeliness, Validity, Accuracy, and Consistency. Completeness is the proportion of data actually collected compared to the full amount of data that theoretically could have been collected. Uniqueness measures the degree of identical data instances. Timeliness is responsible for ensuring that the collected data is not outdated. Validity checks whether the syntax of the data matches its definition. Accuracy measures whether the available data describes the given task correctly. Consistency compares the same data representations with their definitions and measures the differences (Askham et al., 2013). In the field of machine learning, a high quality of data is essential for the prediction of a correct model result. Here, the assessment of data quality becomes more complex because different performance metrics, the search for the best parameters or the model type can distort the data quality (Gudivada et al., 2017). Most derived explanations are based on the outcome of a model and depend on the data that the model uses for its predictions. If the quality of the data is low, the prediction could already be wrong and thus, the explanation could be correct but based on inconsistent data.

7.2 Data Visualization

Data visualization approaches are useful in order to get a first impression of the data. The goal is to visualize the entire data in just two or three dimensions. Visualization techniques can be used for exploratory data analysis and as a complement to a prediction model. Some of the available approaches are listed in Table 13 and are not linked to a specific learning task. Figure 16 illustrates an example of some of the described visualization approaches.

Nomograms can be used to visualize linear SVMs, logistic regression, and Naive Bayes (NB). They visualize the associated weights for each feature (Robnik & Kononenko, 2008). self-organizing map (SOM) are two-layer ANNs that preserve the topology of the underlying

Table 13: Overview of visualization approaches.

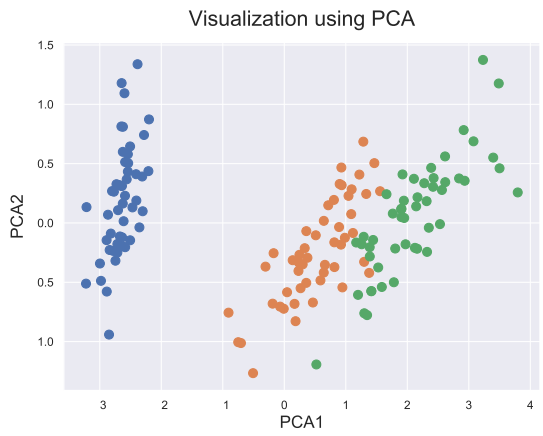
Approach	Input Model	Model	References
SOM	Model-agnostic	-	Martens et al. (2008)
Nomogram	Model-agnostic	Linear Model	Robnik and Kononenko (2008)
Glyphs	Model-agnostic	-	Hall et al. (2017b)
Correlation Graphs	Model-agnostic	-	Hall et al. (2017b)
Autoencoder	Model-agnostic	-	Kramer (1991)
PCA	Model-agnostic	Linear/Non-linear Model	Wold et al. (1987)
MDS	Model-agnostic	-	Mead (1992)
t-SNE	Model-specific	Linear Model	van der Maaten and Hinton (2008)

data. Similar instances are mapped closely together in a SOM. Furthermore, color is assigned to the trained neurons based on their respective classification (Martens et al., 2008). Glyphs represent rows of a data set using color, texture, and alignment (Hall et al., 2017b). Correlation graphs visualize the relationships between input features (Hall et al., 2017b). Residual values are plotted against the predicted values. A random distribution of plotted points implies a good fit of the underlying prediction model (Hall et al., 2017b). An auto-encoder uses an ANN to learn a representation of the data using fewer dimensions than the original data. These dimension can then be visualized in a scatter plot (Hall et al., 2017b). Principal Component Analysis (PCA) extracts the principle components from the data and visualizes these components using a scatter plot. The goal is to find linear combinations of input features that represent the underlying structure of the data while reducing the overall number of features (Hall et al., 2017b). Scatter plots are a good tool for this visualization since they are able to visualize key structural elements such as clusters, hierarchy, outliers, and sparsity. They further project similar aspects close to one another. Multi-Dimensional Scaling (MDS) is a linear projection method that is used to map data on approximate Euclidean metric space and visualize it using a scatter plot (Hall et al., 2017b). The t-SNE visualization technique is a non-linear dimensionality reduction technique that maps data in a low dimensional space while preserving the data's structure. It finds two distributions in high and in low dimensional space and minimizes a metric between them. The high dimensional space can be converted into a matrix with pairwise similarities (Hall et al., 2017b).

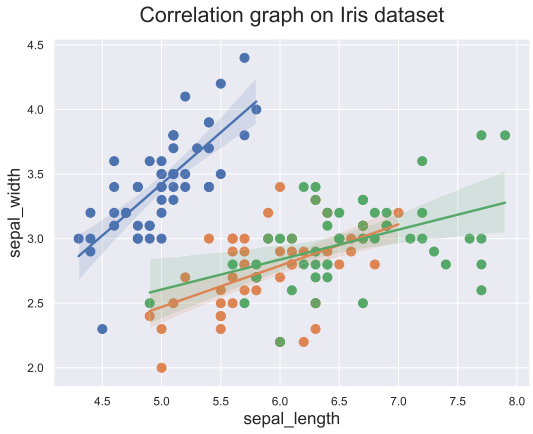
Other visualization techniques for visualizing feature importance and relationship were already mentioned. They include PDP, ICE plots, explainVis using bar plots, VEC plots, VEC surface and contour plots, and heat maps. Figure 13 illustrates some of the described visualization approaches.

7.3 Ontologies

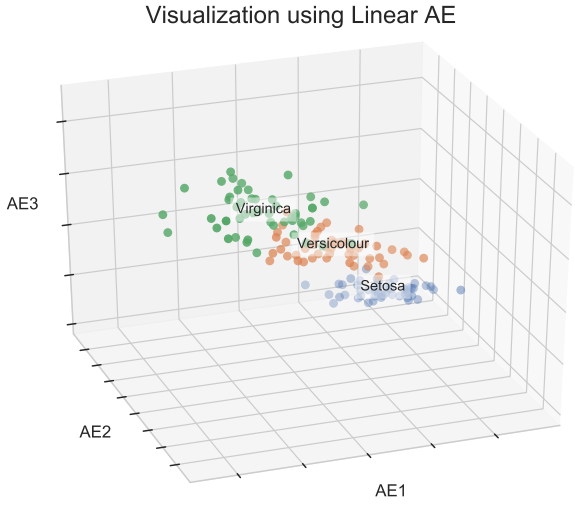
The application of ontologies can be used to improve the data quality but also to generate better explanations. By including domain knowledge that is approved by experts , *e.g.*, consistency checks can be done directly on the data. This can improve the classification performance of the model. Other ontologies can be used to improve the explainability by



(a) PCA (2D)



(b) Correlation graph (2D)



(c) Linear AE (3D)

Figure 16: Data visualization techniques.

including them before training the model , *e.g.*, by summarizing features to facilitate the understanding for users. In what follows, we will give an overview of the topic of ontologies.

7.3.1 Definition of Ontology

Knowledge combines information, experience and skills to enable a concept of understanding. The basis for that process is data which is currently growing faster and faster. All this data is used to acquire different forms of information and to learn about important connections within the collected data. In the end, this knowledge can be used to make possible new advances and discoveries. But in order to fulfill this purpose, knowledge needs to be shared.

Acquired knowledge is important in many fields and needs to be spread accordingly. An easy exchange needs a uniform structure and, thus, a common definition. This is even more important if knowledge is to be shared between computational systems like AIs. One approach to enable this exchange of data and knowledge are ontologies. Gruber et al. (1993) first defined an *Ontology* as a specification of a conceptualization in the context of knowledge sharing. The definition includes essential parts that till this day are the main building blocks of ontologies. An ontology is a vocabulary that represents the categories, properties, and relations of a specific domain in a formal and organized structure. It consists of classes, properties, and individuals which are utilized to define concepts.

Definitions are written in a standardized language with the Web Ontology Language (OWL) (W3C, 2012b) as a collection of languages used to author ontologies. It builds upon the Resource Description Framework (RDF) (W3C, 2014) that was initially conceptualized by the World Wide Web Consortium (W3C) for describing meta data and that is now a center piece of the Semantic Web. This concept defines statements in the form of 3-tuples, consisting of subjects, predicates, and objects which can be represented in a labeled, directed multi-graph. The knowledge representation inside an ontology follows a description logic which utilizes concepts, roles, individuals, and operators. It is part of first order logic (predicate logic) and able to define formal descriptions of logical contexts. A concept represents a unary predicate, whereas a role represents a binary predicate and an individual is a constant. An operator can combine concepts and roles to form new definitions.

In summary, an ontology describes knowledge as formal definitions of the types, properties, and relations that exist in some domain. A uniform structure enables the exchange of knowledge between various fields of science. Ontologies can be combined to create a greater knowledge out of smaller concepts, where upper ontologies build the basis for bigger compounds (Hoehndorf, 2010). The Descriptive Logic enables the use of Semantic Reasoners which infer unstated information from the ontology and check the consistency of that knowledge base.

Representing knowledge in this form of ontologies can have many advantages such as the inference of information from relational concepts. But since different needs require different services, there are other approaches to define relational knowledge.

Taxonomies are mostly used to demonstrate origins and connections of words. They show relations of usage and their counterparts which makes them very useful for structuring information. Therefore, many dictionaries are represented in the form of a taxonomy.

Table 14: Often used ontologies and their use cases.

Approach	**Use Case**	**Reference**
FOAF	Social media	Brickley and Miller (2020)
SIOC	Social media	Bojars and Breslin (2020)
Music Ontology	Music industry	Raimond et al. (2020)
GoodRelations	Online shopping	Hepp (2020)

Even ontologies often use taxonomies as a building block to make their knowledge more structured.

Furthermore, ontologies themselves can be used as building blocks to construct even greater knowledge relations. Those concepts known as *knowledge graphs* are high level structures of connected information. Using them to interlink sources of similar topics makes them a great framework for search engines. The largest knowledge graph today is produced by Google and displays additional and related information in a separate area on the side when searching with Google's search engine. It enables quick access to corresponding knowledge by linking similar information and commonly searched additions.

7.3.2 Ontologies in Practice

Besides sharing knowledge in a uniform way across many areas, ontologies are primarily used for information mining. The focus does not lie on the retrieval of raw data but on the cognition of possible connections within a domain. In the pharmacy industry, searching for the causes of an illness can be done by categorizing identified explicit relationships within a causality relation ontology (Mohammed et al., 2012). This enables a quicker connection of symptoms and possible reasons, thus making treatments easier and better. Another prominent example of data mining capabilities of ontologies is IBM's project *Watson* (Hoyt et al., 2016). Watson is one of the so-called DeepQA projects. It is a program designed to answer questions in natural language. The power of this system is created by the immense amount of data it can process and by its ability to connect relations via ontologies and similar concepts. Other use cases of ontologies enrich Semantic Web mining, mining health records for insights, fraud detection, and semantic publishing.

Since ontologies are designed to share knowledge, there are some popular ontologies (W3C, 2012a) that are primarily used in their genuine field of application. An example for this is the *Friend Of A Friend (FOAF)* ontology (Brickley & Miller, 2020) that is commonly used to describe people and social relationships on the Web. It can be complemented by the *Semantically-Interlinked Online Communities (SIOC)* ontology (Bojars & Breslin, 2020) which extends the FOAF ontology with online communities and the definitions of their content. The *Music Ontology* (Raimond et al., 2020, 2007) is the most used knowledge base to describe information related to the music industry. Search engines like Yahoo, SearchMonkey and BestBuy utilize the *GoodRelations* (Hepp, 2020) vocabulary to describe products sold online and to improve search results when shopping online.

In the next sections, we will discuss research that has been conducted in the field of ontologies and their usage within machine learning and explainability approaches. Because these fields are relatively new, most work is directed at basic concepts that are needed for

Table 15: Overview of research that improves ontologies.

Approach	Description	References
-	Automated generation	Wong et al. (2011)
OntoLearn	Automated extraction, generation	Navigli and Velardi (2004)
-	Semi-Automated engineering	Maedche and Staab (2001)

Table 16: Overview of ontology approaches that improve ML.

Approach	Description	References
GLUE	Improve ontologies with ML	Doan et al. (2004)
-	Improve ML with ontologies	Tsymbal et al. (2007)
-	Improve ML with ontologies	Xu et al. (2015b)

future applications of those systems. Nevertheless, there are some promising steps towards the implementation of domain knowledge in AI and their explanations.

7.3.3 Improving Ontologies

Due to their already growing use, research focuses greatly on improving ontologies. Wong et al. (2011) provide a method for the automation of ontology generation from domain data. *OntoLearn* (Navigli & Velardi, 2004) uses websites and shared documents to extract domain terminology to then arrange it hierarchically and finally create an ontology. Other researchers go even further and attempt to completely automate the development of ontologies. Maedche and Staab (2001) propose semi-automatic import, extraction, pruning, refinement, and evaluation of ontologies, providing the ontology engineer with coordinated tools for ontology modeling.

7.3.4 Ontologies and Machine Learning

In many use cases, more than one ontology is needed and therefore multiple ontologies need to be connected. ML strategies could improve the mapping of those multiple ontologies and thereby enable the construction of large knowledge bases. Doan et al. (2004) developed *GLUE*, a system that employs learning techniques to semi-automatically create semantic mappings between ontologies. Conversely, ontologies are used to refine ML by incorporating them into the training process. An application in bio-medicine (Tsymbal et al., 2007) integrates domain specific ontologies to improve decision-making. The existence of large knowledge bases is used to mitigate the complexity of such a high dimensional field of science and to use combined knowledge for an overall better understanding. A different concept is proposed by Xu et al. (2015b). This concept uses semantic relations among features that are defined inside an ontology to improve random forests for image classification. Splits in each decision tree are determined by semantic relations which automatically include the hierarchical structure of knowledge. This usage of ontologies proves to be very helpful for image recognition, as different detectable things are made up of parts that are all included in the ontology. Table 16 lists an overview of the approaches.

7.3.5 Ontologies and Explainability

The goal of explanations for AI systems is to make complex models understandable for human beings. This includes the connection of related facts and their informational importance on the result. Ontologies as concepts that represent knowledge about the different relations of data provide a huge potential in making complex data structures better understandable.

A first step towards a better understanding of these systems is the *ML-Schema* developed by Publio et al. (2018). This top-level ontology provides a set of classes, properties, and restrictions that can be used to define information on ML algorithms, data sets, and experiments. A uniform description of these elements simplifies the exchange of such collected knowledge. The next step covers the concept of explanations themselves. McGuinness et al. (2007) showcase *PML 2*, a combination of three ontologies that is designed to serve as an interlingua for the sharing of generated explanations. It consists of the *provenance ontology*, the *justification ontology* and the *trust relation* ontology (also known as PML-P, PML-J and PML-T) which are all used to describe information within system responses that can then be used to generate an explanation. This approach focuses on the information about how a system generates output, on its dependencies and on the possible information that could transport trust. *PML 2* creates a uniform definition to share the building blocks of an explanation.

The structure of an ontology does not only provide a good possibility to share knowledge but it can also be utilized directly. Confalonieri et al. (2019) present *Trepan reloaded*, a recent approach that uses domain ontologies while also generating an explanation. A decision tree is used to explain the decision process of a neural network. The goal is to benefit from the structured knowledge within an ontology and to apply it to the next generation of the decision tree. A conducted user study illustrates that explanations which follow a structure similar to the human understanding of data are better understandable. Explanations for neural networks try to make the process from the input to the output understandable. Sarker et al. (2017) analyze this relation with the help of ontologies. Their proof of concept describes the potential of background knowledge for the explainability of such systems. Panigutti et al. (2020) extend the relational concepts over time and present *Doctor XAI*. Showcased for medical situations, this approach deals with multi-labeled, sequential, ontology-linked data. This means it is able to deduce connections between information that happen over time. It creates a new source of knowledge that can be used to improve the predictions and explanations made in such critical areas. Supported by the concept of knowledge sharing, transfer learning (West et al., 2007) is a sub category of ML and aims at reusing learned information and applying it to new learning problems. Chen et al. (2018) use ontologies to implement explanations for transfer learning. The goal is to create human-centered explanations that enable non-ML experts to detect positive and negative transfers. This way, they can decide what to transfer and when to transfer in order to create an optimized transfer learning setting. Geng et al. (2019) present an alternative approach for this that utilizes Knowledge Graphs. Both emphasize the potential of knowledge bases in human-centered explanations. The approach developed by Mahajan et al. (2019) presents a method that uses (partial) structural causal models to generate actionable counterfactuals.

Table 17: Overview of ontology approaches in XAI.

Approach	Description	References
ML-Schema	ML described by an ontology	Publio et al. (2018)
PML 2	Explanations described by an ontology	McGuinness et al. (2007)
Trepan reloaded	Explanations supported by an ontology	Confalonieri et al. (2019)
-	Explanations supported by an ontology	Sarker et al. (2017)
Doctor XAI	Explanations supported by an ontology	Panigutti et al. (2020)
-	Explanations supported by an ontology	Chen et al. (2018)
Feasible counterfactuals	Counterfactual Explanations supported by causal constraints	Chen et al. (2018)
Thales XAI Platform	Explanations supported by an ontology	Lécué et al. (2019)

Other works try to incorporate Knowledge Graphs as well, with the *Thales XAI Platform* (Lécué et al., 2019) being the first of its kind. One approach for making AI systems applicable to critical situations, this platform provides example-based and feature-based explanations or counterfactuals, using textual and visual representations. In addition, explanations based on semantics are generated with the help of Knowledge Graphs. Semantic-Web tools are used to enrich the data for ML with context information. Explanations are then generated by using a Knowledge Graph and the context information of the ML result to identify representative semantic relations.

8. Discussion and Open Challenges

There is a lively debate about the need for explainability in machine learning in general and, more specifically, about what the main research focus should lie on. Proponents of explainable artificial intelligence argue that for a model to be reliable, first and foremost it needs to be understandable. Opponents take the view that sometimes, human reasoning is a complete black box as well, so there is no actual need for explainability for every purpose. Maybe someone wants to understand the reasoning of a doctor for preferring a particular medical treatment but cares less about why his loan was approved. Both sides seem to agree that in regulated areas, the need for understandable models are crucial. To get a sense of the global model behaviour, the methods of, *e.g.*, a simple decision tree or BRL can be applied. While the tree and the rule lists do not vary in the nature of their respective approaches (since a tree can be transformed into a decision list as well), decision lists are more easily understandable and directly convey their supposed meaning to a "lay user". Especially if the model has learned only a few significant features, decision lists approximate their relationships well. If more features are required to make a meaningful distinction, the decision tree has the upper hand while still being small and readable with a reduced amount of training. Both decision trees and rule lists have the characteristic to be reproducible and, thus, being what Lipton (Lipton, 2018) calls *human-simulatable*. SP-Lime does not allow a "lay user" to make that reproducible judgment because the presentation of various single explanations does not necessarily imply how a new instance will be classified. If the submodular-pick is selected wisely and if the variation between instances is reasonably clear, the user will probably be able to make a good guess, but that also requires the understanding of how the sample explanations were generated. Explanation generation methods like PDPs seem to have little value to "lay users". However, for those users who do possess knowledge about the relevant techniques, these methods can yield a very rough approximation of the model judgment based on the features. With regards to local surrogate explanations, we applied SHAP and LIME. As Lundberg and Lee (2017)

state, both of them follow a similar approach of explanation. However, apart from the actual visual representation, they do not make a real difference for the user. The only difference is the approach by which the results are being produced and that yields marginal different values for single features. Since the user probably is interested in both a general direction and the most significant contributions, the underlying method is irrelevant.

When it comes to explainability in particular, dimension reduction methods are mentioned in the literature (Kittler, 1986). However, dimension reduction approaches need to be used carefully when the goal is to achieve more explainability because they could conceal explainability. The pre-processing of the data is important because it can already lead to a more understandable model. Whether the goal is to achieve a more interpretable or a more accurate model, it is never a bad idea to mind and mine the quality of the models' input data. After illustrating different explainability approaches, especially in the local area, it becomes clear that what was shown by Lundberg and Lee (2017) has an influence on the intrinsic quality of explanations as well. Many of the explanations resort to feature importance which could make intuitive sense to a statistician but is likely not the way a "lay user" would like the process to be explained. Besides this, there are several more challenges that need to be tackled in the future:

Interpretable versus black box models - Rethinking the problem in the first place: Before we train models that solve the addressed problem, we need to reflect on the question, "What are we actually looking for? Do we really need a black box model?" Rudin (2018) describes the explosion of research in the field of explainable ML with surrogate models regarding high stakes decisions as problematic. They state that building interpretable models that are accurate as black boxes should be considered. Therefore, more research in the field where a black box and an interpretable model are competing against each other is needed. If we had interpretable models that prove to be reliable as black boxes, would there still be a need to use them?

Measuring and comparing explainability: According to our research, what is missing is a standard procedure to measure, quantify, and compare the explainability of enhancing approaches that allows scientists to compare these different approaches. Although some research is being done in this field 3.6, a standardized procedure is needed. The performance of a classifier is evaluated by, *e.g.*, accuracy, recall, and the F1-score. The need for likewise metrics to evaluate explainability is crucial. Miller (2019) states that most of the work relies on the authors' intuition about explainable machine learning. An essential point within the research of XSML is to have metrics that describe the overall explainability and to be able to compare different models regarding their level of explainability.

Improving explanations with ontologies: Another research area that should be further addressed is the combination of ontologies with explanations. We already addressed this point in section 7.3.5 but further research with practical use cases needs to be done. Furthermore, the advantage of the combination with ontologies must be examined by several user studies.

Trust in machine learning models: What if we measure both the explainability and the trust within a model but both are missing? Can we provide more trust just-in-time? What are the possibilities to raise the trust in the model? All those questions remain hard to answer without more research and especially user-centered experiments in this field. Schmidt and Biessmann (2019) introduced a quantitative measure of trust in ML decisions

and conducted an experiment. In their experiment, they examined two methods, COVAR, a glass-box method, and LIME, a black box method. They found that COVAR yielded more interpretable explanations. Thereby, they highlighted the usefulness of simple methods. Lipton (2017) states that addressing the foundations of the problem by discovering what explainability really is will be crucial to see meaningful progress within this field.

User studies regarding specific explainability aspects: Almost every day, a paper is published that purports to solve the explainability problem algorithmically. However, another important aspect are user studies. There are only a few user experiments in the area of explainability, but much more experiments are needed to cover the topic holistically. Poursabzi-Sangdeh et al. (2018) measure trust by determining the difference between the prediction of the model and the participant's prediction. As a use case, they predict housing prices. Ribeiro et al. (2016c) conducted a user study to measure whether the participants would trust the prediction model. Yin et al. (2019) measure the frequency with which they revise their predictions to match the predictions of the model and their self-reported levels of trust in the model. El-Bekri et al. (2019) evaluate three different explanation approaches based on the users' trust by a within-subject design study. Lage et al. (2019) conducted a user study to find out what makes explanations human-interpretable by systematically varying properties of explanation to measure the effect of these variations on the performance of several tasks. The tasks were: simulating the system's response, verifying a suggested response, and counterfactual reasoning. One of their findings included that, across all experiments, counterfactual questions showed significantly lower accuracy. Herman (2017) differentiates between a descriptive and persuasive explanation generation task. Whereas the first describes an explanation within a feature space generated by the explainable or interpretable approach, the latter adds cognitive functions, user preferences, and expertise to the explanation.

9. Conclusion

The relevance of explainable machine learning can be seen in many areas. The high number of published research papers in certain areas can probably be attributed to the fact that there is a fairly high need to provide explanations to users in these areas , *e.g.*, in the medical domain. For example, explainable ML was already used to learn more about COVID-19 (Chen et al., 2020; Fan et al., 2020; Rezaul et al., 2020; F. Bao et al., 2020).

This paper introduced different problem definitions of explainable SML and categorized and reviewed past and recent approaches in each field. The use cases mentioned throughput the paper were supposed to illustratively depict each approach according to the problem definitions given. The overall goal of the paper was to gather an explanation that offers an broad overview of different approaches in the field of explainable SML. For example, by providing an overview over the most influential features to the decision by considering a local linear approximation of the model, an explanation can be generated. For an intuitive example, we can think of a linear approximation to a complex model as a sort of representative-analysis where we can illustrate for a user how representative the features for the predicted class based on the most important features actually were. This approach can give a reasonable human interpretation that matches with most subconscious processes of decision-making which often rely on the representativeness of a certain instance. However,

when humans try to explain themselves, they use different approaches to knowledge that can hardly be captured in a classifier. This is due to the fact that humans often explains themselves by referring to post-factum coherent stories. Rather than providing two features and an importance of those features with a specified class, the human mind would tend to build a story around those features that explains why it seems obvious that the respective instance belongs to a specific class. In an explanation like this, all sorts of environmental conditions play a role, as the person telling the story seeks to build trust and understanding for her decision. Thus, the most we can strive for when explaining a model is a sort of human graspable approximation of the decision process.

Diving down into the ethical dilemmas of automated decision-making, especially self-driving cars move into the focus. If a self-driving car needs to decide whether it drives into a crowd of elderly or a crowd of young people, the ethical controversy begins. The moral machine (Massachusetts Institute of Technology, 2017) is an attempt to assess ethical dilemmas of the kind in which self-driving cars need to choose between insoluble situations. Would you be able to choose? If so, could you explain your decision?

Acknowledgments

This work is partially supported by the Ministry of Economic Affairs of the state Baden-Württemberg within the KI-Fortschrittszentrum "Lernende Systeme", Grant No. 036-170017. We would like to thank our student assistants (Maximilian Franz, Felix Rittmann, Jonas Steinhäuser and Jasmin Kling) who supported us during our research.

References

Abdollahi, B. & Nasraoui, O. (2016). Explainable restricted boltzmann machines for collaborative filtering. *arXiv preprint arXiv:1606.07129*.

Abdollahi, B. & Nasraoui, O. (2017). Using explainability for constrained matrix factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (pp. 79–83).

ACM (2017). Statement on algorithmic transparency and accountability.

Adadi, A. & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*.

Adler, P., Falk, C., Friedler, S., Rybeck, G., Scheidegger, C., Smith, B., & Venkatasubramanian, S. (2016). Auditing black-box models for indirect influence. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*: IEEE.

Amatriain, X. (2017). More data or better models?

Andrews, R., Diederich, J., & Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*.

Andrzejak, A., Langner, F., & Zabala, S. (2013). Interpretable models from distributed data via merging of decision trees. In *Computational Intelligence and Data Mining (CIDM), 2013 IEEE Symposium on*: IEEE.

Angelov, P. & Soares, E. (2019). Towards explainable deep neural networks (xdnn). *arXiv preprint arXiv:1912.02523*.

Askham, N., Cook, D., Doyle, M., Fereday, H., Gibson, M., Landbeck, U., Lee, R., Maynard, C., Palmerand, G., & Schwarzenbach, J. (2013). The six primary dimensions for data quality assessment. *DAMA UK Working Group*, (pp. 432–435).

Augasta, M. G. & Kathirvalavakumar, T. (2012). Reverse engineering the neural networks for rule extraction in classification problems. *Neural processing letters*.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., & M. Zoeller, K.-R. (2010). How to explain individual classification decisions. *Journal of Machine Learning Research*.

Bahdanau, D., Cho, K., & y. Bengio (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Balestriero, R. (2017). Neural decision trees. *arXiv preprint arXiv:1702.07360*.

Barakat, N. & Diederich, J. (2004). Learning-based rule-extraction from support vector machines. In *The 14th International Conference on Computer Theory and applications ICCTA'2004*: not found.

Barakat, N. H. & Bradley, A. P. (2007). Rule extraction from support vector machines: A sequential covering approach. *IEEE Transactions on Knowledge and Data Engineering*.

Barbella, D., Benzaid, S., Christensen, J. M., Jackson, B., Qin, X. V., & Musicant, D. (2009). Understanding support vector machine classifications via a recommender system-like approach. In *DMIN*.

Bastani, O., Kim, C., & Bastani, H. (2017). Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504*.

Bengio, Y. & Pearson, J. (2016). When ai goes wrong we won't be able to ask it why.

Berkson, J. (1953). A statistically precise and relatively simple method of estimating the bio-assay with quantal response, based on the logistic function. *Journal of the American Statistical Association*.

Bertsimas, D., Chang, A., & Rudin, C. (2011). Ordered rules for classification: A discrete optimization approach to associative classification. In *SUBMITTED TO THE ANNALS OF STATISTICS*: Citeseer.

Bhatt, U., Ravikumar, P., & J. M. F. Moura, J. (2019). Towards aggregating weighted feature attributions. *arXiv preprint arXiv:1901.10040*.

Bien, J. & Tibshirani, R. (2009). Classification by set cover: The prototype vector machine. *arXiv preprint arXiv:0908.2284*.

Bien, J. & Tibshirani, R. (2011). Prototype selection for interpretable classification. *The Annals of Applied Statistics*.

Biran, O. & Cotton, C. (2017). Explanation and Justification in Machine Learning: A survey. In *IJCAI-17 Workshop on Explainable AI (XAI)*.

Biran, O. & McKeown, K. R. (2017). Human-centric justification of machine learning predictions. In *IJCAI*.

Biswas, S. K., Chakraborty, M., Purkayastha, B., Roy, P., & Thounaojam, D. M. (2017). Rule extraction from training data using neural network. *International Journal on Artificial Intelligence Tools*.

Bohanec, M., Bor***v***stnar, M. K., & Robnik-***v***Sikonja, M. (2017). Explaining machine learning models in sales predictions. *Expert Systems with Applications*.

Bojars, U. & Breslin, J. G. (2020). Semantically-interlinked online communities.

Boz, O. (2002). Extracting decision trees from trained neural networks. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*: ACM.

Breiman, L. (2017). *Classification and regression trees*. Routledge.

Brickley, D. & Miller, L. (2020). The foaf project.

Burkart, N., Huber, M. F., & Faller, P. (2019). Forcing interpretability for deep neural networks through rule-based regularization. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)* (pp. 700–705).: IEEE.

Byrum, J. (2017). The challenges for artificial intelligence in agriculture.

Cambridge (2020). *The Cambridge dictionary of psychology*. Cambridge University Press.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., & Elhadad, N. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: ACM.

Charniak, E. (1991). Bayesian networks without tears. *AI magazine*.

Chen, D., Fraiberger, S. P., Moakler, R., & Provost, F. (2015). Enhancing transparency and control when drawing data-driven inferences about individuals. *Proceedings of 2016 ICML Workshop on Human Interpretability in Machine Learning*.

Chen, J., Lécué, F., Pan, J. Z., Horrocks, I., & Chen, H. (2018). Knowledge-based transfer learning explanation. *CoRR*, abs/1807.08372.

Chen, Y., Ouyang, L., Bao, S., Li, Q., Han, L., Zhang, H., Zhu, B., Xu, M., Liu, J., Ge, Y., et al. (2020). An interpretable machine learning framework for accurate severe vs non-severe covid-19 clinical type classification. *medRxiv*.

Clark, P. & Niblett, T. (1989). The cn2 induction algorithm. *Machine learning*.

Cleland, S. (2011). Google's 'infringenovation' secrets.

Cohen, W. (1995). Fast effective rule induction. In *Machine Learning Proceedings 1995*. Elsevier.

Confalonieri, R., del Prado, F. M., Agramunt, S., Malagarriga, D., Faggion, D., Weyde, T., & Besold, T. R. (2019). An ontology-based approach to explaining artificial neural networks. *CoRR*, abs/1906.08362.

Cortez, P. & Embrechts, M. J. (2011). Opening black box data mining models using sensitivity analysis. In *Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on*: IEEE.

Craven, M. & Shavlik, J. W. (1996). Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*.

Cui, Z., Chen, W., He, Y., & Chen, Y. (2015). Optimal action extraction for random forests and boosted trees. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*.

Datta, A., Sen, S., & Zick, Y. (2016). Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *Security and Privacy (SP), 2016 IEEE Symposium on*: IEEE.

Doan, A., Madhavan, J., Domingos, P., & Halevy, A. (2004). *Ontology Matching: A Machine Learning Approach*, (pp. 385–403). Springer Berlin Heidelberg: Berlin, Heidelberg.

Doran, D., Schulz, S., & Besold, T. R. (2017). What does explainable ai really mean? a new conceptualization of perspectives. *arXiv preprint arXiv:1710.00794*.

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

Dosilović, F. K., Brcić, M., & Hlupić, N. (2018). Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*.

Dua, D. & Graff, C. (2017). UCI machine learning repository.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of statistics*.

El-Bekri, N., Kling, J., & Huber, M. F. (2019). A study on trust in black box models and post-hoc explanations. In *International Workshop on Soft Computing Models in Industrial and Environmental Applications*: Springer.

Etchells, T. A. & Lisboa, P. J. G. (2006). Orthogonal search-based rule extraction (osre) for trained neural networks: a practical and efficient approach. *IEEE transactions on neural networks*.

Europa.eu (2017). Official journal of the european union: Regulations.

F. Bao, a. Y. H., Liu, J., Chen, Y., Li, Q., Zhang, C., Han, L., Zhu, B., Ge, Y., Chen, S., et al. (2020). Triaging moderate covid-19 and other viral pneumonias from routine blood tests. *arXiv preprint arXiv:2005.06546*.

Fan, X., Liu, S., Chen, J., & Henderson, T. C. (2020). An investigation of covid-19 spreading factors with explainable ai techniques. *arXiv preprint arXiv:2005.06612*.

Fischer, G., Mastaglio, T., Reeves, B., & Rieman, J. (1990). Minimalist explanations in knowledge-based systems. In *Twenty-Third Annual Hawaii International Conference on System Sciences*, volume 3 (pp. 309–317 vol.3).

Fisher, A., Rudin, C., & Dominici, F. (2018). Model class reliance: Variable importance measures for any machine learning model class, from the" rashomon" perspective. *arXiv preprint arXiv:1801.01489*.

Freitas, A. (2014). Comprehensible classification models: a position paper. *ACM SIGKDD explorations newsletter*.

Friedman, J. H., Popescu, B. E., et al. (2008). Predictive learning via rule ensembles. *The Annals of Applied Statistics*.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine learning*.

Fu, L. (1994). Rule generation from neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*.

Fung, G., Sandilya, S., & Rao, R. B. (2008). Rule extraction from linear support vector machines via mathematical programming. In *Rule Extraction from Support Vector Machines*. Springer.

Geng, Y., Chen, J., Jimenez-Ruiz, E., & Chen, H. (2019). Human-centric transfer learning explanation via knowledge graph [extended abstract].

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*.

Gkatzia, D., Lemon, O., & Rieser, V. (2016). Natural language generation enhances human decision-making with uncertain information. *arXiv preprint arXiv:1606.03254*.

Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*.

Goodman, B. & Flaxman, S. (2016). Eu regulations on algorithmic decision-making and a "right to explanation". In *ICML workshop on human interpretability in machine learning (WHI 2016), New York, NY*.

Gruber, T. R. et al. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2), 199–221.

Gudivada, V., Apon, A., & Ding, J. (2017). Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations. *International Journal on Advances in Software*, 10(1), 1–20.

Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., & Giannotti, F. (2018a). Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820*.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018b). A survey of methods for explaining black box models. *ACM Comput. Surv.*

Gunning, D. (2017). Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA)*.

Gurumoorthy, K. S., Dhurandhar, A., & Cecchi, G. (2017). Protodash: Fast interpretable prototype selection. *arXiv preprint arXiv:1707.01212*.

Hall, P., Gill, N., Kurka, M., & Phan, W. (2017a). Machine learning interpretability with h2o driverless ai. *H2O.ai*.

Hall, P., Phan, W., & Ambati, S. (2017b). Ideas on interpreting machine learning.

Hara, S. & Hayashi, K. (2016). Making tree ensembles interpretable. *arXiv preprint arXiv:1606.05390*.

Hayashi, Y. (2013). Neural network rule extraction by a new ensemble concept and its theoretical and historical background: A review. *International Journal of Computational Intelligence and Applications*.

Helfert, M. & Ge, M. (2016). Big data quality-towards an explanation model in a smart city context. In *proceedings of 21st International Conference on Information Quality, Ciudad Real, Spain*.

Hendricks, L. A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., & Darrell, T. (2016). Generating visual explanations. In *European Conference on Computer Vision*: Springer.

Henelius, A., Puolamäki, K., Boström, H., Asker, L., & Papapetrou, P. (2014). A peek into the black box: exploring classifiers by randomization. *Data mining and knowledge discovery*.

Henelius, A., Puolamäki, K., & Ukkonen, A. (2017). Interpreting classifiers through attribute interactions in datasets. In *2017 ICML Workshop on Human Interpretability in Machine Learning (WHI)*.

Hepp, M. (2020). Good relations.

Herman, B. (2017). The promise and peril of human evaluation for model interpretability. *arXiv preprint arXiv:1711.07414*.

Hilton, D. J. (1990). Conversational processes and causal explanation. *Psychological Bulletin*.

Hinton, G. & Frosst, N. (2017). Distilling a neural network into a soft decision tree. In *Comprehensibility and Explanation in AI and ML (CEX), AI*IA*.

Hoehndorf, R. (2010). What is an upper level ontology? *Ontogenesis*.

Hoffman, R., Mueller, S., Klein, G., & Litman, J. (2018). Metrics for explainable ai: Challenges and prospects. *arXiv preprint arXiv:1812.04608*.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine learning*.

Holzinger, A., Kickmeier-Rust, M., & Müller, H. (2019a). Kandinsky patterns as iq-test for machine learning. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction* (pp. 1–14).: Springer.

Holzinger, A., Langs, G., Denk, H., Zatloukal, K., & Müller, H. (2019b). Causability and explainabilty of artificial intelligence in medicine. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*.

Holzinger, A., Plass, M., Holzinger, K., Crisan, G. C., Pintea, C. M., & Palade, V. (2017). A glass-box interactive machine learning approach for solving np-hard problems with the human-in-the-loop. *arXiv preprint arXiv:1708.01104*.

Holzinger, A., Plass, M., Kickmeier-Rust, M., Holzinger, K., Crişan, G. C., Pintea, C. M., & Palade, V. (2019c). Interactive machine learning: experimental evidence for the human in the algorithmic loop. *Applied Intelligence*, 49(7), 2401–2414.

Hoyt, R. E., Snider, D., Thompson, C., & Mantravadi, S. (2016). Ibm watson analytics: Automating visualization, descriptive, and predictive statistics. *JMIR Public Health Surveill*, 2(2), e157.

Huysmans, J., Baesens, B., & Vanthienen, J. (2006). Iter: an algorithm for predictive regression rule extraction. In *International Conference on Data Warehousing and Knowledge Discovery*: Springer.

Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., & Baesens, B. (2011). An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*.

Jain, S. & Wallace, B. C. (2019). Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Jiang, T. & Owen, A. B. (2002). Quasi-regression for visualization and interpretation of black box functions.

Johansson, U., König, R., & Niklasson, L. (2004). The truth is in there-rule extraction from opaque models using genetic programming. In *FLAIRS Conference*: Miami Beach, FL.

Kabra, M., Robie, A., & Branson, K. (2015). Understanding classifier errors by examining influential neighbors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Kaggle (2017). The state of data science and machine learning.

Kamruzzaman, S. (2010). Rex: An efficient rule generator. *arXiv preprint arXiv:1009.4988*.

Kass, R. & Finin, T. (1988). The Need for User Models in Generating Expert System Explanations. *International Journal of Expert Systems*, 1(4).

Kim, B., Khanna, R., & Koyejo, O. O. (2016). Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*.

Kim, B., Rudin, C., & Shah, J. A. (2014). The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in Neural Information Processing Systems*.

Kim, B., Shah, J. A., & Doshi-Velez, F. (2015). Mind the gap: A generative approach to interpretable feature selection and extraction. In *Advances in Neural Information Processing Systems*.

Kittler, J. (1986). Feature selection and extraction. *Handbook of Pattern Recognition and Image Processing*.

Koh, P. W. & Liang, P. (2017). Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2), 233–243.

Krause, J., Perer, A., & Ng, K. (2016). Interacting with predictions: Visual inspection of black-box machine learning models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*: ACM.

Lage, I., Chen, E., He, J., Narayanan, M., Kim, B., Gershman, S., & Doshi-Velez, F. (2019). An evaluation of the human-interpretability of explanation. *arXiv preprint arXiv:1902.00006*.

Lakkaraju, H., Bach, S. H., & Leskovec, J. (2016). Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: ACM.

Lakkaraju, H., Kamar, E., Caruana, R., & Leskovec, J. (2017). Interpretable & explorable approximations of black box models. *arXiv preprint arXiv:1707.01154*.

Lakkaraju, H., Kamar, E., Caruana, R., & Leskovec, J. (2019). Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*.

Lash, M. T., Lin, Q., Street, W. N., & Robinson, J. G. (2017). A budget-constrained inverse classification framework for smooth classifiers. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*.

Laugel, T., Lesot, M. J., Marsala, C., Renard, X., & Detyniecki, M. (2017). Inverse classification for comparison-based interpretability in machine learning. *arXiv preprint arXiv:1712.08443*.

Laugel, T., Renard, X., Lesot, M., Marsala, C., & Detyniecki, M. (2018). Defining locality for surrogates in post-hoc interpretablity. *arXiv preprint arXiv:1806.07498*.

Lécué, F., Abeloos, B., Anctil, J., Bergeron, M., Dalla-Rosa, D., Corbeil-Letourneau, S., Martet, F., Pommellet, T., Salvan, L., Veilleux, S., & Ziaeefard, M. (2019). Thales xai platform: Adaptable explanation of machine learning systems - a knowledge graphs perspective. In *ISWC Satellites*.

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., & Wasserman, L. (2018). Distribution-free predictive inference for regression. *Journal of the American Statistical Association*.

Lei, T., Barzilay, R., & Jaakkola, T. (2016). Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Lent, M. V., Fisher, W., & Mancuso, M. (2004). An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the national conference on artificial intelligence*.

Letham, B., Rudin, C., McCormick, T. H., & Madigan, D. (2012). Building interpretable classifiers with rules using bayesian analysis. *Department of Statistics Technical Report tr609, University of Washington*.

Letham, B., Rudin, C., McCormick, T. H., & Madigan, D. (2015). Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*.

Lipton, Z., Kale, D., & Wetzel, R. (2016). Modeling missing data in clinical time series with rnns. *arXiv preprint arXiv:1606.04130*.

Lipton, Z. C. (2017). The doctor just won't accept that! *arXiv preprint arXiv:1711.08037*.

Lipton, Z. C. (2018). The mythos of model interpretability. *Queue*, 16(3), 31–57.

Looveren, A. V. & Klaise, J. (2019). Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584*.

Lou, Y., Caruana, R., & Gehrke, J. (2012). Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*: ACM.

Lou, Y., Caruana, R., Gehrke, J., & Hooker, G. (2013). Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*: ACM.

Lu, H., Setiono, R., & Liu, H. (1995). Neurorule: A connectionist approach to data mining. In *Proceedings of the 21st VLDB Conference Zurich, Switzerland*.

Lundberg, S. M. & Lee, S. (2017). A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30* (pp. 4765–4774). Curran Associates, Inc.

Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Maedche, A. & Staab, S. (2001). Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16, 72–79.

Mahajan, D., Tan, C., & Sharma, A. (2019). Preserving causal constraints in counterfactual explanations for machine learning classifiers. *arXiv preprint arXiv:1912.03277*.

Malioutov, D. M., Varshney, K. R., Emad, A., & Dash, S. (2017). Learning interpretable classification rules with boolean compressed sensing. In *Transparent Data Mining for Big and Small Data*. Springer.

Markowska-Kaczmar, U. & Chumieja, M. (2004). Discovering the mysteries of neural networks. *International Journal of Hybrid Intelligent Systems*.

Martens, D., Backer, M. D., Haesen, R., Vanthienen, J., Snoeck, M., & Baesens, B. (2007a). Classification with ant colony optimization. *IEEE Transactions on Evolutionary Computation*.

Martens, D., Baesens, B., & Gestel, T. V. (2009). Decompositional rule extraction from support vector machines by active learning. *IEEE Transactions on Knowledge and Data Engineering*.

Martens, D., Baesens, B., Gestel, T. V., & Vanthienen, J. (2007b). Comprehensible credit scoring models using rule extraction from support vector machines. *European journal of operational research*.

Martens, D., Huysmans, J., Setiono, R., Vanthienen, J., & Baesens, B. (2008). Rule extraction from support vector machines: an overview of issues and application in credit scoring. *Rule extraction from support vector machines*.

Martens, D. & Provost, F. (2014). Explaining data-driven document classifications. *Mis Quarterly*.

Martens, D., Vanthienen, J., Verbeke, W., & Baesens, B. (2011). Performance of classification models from a user perspective. *Decision Support Systems*.

Mashayekhi, M. & Gras, R. (2017). Rule extraction from decision trees ensembles: New algorithms based on heuristic search and sparse group lasso methods. *International Journal of Information Technology & Decision Making*.

Massachusetts Institute of Technology (2017). The moral machine.

McGuinness, D. L., Ding, L., da Silva, P., & Chang, C. (2007). Pml 2: A modular explanation interlingua. In *ExaCt*.

Mead, A. (1992). Review of the development of multidimensional scaling methods. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 41(1), 27–39.

Meinshausen, N. (2010). Node harvest. *The Annals of Applied Statistics*.

Melis, D. A. & Jaakkola, T. (2018). Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*.

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

Mohammed, O., Benlamri, R., & Fong, S. (2012). Building a diseases symptoms ontology for medical diagnosis: An integrative approach. In *The First International Conference on Future Generation Communication Technologies*.

Molnar, C. (2018). A guide for making black box models explainable. *URL: https://christophm. github. io/interpretable-ml-book*.

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., & Müller, K. R. (2017). Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*.

Montavon, G., Samek, W., & Müller, K. R. (2018). Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*.

Murdoch, J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. (2019). Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*.

Navigli, R. & Velardi, P. (2004). Learning domain ontologies from document warehouses and dedicated web sites. *Computational Linguistics*, 30(2), 151–179.

Ninama, H. (2013). Ensemble approach for rule extraction in data mining. *Golden Reaserch Thoughts*.

Odajima, K., Hayashi, Y., Tianxia, G., & Setiono, R. (2008). Greedy rule generation from discrete data and its use in neural network rule extraction. *Neural Networks*.

Otero, F. E. B. & Freitas, A. (2016). Improving the interpretability of classification rules discovered by an ant colony algorithm: Extended results. *Evolutionary Computation*.

Panigutti, C., Perotti, A., & Pedreschi, D. (2020). Doctor xai: An ontology-based approach to black-box sequential data classification explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20 (pp. 629–639). New York, NY, USA: Association for Computing Machinery.

Park, D. H., Hendricks, L. A., Akata, Z., Schiele, B., Darrell, T., & Rohrbach, M. (2016). Attentive explanations: Justifying decisions and pointing to the evidence. *arXiv preprint arXiv:1612.04757*.

Phillips, R. L., Chang, K. H., & Friedler, S. A. (2017). Interpretable active learning. *arXiv preprint arXiv:1708.00049*.

Plumb, G., Molitor, D., & Talwalkar, A. S. (2018). Model agnostic supervised local explanations. In *Advances in Neural Information Processing Systems*.

Poursabzi-Sangdeh, F., Goldstein, D. G., Hofman, J. M., Vaughan, J. W., & Wallach, H. (2018). Manipulating and measuring model interpretability. *arXiv preprint arXiv:1802.07810*.

Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., & Flach, P. (2020). Face: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* (pp. 344–350).

Publio, G. C., Esteves, D., Lawrynowicz, A., ce Panov, P., Soldatova, L., Soru, T., Vanschoren, J., & Zafar, H. (2018). Ml-schema: Exposing the semantics of machine learning with schemas and ontologies.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*.

Quinlan, J. R. (1996). Bagging, boosting, and c4.5. In *AAAI/IAAI, Vol. 1*.

Quinlan, J. R. (2014). *C4.5: programs for machine learning*. Elsevier.

Raimond, Y., Abdallah, S., Sandler, M., & Giasson, F. (2007). The music ontology. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*.

Raimond, Y., Abdallah, S., Sandler, M., & Giasson, F. (2020). The music ontology.

Rezaul, K., DÃ**P**hmen, T., Rebholz-Schuhmann, D., Decker, S., Cochez, M., & Beyan, O. (2020). Deepcovidexplainer: Explainable covid-19 predictions based on chest x-ray images. *arXiv*, (pp. arXiv–2004).

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016a). Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016b). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: ACM.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016c). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*: ACM.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.

Robnik, M. & Kononenko, I. (2008). Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering.*

Robnik-*v*Sikonja, M. & Kononenko, I. (2008). Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering.*

Rudin, C. (2018). Please stop explaining black box models for high stakes decisions. *CoRR.*

Rüping, S. (2005). Learning with local models. In *Local Pattern Detection* (pp. 153–170).

Rüping, S. (2006). Learning interpretable models. *Doctoral Dissertation, University of Dortmund.*

Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685.*

Russell, C. (2019). Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (pp. 20–28).

Saabas, A. (2015). Treeinterpreter. `https://github.com/andosa/treeinterpreter`.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., & Müller, K., Eds. (2019). *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning.* Springer.

Samek, W., Wiegand, T., & Müller, K. R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296.*

Sarker, M. K., Xie, N., Doran, D., Raymer, M., & Hitzler, P. (2017). Explaining trained neural networks with semantic web technologies: First steps.

Schaaf, N. & Huber, M. F. (2019). Enhancing decision tree based interpretation of deep neural networks through l1-orthogonal regularization. *arXiv preprint arXiv:1904.05394.*

Schetinin, V., Fieldsend, J. E., Partridge, D., Coats, T. J., Krzanowski, W. J., Everson, R. M., Bailey, T. C., & Hernandez, A. (2007). Confident interpretation of bayesian decision tree ensembles for clinical applications. *IEEE Transactions on Information Technology in Biomedicine.*

Schmidt, P. & Biessmann, F. (2019). Quantifying interpretability and trust in machine learning systems. *arXiv preprint arXiv:1901.08558.*

Schmitz, G., Aldrich, C., & Gouws, F. S. (1999). Ann-dt: an algorithm for extraction of decision trees from artificial neural networks. *IEEE Transactions on Neural Networks.*

Selvaraju, R. R., Das, A., Vedantam, R., Cogswell, M., Parikh, D., & Batra, D. (2016). Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391.*

Sestito, S. & Dillon, T. (1992). Automated knowledge acquisition of rules with continuously valued attributes. In *Proceedings of the 12th international conference on expert systems and their applications, 1992.*

Sethi, K. K., Mishra, D. K., & Mishra, B. (2012). Extended taxonomy of rule extraction techniques and assessment of kdruleex. *International Journal of Computer Applications*.

Setiono, R., Azcarraga, A., & Hayashi, Y. (2014). Mofn rule extraction from neural networks trained with augmented discretized input. In *Neural Networks (IJCNN), 2014 International Joint Conference on*: IEEE.

Setiono, R., Baesens, B., & Mues, C. (2008). Recursive neural network rule extraction for data with mixed attributes. *IEEE Transactions on Neural Networks*.

Setiono, R. & Liu, H. (1997). Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*.

Shapley, L. S. (1951). *Notes on the n-Person Game–II: The Value of an n-Person Game*. Technical report, U.S. Air Force, Project Rand.

Shrikumar, A., Greenside, P., Shcherbina, A., & Kundaje, A. (2016). Not just a black box: Learning important features through propagating activation differences. In *33rd International Conference on Machine Learning*.

Si, Z. & Zhu, S. C. (2013). Learning and-or templates for object recognition and detection. *IEEE transactions on pattern analysis and machine intelligence*.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., & Wattenberg, M. (2017). Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

Strumbelj, E., Bosnić, Z., Kononenko, I., Zakotnik, B., & Kuhar, C. (2010). Explanation and reliability of prediction models: the case of breast cancer recurrence. *Knowledge and information systems*.

Strumbelj, E. & Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*.

Su, G., Wei, D., Varshney, K. R., & Malioutov, D. M. (2015). Interpretable two-level boolean rule learning for classification. *arXiv preprint arXiv:1511.07361*.

Su, G., Wei, D., Varshney, K. R., & Malioutov, D. M. (2016). Learning sparse two-level boolean rules. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*: IEEE.

Subianto, M. & Siebes, A. (2007). Understanding discrete classifiers with a case study in gene prediction. In *Seventh IEEE International Conference on Data Mining 2007* (pp. 661–666).: IEEE.

Sundararajan, M., Taly, A., & Yan, Q. (2016). Gradients of counterfactuals. *arXiv preprint arXiv:1611.02639*.

Swartout, W., Paris, C., & Moore, J. (1991). Explanations in knowledge systems: design for explainable expert systems. *IEEE Expert*, 6(3), 58–64.

Taha, I. & Ghosh, J. (1996). Three techniques for extracting rules from feedforward networks. *Intelligent Engineering Systems Through Artificial Neural Networks*.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*.

Tjoa, E. & Guan, C. (2019). A survey on explainable artificial intelligence (xai): Towards medical xai. *arXiv preprint arXiv:1907.07374*.

Tolomei, G., Silvestri, F., Haines, A., & Lalmas, M. (2017). Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: ACM.

Tsymbal, A., Zillner, S., & Huber, M. (2007). Ontology - Supported Machine Learning and Decision Support in Biomedicine. In *International Conference on Data Integration in the Life Sciences*, volume 4544 (pp. 156–171).

Turner, R. (2016). A model explanation system. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*.

Tversky, A. & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science*.

Tversky, A. & Kahneman, D. (1981). The framing of decisions and the psychology of choice. *Science*.

Ustun, B. & Rudin, C. (2014). Methods and models for interpretable linear classification. *arXiv preprint arXiv:1405.4047*.

Ustun, B. & Rudin, C. (2016). Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*.

Ustun, B. & Rudin, C. (2017). Optimized risk scores. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: ACM.

van der Maaten, L. & Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov), 2579–2605.

Vedantam, R., Bengio, S., Murphy, K., Parikh, D., & Chechik, G. (2017). Context-aware captions from context-agnostic supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Verbeke, W., Martens, D., Mues, C., & Baesens, B. (2011). Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert Systems with Applications*.

Voosen, P. (2017). How AI detectives are cracking open the black box of deep learning. *Science Magazine*.

W3C (2012a). *Good Ontologies*. W3C recommendation, W3C. https://www.w3.org/wiki/Good_Ontologies.

W3C (2012b). *OWL 2 Web Ontology Language Document Overview (Second Edition)*. W3C recommendation, W3C. https://www.w3.org/TR/2012/REC-owl2-overview-20121211/.

W3C (2014). *Ressource Description Framework(RDF)*. W3C recommendation, W3C. https://www.w3.org/RDF/.

Wachter, S., Mittelstadt, B., & Russell, C. (2018). Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law & Technology*, 31(2).

Wang, F. & Rudin, C. (2014). Falling rule lists. *arXiv preprint arXiv:1411.5899*.

Wang, F. & Rudin, C. (2015). Falling rule lists. In *18th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Wang, J., Fujimaki, R., & Motohashi, Y. (2015a). Trading interpretability for accuracy: Oblique treed sparse additive models. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: ACM.

Wang, R. Y. & Strong, D. M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*, 12(4), 5–33.

Wang, T., Rudin, C., Doshi-Velez, F., Liu, Y., Klampfl, E., & MacNeille, P. (2015b). Or's of and's for interpretable classification, with application to context-aware recommender systems. *arXiv preprint arXiv:1504.07614*.

Wang, T., Rudin, C., Velez-Doshi, F., Liu, Y., Klampfl, E., & MacNeille, P. (2016). Bayesian rule sets for interpretable classification. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*: IEEE.

Weiner, J. (1980). Blah, a system which explains its reasoning. *Artificial intelligence*, 15(1-2), 19–48.

Weller, A. (2017). Challenges for transparency. *arXiv preprint arXiv:1708.01870*.

West, J., Ventura, D., & Warnick, S. (2007). Spring research presentation: A theoretical foundation for inductive transfer. Retrieved 2007-08-05.

Wiegreffe, S. & Pinter, Y. (2019). Attention is not not explanation. *arXiv preprint arXiv:1908.04626*.

Wold, S., Esbense, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3), 37–52.

Wong, W., Liu, W., & Bennamoun, M. (2011). Ontology learning from text: A look back and into the future. *ACM Computing Surveys - CSUR*, 44, 1–36.

Wu, M., Hughes, M. C., Parbhoo, S., Zazzi, M., Roth, V., & Doshi-Velez, F. (2018). Beyond sparsity: Tree regularization of deep models for interpretability. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., & Bengio, Y. (2015a). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*.

Xu, N., Jiangping, W., Qi, G., Huang, T., & Lin, W. (2015b). Ontological random forests for image classification. *International Journal of Information Retrieval Research*, 5, 61–74.

Yang, C., Rangarajan, A., & Ranka, S. (2018a). Global model interpretation via recursive partitioning. *arXiv preprint arXiv:1802.04253*.

Yang, H., Rudin, C., & Seltzer, M. (2016). Scalable bayesian rule lists. unpublished.

Yang, Y., Morillo, I. G., & Hospedales, T. M. (2018b). Deep neural decision trees. *arXiv preprint arXiv:1806.06988*.

Yin, M., Vaughan, J. W., & Wallach, H. (2019). Understanding the effect of accuracy on trust in machine learning models. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*: ACM.

Yin, X. & Han, J. (2003). Cpar: Classification based on predictive association rules. In *Proceedings of the 2003 SIAM International Conference on Data Mining*: SIAM.

Zhang, Y. & Chen, X. (2018). Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*.

Zhao, X., Wu, Y., Lee, D. L., & Cui, W. (2019). iforest: Interpreting random forests via visual analytics. *IEEE transactions on visualization and computer graphics*.

Zhou, Z. H., Chen, S. F., & Chen, Z. Q. (2000). A statistics based approach for extracting priority rules from trained neural networks. In *ijcnn*: IEEE.

Zhou, Z. H., Jiang, Y., & Chen, S. F. (2003). Extracting symbolic rules from trained neural network ensembles. *Ai Communications*.

Zilke, E., Mencía, L., & Janssen, F. (2016). Deepred–rule extraction from deep neural networks. In *International Conference on Discovery Science*: Springer.