

Prolog-based agnostic explanation module for structured pattern classification

Gonzalo Nápoles^{a,*}, Fabian Hoitsma^a, Andreas Knoblen^a, Agnieszka Jastrzebska^b, Maikel Leon Espinosa^c

^a*Department of Cognitive Science & Artificial Intelligence, Tilburg University, The Netherlands.*

^b*Faculty of Mathematics and Information Science, Warsaw University of Technology, Poland.*

^c*Department of Business Technology, Miami Herbert Business School, University of Miami, USA*

Abstract

This paper presents a Prolog-based reasoning module to generate counterfactual explanations given the predictions computed by a black-box classifier. The proposed symbolic reasoning module can also resolve what-if queries using the ground-truth labels instead of the predicted ones. Overall, our approach comprises four well-defined stages that can be applied to any structured pattern classification problem. Firstly, we pre-process the given dataset by imputing missing values and normalizing the numerical features. Secondly, we transform numerical features into symbolic ones using fuzzy clustering such that extracted fuzzy clusters are mapped to an ordered set of predefined symbols. Thirdly, we encode instances as a Prolog rule using the nominal values, the predefined symbols, the decision classes, and the confidence values. Fourthly, we compute the overall confidence of each Prolog rule

*Corresponding author

Email address: `g.r.napoles@uvt.nl` (Gonzalo Nápoles)

using fuzzy-rough set theory to handle the uncertainty caused by transforming numerical quantities into symbols. This step comes with an additional theoretical contribution to a new similarity function to compare the previously defined Prolog rules involving confidence values. Finally, we implement a chatbot as a proxy between human beings and the Prolog-based reasoning module to resolve natural language queries and generate counterfactual explanations. During the numerical simulations using synthetic datasets, we study the performance of our system when using different fuzzy operators and similarity functions. Towards the end, we illustrate how our reasoning module works using different use cases.

Keywords: Explainable artificial intelligence, counterfactual explanations, symbolic reasoning, fuzzy clustering, fuzzy-rough sets.

1. Introduction

Within the AI contributions in recent years, one can observe a growing number of approaches related to model explainability and interpretability [13]. In simple terms, model explainability is a property that relies on the ability to describe how a given algorithm works, or more specifically, how a given algorithm decides for certain input data. Such explanations can be generated using the model’s knowledge structures or an agnostic proxy method. In contrast, model interpretability comprises a spectrum given by three properties [22, 4]: *transparency*, *decomposability* and *simulatability*, although the last one could be subject to debate. Unfortunately, state-of-the-art algorithms devoted to solving complex problems rarely have these valuable properties by themselves.

As many domains (e.g., healthcare, education, politics, law) look to deploy deep learning systems, accountability [4], and transparency [35] have become of paramount relevance. One of the reasons is that if we cannot deliver explainability properly, we will simply and seriously limit AI applications' potential scope. However, aside from the legal and professional considerations that need to be made, there is also an argument that improving explainability is essential, even in more prosaic business scenarios, as understanding how an algorithm is actually working can help align better the activities of data scientists and analysts with the key questions and needs of their organization.

The literature reports interesting agnostic post-hoc methods to produce explanations from black boxes, but they involve some limitations. The first shortcoming is the insufficient focus on the symbolic explanations. In practice, this results in a relatively inflexible approach to generating descriptions, where descriptions are almost entirely static (such as in the dictionary approaches mentioned by Stepin et al. [36]). Moreover, there is a clear benefit in producing explanations using comprehensible symbols that humans can easily understand. The second neglected aspect is the automatic quality evaluation of the generated explanations. Relating generated descriptions with a robust mathematical approach is essential to quantify the explanations' confidence. This aspect is almost entirely neglected in the literature. Lastly, many existing methods are tailored to work only with a particular machine learning model. For example, Hatwell et al. [16] delivered a method to be used with random forests, while Montavon et al. [26] proposed an approach with a limited application to deep neural networks.

The issues mentioned above serve as the motivation behind our proposal. In particular, we see the problem of generating explanations from black boxes as a reasoning problem rather than a merely descriptive task. We will focus on producing counterfactual explanations resolving natural language queries to accomplish our goal. The counterfactual explanation basis can be summarized with the following question: what small changes could be made to the feature vector to reach the desired outcome?

Therefore, the main contribution of this paper consists of a symbolic reasoning module to generate counterfactual explanations from the predictions computed by a black-box classifier. The module can also resolve what-if queries using the ground-truth labels instead of the predicted ones. The proposed approach uses a potentially trainable knowledge base consisting of Prolog fuzzy rules mined from a set of labeled instances. The workflow of our proposal is briefly described as follows. After imputing the missing values and normalizing the numerical features, we build a set of fuzzy prototypes to describe each numerical feature during the granulation step. Such prototypes are obtained using a fuzzy clustering algorithm and subsequently associated with a predefined set of symbols that fulfill an order relation. Afterward, we create the Prolog knowledge base where problem instances are encoded as fuzzy symbolic rules such that each symbol is paired with its confidence value. To quantify the confidence of explanations, we compute the confidence of each rule using fuzzy-rough sets as defined by the membership value of each rule to the corresponding positive region. In that regard, we propose a new parametrized similarity function to compare Prolog fuzzy rules, which focuses on the certainty of symbolic terms in the rule antecedent. Finally,

we implement a chatbot as a proxy between human beings and the Prolog-based reasoning module to resolve natural language queries and generate counterfactual explanations.

It should be stated that the trainable component mentioned above concerns the parameters attached to the fuzzy granulation process, such as the fuzzy implicators or the similarity function used to build the fuzzy-rough sets. However, this paper does not explore such training capabilities beyond a sensitivity analysis conducted with synthetic datasets.

The remainder of the paper is organized as follows. Section 2 reports relevant pieces of research concerning the generation of agnostic explanations. Section 3 elaborates on the knowledge base construction steps, which are related to extracting the fuzzy prototypes from numerical representations and creating the Prolog fuzzy rules. Section 4 explains how to compute the rule confidence using fuzzy-rough sets, while Section 5 briefly goes over the technical details concerning the chatbot handling the natural language queries. Section 6 conducts a sensitivity analysis and presents a proof-of-concept with several use cases. Section 7 provides concluding remarks and comments on future research endeavors.

2. Related work on agnostic explanations

For a long time, the non-direct interpretation issue of popular machine learning models (such as random forests, gradient boosting machines, and neural networks) has been recognized and reported [28]. The existing approaches that aim to explain the inner workings of black-box data processing mechanisms differ in several aspects. Firstly, we distinguish models that op-

erate globally and locally. The former describes a machine learning model behavior as a whole. An example of such an approach is the accumulated local effects method [2]. The latter group aims at explaining individual predictions. This section focuses on methods falling into this category since the proposed model belongs to this group.

An essential ability of local models is to describe the contribution of each feature to the decisions. We may distinguish several approaches that focus on features in the existing state-of-the-art solutions. Examples of such approaches are LIME (Local Interpretable Model-agnostic Explanations) [34] and SHAP (SHapley Additive exPlanations) [24]. LIME explains the predictions of any model by building a white-box local surrogate model. This interpretable surrogate model can be used to explain the individual prediction. An important disadvantage reported in [39] is that the process of building LIME explanations is nondeterministic, which intuitively goes against the idea that explanatory models should be transparent [23]. SHAP is a game-theoretic approach to explain the predictions of a machine learning algorithm. This method connects optimal credit allocation with local explanations using the classical Shapley values from game theory. The authors in [20] show that mathematical problems arise when Shapley values are used for feature importance and that the mitigating solutions necessarily induce further issues. In addition, they argue that Shapley values are not a natural solution to the human-centric goals of explainability.

In the subject domain, a much-desired property is model-agnosticism, which means the method is suitable for explaining the predictions of any prediction-oriented model. On the opposite side of the spectrum, we have

model-specific approaches. In the group of model-specific approaches, we shall mention the high share of approaches paired with neural models, which pose particular troubles when interpreting their behavior. We may mention, for example, LRP (Layer-wise Relevance Propagation technique) [3, 26] and NAMs (Neural Additive Models) [1]. LRP can explain the predictions of a complex neural network by linking the network’s output with input features. The method propagates the prediction backward in the network using propagation rules to produce the explanation. NAMs is a method dedicated to deep neural network classifiers and is constructed as a linear combination of neural networks. NAM describes the relationship between the given input feature and the produced output.

Generating counterfactual explanations is often solved with the aid of decision rules to provide an end-user with an explanation of the reasons that led to the final prediction. Examples of rule-generating approaches are explanations with high-precision rules called anchors, representing local, sufficient conditions for predictions [33], methods extracting decision trees from trained neural networks [7], Collection of High Importance Random Path Snippets (CHIRPS) [16], LORE (LOcal Rule-based Explanations) [15], and Multi-Objective Counterfactuals (MOC) method [8]. Let us briefly describe the inner working of the last three methods.

CHIRPS extracts a decision path from each tree in the forest that contributes to the majority classification and then uses frequent pattern mining to identify the most commonly occurring split conditions. Then, a conjunctive form rule is constructed where the antecedent terms are derived from the attributes that had the most influence on the classification. This rule

is returned alongside estimates of the rule’s precision and coverage on the training data, along with counterfactual details. LORE is model-agnostic and aims to find a local interpretable predictor on a synthetic neighborhood generated by a genetic algorithm. Then it derives from the logic of the local interpretable predictor a meaningful explanation consisting of a decision rule – which explains the reasons of the decision – and a set of counterfactual rules, suggesting the changes in the instance’s features leading to a different outcome. MOC provides counterfactual explanations of how given features contributed to a decision for a given single problem instance. The advantage of MOC is that it is model-agnostic and works for numerical and categorical input features. It relies on solving a multi-objective optimization problem – it searches for the optimal set of counterfactual explanations in a predefined search space. An open problem for MOC is how to let a user select the counterfactuals that meet their particular trade-off between the objectives. Furthermore, we should mention the method by Wachter et al. [38] which minimizes a loss function between a model prediction for the given counterfactual and the desired outcome. Unfortunately, this method is not scalable and does not handle categorical features.

Yet another strategy of generating counterfactual descriptions for a trained machine learning model that should be mentioned involves generating an opposite prediction and then describing the differences that had to take place to get the opposite prediction. Examples of this approach include are Model Agnostic suPervised Local Explanations (MAPLE) [31], DIverse Counterfactual Explanations (DICE) [27], Feasible and Actionable Counterfactual Explanations (FACE) [32]. MAPLE runs in a post-hoc manner and is local (concerns

a single instance). It combines random forests with feature selection methods to return counterfactual explanations based on feature importance. DICE is based on an optimization mechanism with constraints. FACE focuses on returning "reachable" counterfactual explanations (the simplest paths leading to explanations are considered, the whole model is a graph).

We should mention that Hoitsma et al. [17] presented an approach for generating counterfactual explanations, which is conceptually related to the algorithm presented in this paper. The cited article discusses a method that produces counterfactual explanations from the predictions computed by a Fuzzy Cognitive Map-based classifier. This method outputs a set of Prolog rules where the feature vector was encoded as a symbolic vector denoting the rule antecedent, while the rule consequent encoded the classifier's predictions. However, the approach in [17] has several limitations. Firstly, symbolic descriptions are produced using a relatively inconvenient approach since the domain expert must fine-tune Gaussian membership functions manually. Secondly, the proposed module does not have an option to output sentences in natural language. Finally, it lacks a mechanism to evaluate the consistency of the generated fuzzy rules. These issues are resolved in the new algorithm presented in the following sections.

3. Building the Prolog knowledge base

This section elaborates on the steps needed to build the Prolog knowledge base consisting of a set of fuzzy rules. Firstly, we transform numerical quantities into meaningful symbols using a fuzzy clustering algorithm. Relying on a symbolic paradigm is challenging for AI systems, as pointed out by

d'Avila Garcez et al. [9]. Nonetheless, existing studies have demonstrated that such an approach can enhance the explanation capability of learning systems [10, 25]. Secondly, we create fuzzy Prolog rules using the derived symbolic terms and the decision classes determined by the classifier. Besides, each symbolic term is attached with a confidence value to assess the extent to which the symbolic representation is accurate. It should be stated that the literature includes other approaches to Prolog knowledge base construction and uncertainty management, such as the probabilistic approach discussed by De Raedt and Kimmig [11]. However, probability theory, fuzzy sets, and fuzzy-rough sets tackle different types of uncertainties, and replacing one with another often leads to misleading interpretations. In this paper, we use fuzzy set theory as a mathematical model for imprecision and rough set theory as a mathematical model for inconsistency. The former is needed when obtaining the symbolic knowledge representations, while the latter is needed when quantifying the consistency of each rule.

3.1. Deriving the symbolic terms

Structured pattern classification problems regularly involve both discrete and continuous features. While the values of discrete features can be included in a Prolog knowledge base, performing symbolic reasoning with continuous features might be far from ideal. This section shows the use of fuzzy logic theory to transform numerical features into symbolic knowledge representation and quantify the uncertainty of such a transformation.

Let $\mathbf{F} = \mathbf{N} \cup \mathbf{D}$ the set of problem features describing a specific pattern classification problem where \mathbf{N} is the set of numerical features while \mathbf{D} is the set of discrete features. As mentioned, the values of $f_i \in \mathbf{D}$ are symbols, and

their confidence degrees are assumed to be one.

To obtain meaningful symbols from numerical quantities, we build c fuzzy sets for each feature $f_i \in \mathbf{N}$. Each fuzzy set is associated with a linguistic term (e.g., *low*, *medium* and *high*) that provides meaning to the fuzzy set. Therefore, in our approach, meaning is closely related to the magnitude of numerical representations in a normalized scale. The number of fuzzy sets c defines the granularity level of explanations: the more fuzzy sets, the more detailed the explanations.

In this paper, the fuzzy sets for each numerical feature are determined by running the fuzzy c -means algorithm [5] on a two-dimensional space where each data point x_i is represented by a symmetric tuple (x_i, x_i) . This means that the algorithm will discover c fuzzy sets along the identity line. The fuzzy c -means algorithm creates a membership matrix $\mathbf{U}_{k \times c}$ where k is the number of data points to be processed (i.e., the number of instances in the dataset). Hence, $\mu_{ij} \in \mathbf{U}$ represents the degree to which the i -th data point belongs to the j -th fuzzy cluster. Furthermore, the algorithm returns a matrix of prototypes $\mathbf{Z}_{1 \times c}$ denoting the cluster centers. The fuzziness of this clustering algorithm is controlled by a fuzzification coefficient $m \in [1, \infty]$ where larger values indicate more fuzziness. Equations (1) and (2) display how to compute the membership values and the prototypes:

$$\mu_{ij} = \frac{1}{\sum_{l=1}^c \left(\frac{\|x_i - z_j\|}{\|x_i - z_l\|} \right)^{2/(m-1)}} \quad (1)$$

$$z_j = \frac{\sum_{i=1}^k \mu_{ij}^m \cdot x_i}{\sum_{i=1}^k \mu_{ij}^m}. \quad (2)$$

After computing the fuzzy sets, we must associate each data point with a linguistic term. Let $\mathbf{L} = \{l_i\}_{i=1}^c$ denote a partially ordered set consisting of c linguistic terms such that l_i precedes l_{i+1} in the ordering (e.g., *low* precedes *medium*). If we order the prototypes in \mathbf{Z} in ascending order according to their norm, we can establish a direct mapping between the symbolic terms in \mathbf{S} and the prototypes in \mathbf{Z} . Finally, we can represent the numerical quantity x_i with the fuzzy set reporting the largest membership value. This can be done with a function $\mu_{lj}(x_i)$ that produces its maximal value when x_i reports the largest membership value to the j -th fuzzy set.

3.2. Deriving the fuzzy Prolog rules

The next step is about building the fuzzy rules to be translated into a Prolog knowledge base. The fuzzy rules are symbolic because numerical values will be encoded as symbols using the procedure described in the previous sub-section. Such rules will include two types of confidence values. The confidence value associated with each linguistic term denotes how accurate the symbolic value represents the numerical ones. The confidence value associated with each rule denotes the extent to which that rule does not conflict with others. In the next section, we will explain how to compute the confidence values of rules using fuzzy-rough sets.

Let \mathbf{S} be the unordered version of \mathbf{L} where sub-indices no longer indicate that a linguistic term precedes another. The fuzzy rules have the structure: “IF $f_1 \in \mathbf{N}$ is $s_1 \in \mathbf{S}$ with certainty $\mu_{s_1}(x_1)$ AND ... AND $f_i \in \mathbf{N}$ is $s_i \in \mathbf{S}$

with certainty $\mu_{s_i}(x_i)$ AND ... AND $f_n \in \mathbf{N}$ is $s_n \in \mathbf{S}$ with certainty $\mu_{s_n}(x_n)$ THEN y_j with certainty $P(y_j|x_1, x_2, \dots, x_n)$ ". In these rules, x_i denotes the value of the i -th feature while s_i represents the corresponding symbolic term, whereas $P(y_j|x_1, x_2, \dots, x_n)$ is the confidence to which the classifier labeled the original instance with the decision class $y_j \in \mathbf{Y}$ given an instance with feature values x_1, x_2, \dots, x_n . In the case of discrete features, we assume that $s_i = x_i$ and $\mu_{s_i}(x_i) = 1$ to make the notation consistent.

The next step concerns the creation of a Prolog knowledge base to resolve natural language queries. In particular, we are interested in producing explanations for counterfactual queries with the format: "Which symbolic values should the features in $\mathbf{Q} \subset \mathbf{F}$ have taken to produce $y_j \in \mathbf{Y}$ instead of $y_i \in \mathbf{Y}$, given the symbolic values of features in $\mathbf{G} \subset \mathbf{F}$, such that $\mathbf{Q} \neq \emptyset$, $\mathbf{G} \neq \emptyset$ and $\mathbf{Q} \cap \mathbf{G} = \emptyset$?" . The experts are expected to provide the set of features $\mathbf{Q} \subset \mathbf{F}$ to be investigated, the symbolic values of known features $\mathbf{G} \subset \mathbf{F}$ and the alternative outcome $y_j \in \mathbf{Y}$. It should be mentioned that we can resolve these queries even when $\mathbf{Q} \cup \mathbf{G} \neq \mathbf{F}$. It is assumed that all numerical features have already been mapped to symbolic ones while retaining nominal ones defined in the original dataset.

Aiming at translating the fuzzy rules into Prolog ones, we rely on two predicates named **input** and **output**. The **input** predicate involves the symbolic terms describing a given instance and their confidence values, whereas the **output** predicate involves the decision class and the classifier's confidence. The Prolog rules have the format: **input**(id, $[[s_1, \mu_{s_1}(x_1)], \dots, [s_i, \mu_{s_i}(x_i)] , \dots, [s_n, \mu_{s_n}(x_n)], \mu_{R_*(\Theta_j)}(x)]$) :- **output**($[y_j, P(y_j|x_1, x_2, \dots, x_n)]$), where $\mu_{R_*(\Theta_j)}(x)$ stands for the confidence of the whole rule (to be addressed in the following

section). Observe that the antecedent and consequent in these Prolog rules are reversed when compared with the fuzzy rules defined above to match the inner semantic of counterfactual questions.

During the querying, experts need to provide all pieces of knowledge they have to resolve the query. This can be done by (i) asserting the dynamic predicate `output` with the desired outcome, (ii) instantiating some variables corresponding to features in $\mathbf{G} \subset \mathbf{F}$ when calling the `input` predicate, and (iii) asking for the symbolic values and confidences attached to features in $\mathbf{Q} \subset \mathbf{F}$. Moreover, it is advised to retract all asserted predicated before running any new query to avoid unexpected issues. The Prolog queries have the following format: `retractall(output()), assertz(output([y_j, p_0])), input([[v_1, p_1], ..., [v_i, p_i], ..., [v_n, p_n], p_{n+1}])`, where v_i is either the anonymous variable or a symbolic term used to instantiate the i -th symbolic variable, while p_i is either the anonymous variable or a confidence value. In addition, we can add constraints to symbolic terms or confidence values associated to variables to be resolved. The same applies to the confidence values of rules. For example, we can ask for symbolic solutions excluding certain linguistic terms, that have a confidence value greater than 0.6 and that appear in rules having a confidence degree greater than 0.8. Section 6 will introduce a case study illustrating the querying features.

It is worth mentioning that, although the knowledge base is composed of fuzzy Prolog rules, the symbolic reasoning is crisp. Therefore, after resolving a query, confidence values will be used to assess the extent to which the user can trust the explanations generated by the module.

4. Uncertainty quantification of fuzzy rules

In the procedure described in the previous section, each symbolic term resulting from the fuzzy granulation process was attached with a confidence value. However, such a confidence value would not provide information on the extent to which a rule conflicts with others. In this section, we develop a rule confidence measure based on the fuzzy-rough set formalism [12, 18] that takes into account the certainty of symbolic values involved in both the antecedent and the consequent of fuzzy Prolog rules.

Let Ω be the set of fuzzy rules, $\Theta \in \Omega$ a fuzzy set with membership function $\mu_{\Theta}(x)$ and $R \in \Psi(\Omega \times \Omega)$ a fuzzy binary relation with membership function $\mu_R(y, x)$. The membership function $\mu_{\Theta} : \Omega \rightarrow [0, 1]$ determines the degree to which $x \in \Omega$ is a member of Θ , while $\mu_R : \Omega \times \Omega \rightarrow [0, 1]$ is the degree to which y is a member of Θ taking into account the extent to which x is a member of Θ . If no confusion arises, we will represent $R(x)$ with its membership function $\mu_{R(x)}(y) = \mu_R(y, x)$ to lighten the notation. Finally, let $\Theta_j \in \Omega$ denote the set of fuzzy rules associated with the j -th decision class as determined by the black-box classifier.

The first step to derive the rule certainty measure is to build a fuzzy set for every rule set Θ_j resulting from partitioning Ω . Equation (3) shows the corresponding membership function,

$$\mu_{\Theta_j}(x) = \begin{cases} P(y_j|x), & x \in \Theta_j \\ 0, & x \notin \Theta_j \end{cases} \quad (3)$$

where $P(y_j|x)$ denotes the confidence of decision class y_j being correct given the instance encoded with the fuzzy rule $x \in \Omega$. In a nutshell, $P(y_j|x)$ is

the confidence attached to that decision class. If these probabilities are not available, then we can make $P(y_j|x) = 1$ when $x \in \Theta_j$.

The second step concerns the fuzzy binary relation $R \in \Psi(\Omega \times \Omega)$ with membership function $\mu_R(y, x)$ such that $x, y \in \Omega$. The latter function can be computed by combining the piece of knowledge $\mu_{\Theta_j}(x)$ with the similarity value between fuzzy rules x and y . Equation (4) formalizes how to compute the $\mu_R(y, x)$ function following this reasoning,

$$\mu_R(y, x) = \mu_{\Theta_j}(x)e^{-\lambda d(x,y)} \quad (4)$$

such that $\lambda > 0$ is a smoothing parameter and

$$d(x, y) = \sum_{i=1}^{|\mathbf{F}|} \sigma_i(x, y) \quad (5)$$

where $\sigma_i(x, y)$ is a feature-wise distance function that quantifies the dissimilarity between two linguistic terms. Equation (6) shows a simple feature-wise distance function used to compare nominal features, as turns out to be the case of symbolic terms describing a numerical quantity,

$$\sigma_i(x, y) = \begin{cases} 0, & s(y_i) = s(x_i) \\ 1, & s(y_i) \neq s(x_i). \end{cases} \quad (6)$$

The primary limitation of the feature-wise distance function depicted in Equation (6) is that it does not take into account the membership values associated with the linguistic terms. However, it might be the case that two numeric values are described by the same linguistic terms having attached quite different membership values. Equation (7) presents a fuzzy feature-wise distance function that tackles such a limitation,

$$\sigma_i(x, y) = \begin{cases} \Delta_i(x, y), & s(y_i) = s(x_i) \\ 1, & s(y_i) \neq s(x_i) \end{cases} \quad (7)$$

such that $\Delta_i(x, y) = 0.5 \cdot (1 - \min\{\mu_{s(x_i)}(x_i), \mu_{s(y_i)}(y_i)\})$. It should be highlighted that Equation (7) uses both the symbolic terms and their membership degrees when computing the distance between the fuzzy rules being compared. More explicitly, the dissimilarity is computed using the minimal membership degree attached to the symbolic terms in the antecedent of the rules. This value is subtracted since the dissimilarity between two symbolic values should increase as the minimal membership value decreases. Moreover, we enforce a constraint to ensure that the maximal value reported by our feature-wise distance function when the symbolic terms match is not greater than 0.5. Overall, we can assert that $d(x, y)$ will report differences even when all linguistic terms in the antecedent of rules x and y match, but their membership values are not maximal.

Finally, we can compute the confidence of rule $x \in \Omega$ as the membership degree of x to the fuzzy-rough positive region [18] associated with the rule's decision class. Fuzzy-rough positive regions are equivalent to the fuzzy-rough lower approximations, which can be understood as the fuzzy set of rules that do not conflict with each other (i.e., rules having a reasonably similar antecedent have different consequents). To obtain the fuzzy-rough lower approximations, we can use the degree of x being a member of Θ_j as defined by the fuzzy binary relation. This can be measured by the truth value of the statement “ $y \in R(x)$ implies $y \in \Theta_j$ ” under fuzzy sets $R(x)$ and Θ_j . The authors in [18] suggested using a necessity measure $\inf_{y \in \Omega} \mathcal{I}(\mu_R(y, x), \mu_{\Theta_j}(y))$ for such a quantification, where $\mathcal{I} : [0, 1] \times [0, 1] \rightarrow [0, 1]$ is a fuzzy implication

function. During numerical simulations, we will conduct a sensitivity analysis involving several fuzzy implicators.

Equation (8) shows the membership function for the lower approximation set $R_*(\Theta_j)$ associated with the j -th decision class,

$$\mu_{R_*(\Theta_j)}(x) = \min \left\{ \mu_{\Theta_j}(x), \inf_{y \in \Omega} \mathcal{I}(\mu_R(y, x), \mu_{\Theta_j}(y)) \right\}. \quad (8)$$

It can be observed that this fuzzy derivation of the rough lower approximations does not assume that $\mu_R(x, x) = 1, \forall x \in \Omega$. Instead, we compute the minimum between $\mu_{\Theta_j}(x)$ and $\inf_{y \in \Omega} \mathcal{I}(\mu_R(y, x), \mu_{\Theta_j}(y))$ to preserve the inclusiveness of $R_*(\Theta_j)$ in the fuzzy set Θ_j . Similarly, we could compute the fuzzy-rough lower approximations and the positive and boundary regions; however, they are irrelevant to our study.

5. Natural language Prolog queries

In this section, we go over the technical details concerning the chatbot handling the natural language queries posed by users. Moreover, we formalize the structure of these questions and gather them into categories ranging from exploratory data analysis to symbolic explanations.

5.1. Motivation

A Prolog knowledge base can be queried directly using Prolog but it might require some expertise from the user concerning declarative programming. Thus, we implemented a conversational agent (chatbot) intending to enable users without knowledge of the Prolog language and syntax to interact with the knowledge base by running natural language queries. By doing

that, we used the Rasa platform [6], which provides an open-source tool that can be used to build conversational virtual assistants. It allows for tight integration with the Python programming language, which is especially useful in our case, as the chatbot needs to interact with Prolog. The Python-Prolog interaction is enabled through the `pyswip` module [37].

Besides running queries on the Prolog knowledge base to answer what-if and counterfactual questions, the conversational agent enables users to explore data, train and validate a classifier, and construct a knowledge base using natural language. Furthermore, the chatbot can answer questions about these steps and create visualizations.

5.2. Implementation

The chatbot in Rasa consists of the following components: Natural Language Understanding (NLU) with intents and entities, rules, actions, and back-end Python script. The NLU component involves example phrases for each intent with annotated entities (if any). An intent represents what the user wants to achieve with the sent message, while an entity is a crucial piece of information within the sent message that needs to be considered in the chatbot's action. An entity can be used as a parameter for the action that follows. The example phrases are the training data for the NLU, which allows it to classify intents and extract entities from new user messages.

The actions are the predetermined responses the chatbot can return given a specific intent. This can be a simple piece of text or an image and can be extended through custom actions. These are actions run using the functions in the back-end Python script, used to load the data, perform exploratory data analysis and visualization, train the classifier, create the knowledge

base and run queries. The rules component specifies what action should be performed according to the classified intent.

5.3. Question categories

This section will formalize the different queries supported by the conversational agent using the proposed reasoning module as the back-end. The formalization will adopt the same notation as in the previous sections for mathematical rigor, although the query to be resolved should not be expected to contain any mathematical symbol. It shall be noted that the chatbot can handle different formulations for these queries.

5.3.1. Loading the classification dataset

The user can request the chatbot to load a certain dataset. The chatbot extracts the provided name of the dataset as an entity and searches for it in a folder with available datasets. The expected files should use the Attribute-Relation File Format (ARFF), an ASCII text file that describes a list of instances sharing a set of problem features.

5.3.2. Conducting exploratory data analysis

After loading the dataset, the user can explore the data by asking for visualizations and further information about the data. These can be roughly subdivided into the following categories.

Questions such as “*Let me see what the dataset contains*” return the number of instances and the features in the dataset. Also, the correlation between two features can be plotted by asking “*Show how $f_i \in \mathbf{N}$ and $f_j \in \mathbf{N}$ are correlated*”. Moreover, the correlation matrix can be asked for, e.g., with “*What does the correlation for the entire dataset look like?*”. The distribution

of a given feature can be plotted in a histogram by asking a question such as “*How are the values for $f_i \in \mathbf{N}$ distributed?*”.

5.3.3. *Training the classification model*

When asked, the chatbot trains a random forest classifier after splitting the data into a training and test set. The chatbot returns the accuracy on the test set, together with a confusion matrix. The user can provide the parameters (n_estimators, max_depth) for training the classifier, or default values are used. An example input from the user could be: “*Train the model with a maximum depth of 7 and use 200 estimators.*” After the training is complete, the user can ask how the data was split: “*How did you split the data for training and testing?*”

5.3.4. *Building the explanation module*

Once the classifier has been trained, the user can ask to construct the symbolic explanation module. This can be done by typing: “*Create the explanation module for this classifier*”. In that way, the symbolic terms for the numerical features and the fuzzy-rough sets, and the knowledge base in Prolog are constructed. The user can also ask for a plot of the membership of each instance to the fuzzy-rough regions: “*Plot the memberships of all the instances to the fuzzy-rough regions.*”

The user can request top rules in the Prolog knowledge base, for example, by asking: “*What are top rules in the knowledge base?*”. The top rules are those with the highest membership values to the positive fuzzy-rough region they belong to. Similarly, the difficulty level of the loaded classification problem can be requested by asking: “*Tell me the complexity of the classification*

problem please". The complexity of the loaded problem is calculated as one minus the average membership values of the instances to the positive fuzzy-rough region they belong to. Moreover, the user can ask for the explicit bias attached to a given protected feature, for example: "*Can you quantify the bias attached to $f_i \in \mathbf{F}$?*" The bias is calculated as proposed by Nápoles and Koumeri [29] using fuzzy-rough sets.

5.3.5. Answering what-if questions

The chatbot allows users to run queries on the Prolog knowledge base without being familiar with the declarative programming paradigm. These queries can be divided into what-if questions and counterfactual questions. With what-if questions, the user provides symbolic values for selected input features and the decision class and asks for the symbolic values of unknown input features. The structure of these queries and the expected answers can be formalized, as illustrated below.

User: If $g_1 \in \mathbf{G}$ is $s_1 \in \mathbf{S}$, ..., $g_a \in \mathbf{G}$ is $s_a \in \mathbf{S}$ and the outcome is $y_j \in \mathbf{Y}$, then what is $q_1 \in \mathbf{Q}$, ..., $q_b \in \mathbf{Q}$?

Bot: I have run the query for you. These are the results: $q_1 \in \mathbf{Q}$ is $s_1 \in \mathbf{S}$ with a certainty of $\mu_{s_1}(x_1)$, ..., $q_b \in \mathbf{Q}$ is $s_b \in \mathbf{S}$ with a certainty of $\mu_{s_b}(x_b)$. The entire rule has a certainty of $\mu_{R_*(\Theta_j)}(x)$.

5.3.6. Answering counterfactual questions

Counterfactual queries are similar to what-if queries, but they involve an alternative decision. To do that, the user provides an instantiation for known features and requests symbolic values for certain features to obtain the desired outcome. Counterfactual queries posed by the user and the answers

provided by the bot can be formalized as follows.

User: What values should $q_1 \in \mathbf{Q}$, \dots , $q_b \in \mathbf{Q}$ have taken, assuming that $g_1 \in \mathbf{G}$ is $s_1 \in \mathbf{S}$, \dots , $g_a \in \mathbf{G}$ is $s_a \in \mathbf{S}$, for the outcome to be y_j instead of y_i ?

Bot: I have run the query for you. These are the results: $q_1 \in \mathbf{Q}$ should be $s_1 \in \mathbf{S}$ with a certainty of $\mu_{s_1}(x_1)$, \dots , $q_b \in \mathbf{Q}$ should be $s_b \in \mathbf{S}$ with a certainty of $\mu_{s_b}(x_b)$. The entire rule has a certainty of $\mu_{R_(\Theta_j)}(x)$.*

While the literature reports other conversational agents to generate explanations, they cannot resolve queries leading to counterfactual explanations. For example, the agent in [14] uses decision trees and subgroup discovery algorithms to answer what-if questions in a decision-making context. In contrast, the agent in [21] is primarily devoted to answering what-if questions and computing feature relevance scores.

5.3.7. Exploring the recovered rules

After running a query, the user can request additional information about the rule(s) used to answer the query. The membership of the returned rule to the fuzzy-rough regions can be plotted by asking: “*Show the membership of this rule to the fuzzy rough regions*”. Similarly, the user can request the closest instances to a given rule as follows: “*What rule is the closest to this one?*” The closest instance to a given instance is determined by using the proposed distance function, as described in Section 4. Moreover, the difficulty of an instance can be asked as follows: “*Tell me how difficult this rule is*”. The difficulty level of an instance is assumed to be inversely correlated to its membership degree to its corresponding fuzzy-rough region: the larger this

membership value, the easier the instance.

6. Numerical simulations

In this section, we will assess the proposed Prolog-based reasoning module for symbolic reasoning. Firstly, we will conduct a sensitivity analysis to study the impact of the granulation parameters on the confidence of the induced fuzzy rules. Secondly, we will present a proof-of-concept illustrating the main use cases of the resulting conversational agent.

6.1. Sensitivity analysis

Aiming at conducting the sensitivity analysis, we will rely on ten structured pattern classification datasets having different properties (see Table 1). The number of instances in these datasets range from 57 to 900, the number of features from 6 to 28, and the number of decision classes from 2 to 11. Moreover, we have two highly imbalanced datasets as defined by the ratio between the minority class and the majority class.

The sensitivity analysis attempts to determine the impact of the choice of a fuzzy implicator on the granulation step (i.e., the process of extracting concepts from numerical data). A fuzzy implicator is a mapping that is used in our model to determine rule confidence (see Equation (8)). Overall, fine-tuning this parameter is necessary to determine optimal model parameters when it comes to the confidence of explanations. There are certain formal assumptions that a fuzzy implicator must satisfy: for any $x, x_1, x_2, y_1, y_2 \in [0, 1]$: if $x_1 \leq x_2$, then $\mathcal{I}(x_1, y) \geq \mathcal{I}(x_2, y)$ and if $y_1 \leq y_2$, then $\mathcal{I}(x, y_1) \leq \mathcal{I}(x, y_2)$. In addition, the following boundary conditions must be satisfied: $\mathcal{I}(0, 0) = 1$, $\mathcal{I}(1, 0) = 0$, and $\mathcal{I}(1, 1) = 1$. The literature concerning fuzzy

Table 1: Properties of each dataset used in the numerical simulations.

Index	Dataset	Instances	Numeric	Categorical	Classes	Imbalance
1	australian	690	8	6	2	no
2	crx	653	6	9	2	no
3	ecoli	336	7	0	8	71:1
4	flags	194	28	0	8	15:1
5	labor	57	8	8	2	no
6	diabetes	768	8	0	2	no
7	vehicle	846	18	0	4	no
8	vertebral3	310	6	0	3	no
9	vowel	990	10	3	11	no
10	wine	178	13	0	3	no

operators discusses several popular fuzzy implicators [19, 30]. In this paper, we will study the fuzzy implicators listed below.

- Fodor implicator, given by the following equation:

$$\mathcal{I}_{FD}(x, y) = \begin{cases} 1, & \text{if } x \leq y \\ \max(1 - x, y), & \text{otherwise} \end{cases} \quad (9)$$

- Goguen implicator, given by the following equation:

$$\mathcal{I}_{GG}(x, y) = \begin{cases} 1, & \text{if } x \leq y \\ \frac{y}{x}, & \text{otherwise} \end{cases} \quad (10)$$

- Gödel implicator, given by the following equation:

$$\mathcal{I}_{GD}(x, y) = \begin{cases} 1, & \text{if } x \leq y \\ y, & \text{otherwise} \end{cases} \quad (11)$$

- Łukasiewicz implicator, given by the following equation:

$$\mathcal{I}_{GD}(x, y) = \mathcal{I}_{LK}(x, y) = \min(1, 1 - x + y) \quad (12)$$

Figure 1 shows three-dimensional plots of listed fuzzy implicators. We selected them as candidate mappings because of their different behavior for various input values. While Łukasiewicz and Fodor are symmetric and exhibit abrupt changes in their shapes, Gödel and Goguen are neither symmetric nor change as abruptly as the former operators.

Figure 2 shows the average rule confidence for each dataset when varying the number of symbolic terms for different fuzzy implicators in Equation (8) and feature-wise distance functions in Equation (5). It can be noticed that Gödel and Goguen produce shallow confidence values while Łukasiewicz and Fodor report the same, highest confidence values. The results also suggest that using more symbolic terms does not necessarily lead to higher confidence values. Instead, more symbols will harm the interpretability due to the high granularity of the symbolic explanations. Finally, we can conclude that the proposed feature-wise distance function leads to higher confidence values than the baseline function when using less symbolic terms.

Figure 3 shows the average rule confidence for each dataset when varying the smoothing parameter $\lambda > 0$ in Equation (4) for different fuzzy implicators in Equation (8) and feature-wise distance functions in Equation (5).

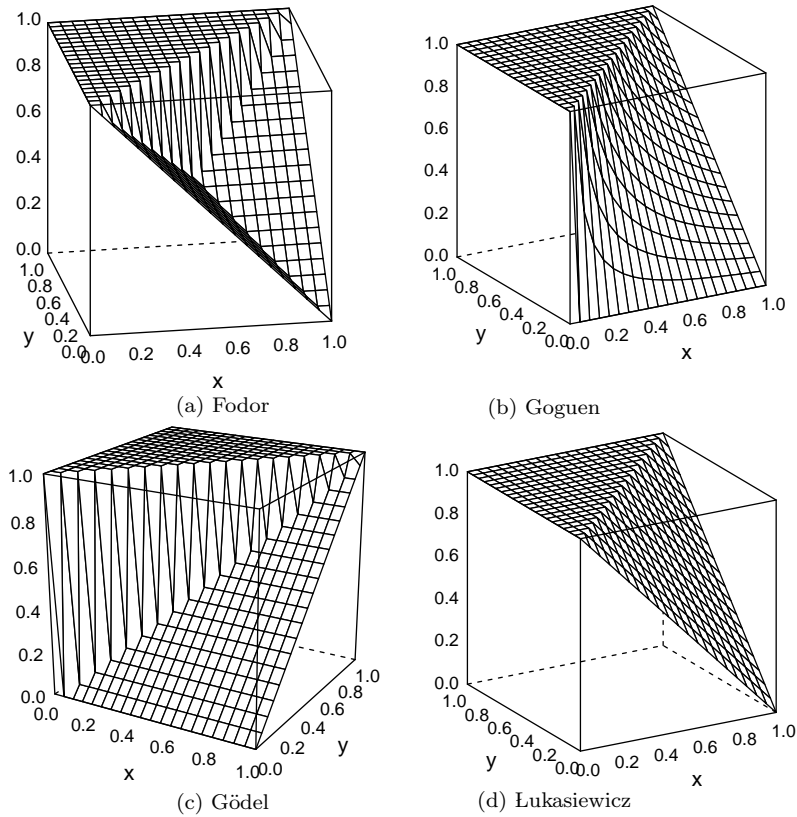
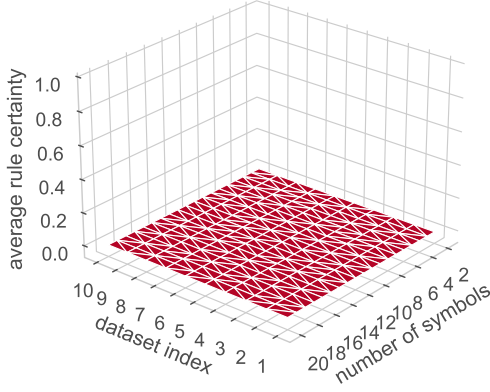


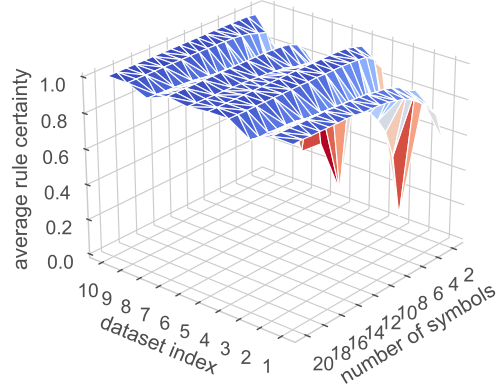
Figure 1: Plots of selected fuzzy implicators for arguments (x, y) .

Similar to the previous simulation results, Łukasiewicz and Fodor are the best-performing fuzzy operators. Moreover, it can be concluded that the larger smoothing parameter values lead to larger confidence values since the symbolic rules become easier to separate.

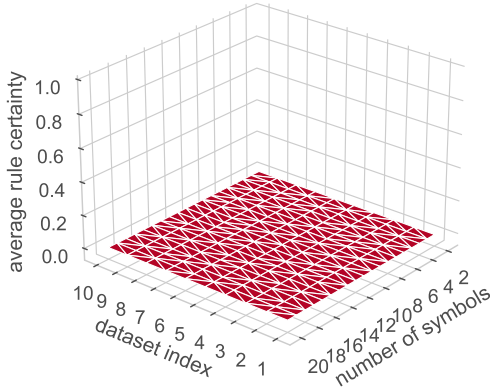
We configure the parameters associated with the information granulation and uncertainty quantification of rules from the sensitivity analysis. Łukasiewicz or Fodor is the recommended fuzzy implication operator, and the feature-wise distance function is the one proposed in Equation (7) while the smoothing parameter in Equation (4) is set to one.



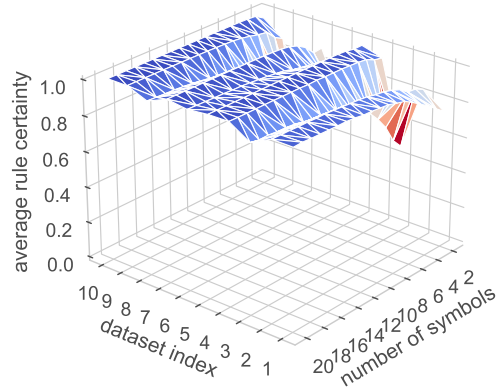
(a) Gödel, Goguen and Equation (6)



(b) Lukasiewicz, Fodor and Equation (6)



(c) Gödel, Goguen and Equation (7)

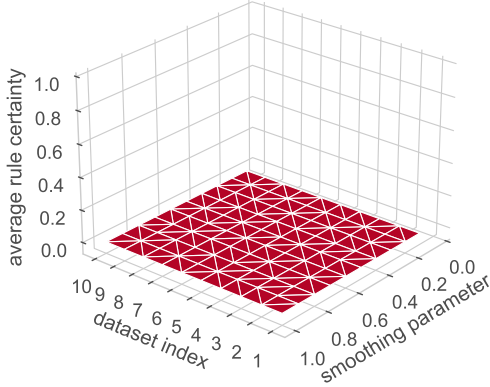


(d) Lukasiewicz, Fodor and Equation (7)

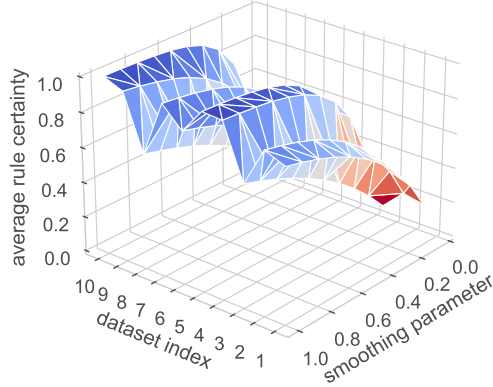
Figure 2: Average rule confidence for each dataset when varying the number of symbolic terms used to describe the numerical features for different fuzzy implication functions in Equation (8) and feature-wise distance functions in Equation (5).

6.2. Detailed case study

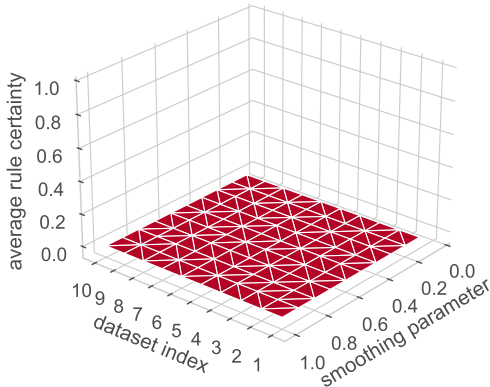
Next, we will illustrate the different questions supported by the proposed conversational agent using the *diabetes* dataset. The structure of these questions and their categories are introduced in Section 5.



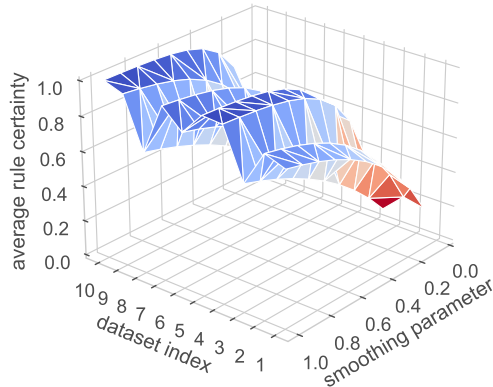
(a) Gödel, Goguen and Equation (6)



(b) Łukasiewicz, Fodor and Equation (6)



(c) Gödel, Goguen and Equation (7)



(d) Łukasiewicz, Fodor and Equation (7)

Figure 3: Average rule confidence for each dataset when varying the smoothing parameter $\lambda > 0$ in Equation (4) for different fuzzy implication functions in Equation (8) and feature-wise distance functions in Equation (5).

First of all, the user should ask the chatbot to load the desired dataset, for example, by typing “Load the diabetes dataset”. The recognized intent is “load_data”. The dataset to be loaded is extracted by Rasa as the entity “dataset”. Once the dataset is loaded, the user can request further

information, such as the features describing the problem and the number of instances. Figure 4 shows an example of this step where “data_stats” is the intent recognized by the conversational agent.

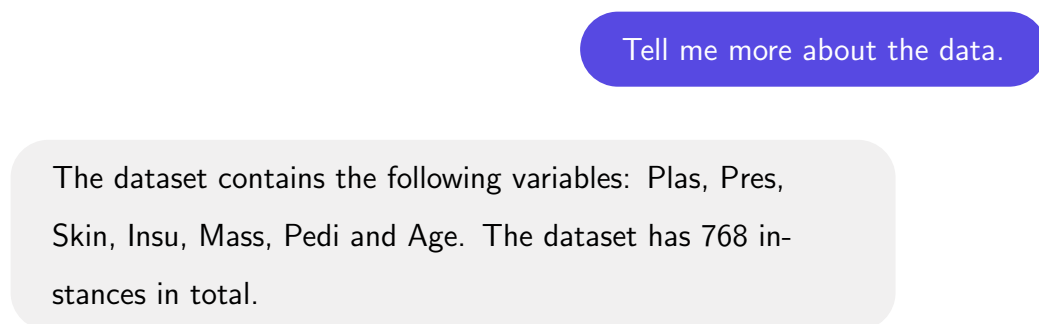


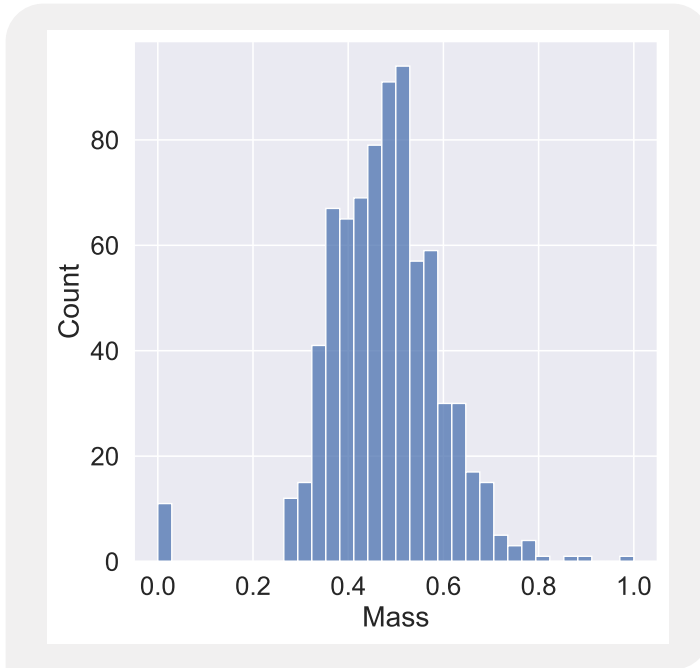
Figure 4: Basic statistics after having loaded a dataset.

Furthermore, the user can ask questions about problem features, e.g., how a continuous variable is distributed or how two continuous variables are related (using a scatter plot). In the former, the recognized intent is “plot_histogram”, and the extracted entity is “variable”. In the latter, the intent is “plot_correlation”, and Rasa extracts twice the entity “variable”. Examples can be seen in Figure 5 and Figure 6.

A correlation matrix may also be generated, as can be seen in Figure 7, provided that the dataset contains numeric features. This is performed when the intent “correlation_matrix” is recognized.

A random forest classifier can be trained on the dataset, as the intent “train_model” is recognized. The number of estimators and the maximum depth for the random forest can be specified. The chatbot also returns the training accuracy (see Figure 8). It is worth mentioning that including other classifiers is possible and would require little effort.

How is BMI distributed?



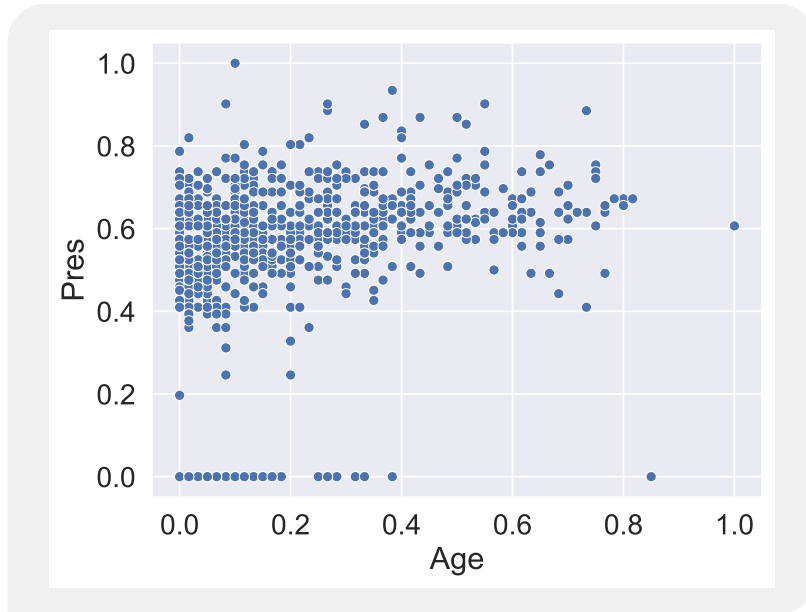
There you go!

Figure 5: An example of a histogram generated by the chatbot.

The user can request more information about the data split used during the training process, as illustrated in Figure 11. The default configuration is 80% for training and 20% for testing purposes. Upon recognizing the intent “train_test_samples”, the chatbot returns the train-test split configuration as a percentage and number of instances.

The user can ask the chatbot to construct the symbolic explanation module. As this request is recognized by the intent “train_explanation_module”,

How are Age and Pres correlated?

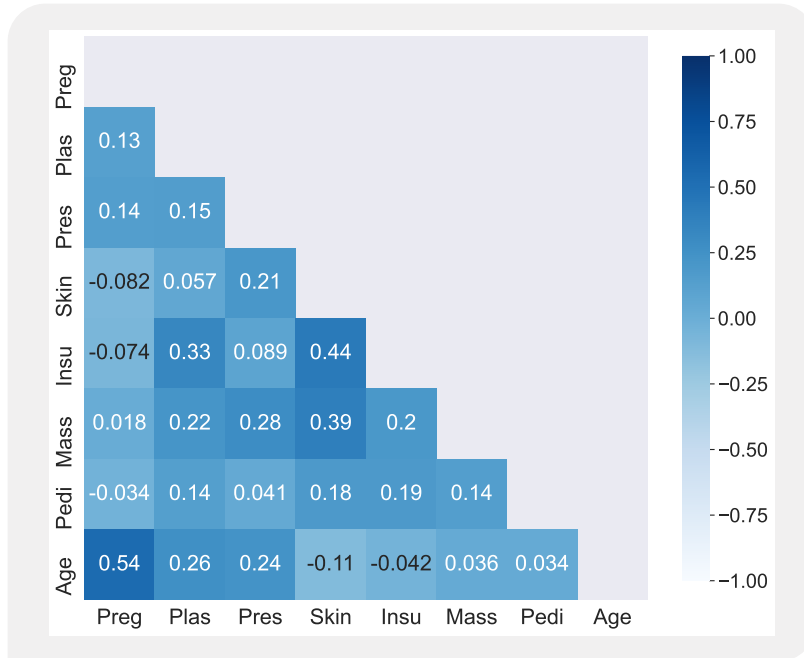


The correlation value between the submitted variables is 0.24. The associated p-value is $1.7520607075290974e-11$. The correlation does not seem to be very strong.

Figure 6: An example a pairwise correlation analysis performed by the chatbot.

the chatbot performs several steps that range from building the symbolic terms to quantifying the uncertainty of symbolic rules. Figure 10 shows the fuzzy prototypes extracted from numerical features describing the diabetes dataset. The axes denote the normalized values of these features; thus, prototypes will be located along the identity line. In this example, we extract

Show me the correlation matrix.



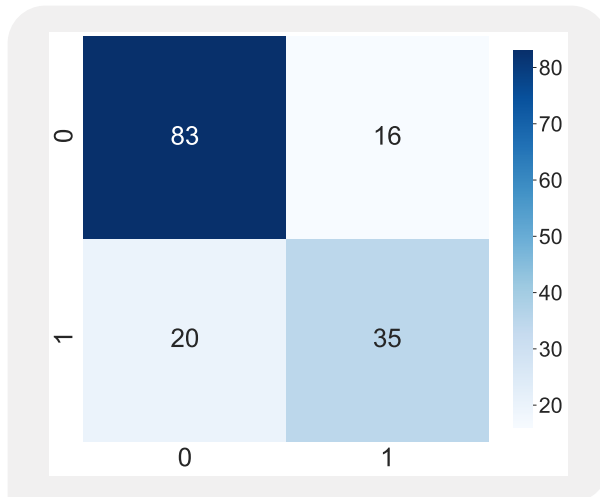
This is what the correlation matrix looks like.

Figure 7: An example of a correlation matrix generated by the chatbot.

$c = 5$ fuzzy prototypes for each numerical feature using the fuzzy c -means algorithm. Next, these prototypes are ordered and labeled with a predefined set of symbolic terms (*very low* - VL, *low* - L, *medium* - M, *high* - H, and *very high* - VH) that will be used to create the symbolic fuzzy rules. Finally, the fuzzy-rough regions are created, and the Prolog knowledge base is built. Figure 12 an excerpt from this knowledge base.

Once the symbolic explanation module is built, the user can pose what-if

Train the model on this data.



I successfully trained the random forest classifier on the dataset! The accuracy on the test set is 0.766. I included the confusion matrix for the predictions on the test set. If you want to achieve higher accuracy, feel free to train the model again with other parameters for the number of estimators and/or maximum depth. If you are satisfied with the achieved accuracy, we can continue!

Figure 8: Training of a Random Forest classifier by the chatbot.

or counterfactual questions (the latter being discussed later). What-if questions are recognized by the intent “run_full_query”. Rasa extracts the entities “known_concept” for variables specified as known, “value” for their respec-

How did you split the data?

I used 80% of the data to train the model, in this case that are 614 instances. For testing, I used the remaining 20% of the data, so 154 instances in total.

Figure 9: An example of requesting the data split in the chatbot.

tive values, “outcome” for the desired outcome, and “unknown_concept” for the variable to be resolved by the chatbot. Figure 13 shows an example of the answer provided by the conversational agent for a what-if question.

As the conversational agent returns a rule, the user can inquire about the membership of that rule to the fuzzy-rough regions and the difficulty of the rule. The negative region contains the rules not related to the decision class, and the positive region includes the rules associated with the class, while the boundary region contains the inconsistent rules. The recognized intents are “plot_instance_FRS” and “instance_difficulty”, respectively.

Furthermore, the user can request the closest rule to the one used to resolve the last query. This is recognized by the intent “closest_instance”, and an example can be seen in Figure 15.

As mentioned, the user can also ask the chatbot counterfactual questions, an example of which can be seen in Figure 16. Rasa recognizes such a question by the intent “run_cf_query”. It extracts the variables that should be regarded as unknown as the entity “unknown_concept”, the variables that

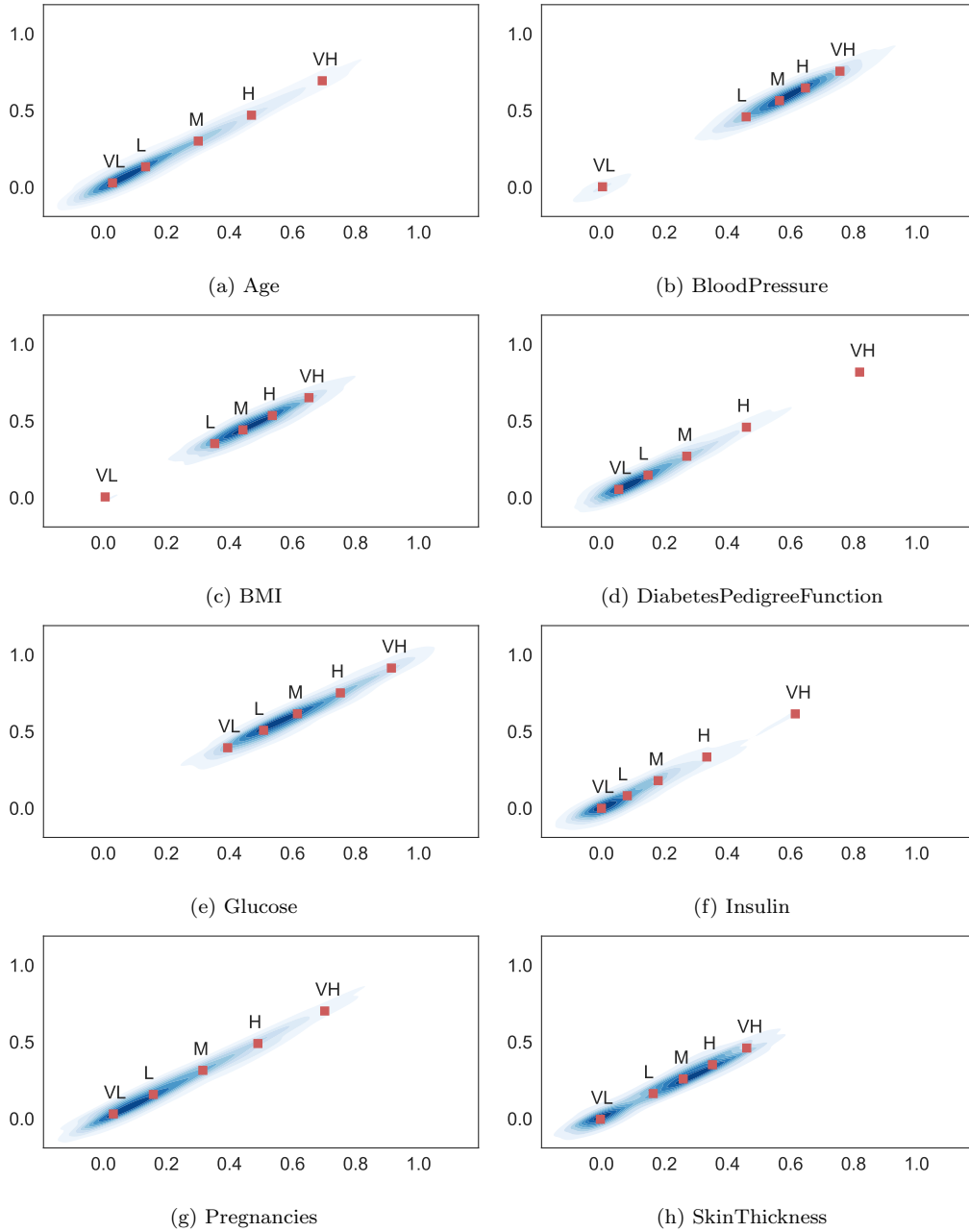


Figure 10: Fuzzy prototypes extracted from numerical features describing the diabetes dataset, which are labeled with predefined linguistic terms.

Can you construct the symbolic explanation module?

Done! I calculated the symbolic terms for the values of the input variables. I created fuzzy-rough regions and built a knowledge base in Prolog to run queries. I would appreciate it if you used the correct feature and class names. As a reminder, the feature names are las, Pres, Skin, Insu, Mass, Pedi, Age, and Class. The outcome classes are tested_negative and tested_positive.

Figure 11: Construction of the symbolic explanation module in the chatbot.

```
input([503,[v1,0.98],[1,0.962],[m,0.988],[1,0.956],[v1,0.962],[v1,0.987],[1,0.965],[v1,0.975],0.954]) :- output([tested_negative,1.0]).
input([504,[vh,0.945],[vh,0.973],[m,0.955],[h,0.945],[v1,0.999],[m,0.956],[v1,0.998],[m,0.972],0.877]) :- output([tested_positive,0.85]).
```

Figure 12: An excerpt from the Prolog knowledge base.

are known as the entity “known_concept”, their values as the entity “value”, and finally, the outcome as the entity “outcome”.

Additionally, the problem complexity can be quantified by the chatbot (see Figure 17). This is recognized as the intent “complexity” and involves the fuzzy-rough regions used to determine the complexity of rules. The intu-

If Preg is very low, Gluc is low, and the outcome is test_negative, what is Age?

I have run the query for you. These are the results: Age is very low with a certainty of 0.991. The entire rule has a certainty of 0.954.

Figure 13: Example of a what-if question resolved by the chatbot.

What difficulty does this rule have?

This instance has a difficulty of 0.046.

Figure 14: Difficulty of the rule used to answer the previous what-if question. The higher the difficulty level, the less confident the answer.

ition is that more difficult problems should have more significant boundary regions, while easier ones should have larger positive regions.

The fuzzy-rough uncertainty [29] associated with a protected feature can also be requested by the user. This measure quantifies the fuzzy-rough regions change when the protected feature is removed, which can be seen as a proxy for measuring explicit bias. An example can be seen in Figure 18. Rasa recognizes this as the intent “bias”, and extracts the entity “variable”, which indicates which feature is to be suppressed.

What rule is closest to this one?

I found that the following rule is the closest: If Preg is very low, Gluc is low, Pres is medium, Skin is low, Insu is very low, Mass is low, Pedi is very low, Age is very low, then Class is tested_negative.

Figure 15: Closest rule to the last one used to resolve a query.

What values should Preg and Gluc take while Age is medium for an outcome to be tested_negative instead of tested_positive?

I have run the query for you. These are the results: Preg is very high with a certainty of 0.929. Gluc is high with a certainty of 0.924. The entire rule has a certainty of 0.883.

Figure 16: Example of a counterfactual question resolved by the chatbot.

Additionally, the user can request the top rules in the knowledge base, as can be seen in Figure 19. The chatbot recognizes this task through the intent “top_rules_kb”. These rules provide valuable knowledge about the problem that domain experts can further inspect.

Before concluding our paper, we should mention that the proposed conver-

What is the complexity of the loaded problem?

The complexity of the classification problem on this dataset is 0.862. This complexity measure is calculated by taking the sum of the membership values according to positive regions, divided by the total number of instances.

Figure 17: Complexity of the loaded classification problem.

What is the explicit bias associated with Age?

The fuzzy-rough uncertainty (bias) of this feature is 0.662. This measure quantifies the changes in the decision boundaries after removing a protected feature as a proxy for bias quantification.

Figure 18: Explicit bias attached to a protected feature.

sational agent can be extended significantly in different ways. For example, we can integrate popular fairness measures to quantify and mitigate bias, include pre-processing pipelines, or incorporate feature attribution metrics as another way of providing interpretability.

What are the top rules in the knowledge base?

These are the top 3 rules in the knowledge base:

Rule #1: If Preg is medium, Plas is low, Pres is low, Skin is high, Insu is medium, Mass is low, Pedi is high, Age is low, then Class is tested_negative.

Rule #2: If Preg is high, Plas is very high, Pres is very high, Skin is medium, Insu is high, Mass is very high, Pedi is medium, Age is high, then Class is tested_positive.

Rule #3: If Preg is high, Plas is very high, Pres is low, Skin is high, Insu is high, Mass is medium, Pedi is medium, Age is low, then Class is tested_positive.

Figure 19: Top rules in the knowledge base.

7. Concluding remarks

In this paper, we presented a reasoning module powered by Prolog to generate symbolic explanations from the predictions made by a black-box classifier. The main procedure of this module concerns the induction of fuzzy rules used to derive the explanations. The main advantages of our proposal are summarized as follows. Firstly, the rules are automatically mined using a data-driven approach that relies on granular computing. Secondly, the explanation module is agnostic and can be coupled with any machine learning classifier. Thirdly, the module computes the confidence of rules as a tool to

assess the correctness of explanations. Lastly, users can pose their questions using natural language queries and receive understandable answers without involving complex theoretical concepts.

The numerical simulations using several pattern classification problems allowed us to draw interesting conclusions about the parameters involved in the fuzzy rule construction. Concerning the number of symbols, we noticed that having more granularity does not always lead to higher confidence values, but it does harm interpretability. It was also noticed that the proposed feature-wise distance function yields better results than the baseline when having less granularity, while the smoothing parameter should be as close to one as possible. Concerning the fuzzy operators, Łukasiewicz and Fodor were the best-performing operators in our simulations.

Within the limitations of our proposal, we can mention the trade-off between precision and significance. On the one hand, more precision will harm interpretability since we need more symbols to describe numerical variables. Less precision will improve interpretability but at the expense of harming the confidence of rules. Another limitation is that experts must define the symbols they want to produce semantically coherent explanations. For example, it would be grammatically odd to describe the variable age in terms of “low age” and “high age” instead of using the linguistic terms “young” and “old”, respectively. Our future research effort will be devoted to inferring such linguistic terms from the feature names.

References

- [1] Agarwal, R., Melnick, L., Frosst, N., Zhang, X., Lengerich, B., Caruana, R., Hinton, G., 2021. Neural additive models: Interpretable machine learning with neural nets, in: *Advances in Neural Information Processing Systems*.
- [2] Apley, D.W., Zhu, J., 2020. Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 82, 1059–1086.
- [3] Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W., 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE* 10, 1–46.
- [4] Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F., 2020. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58, 82–115.
- [5] Bezdek, J.C., Ehrlich, R., Full, W., 1984. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences* 10, 191–203.
- [6] Bocklisch, T., Faulkner, J., Pawlowski, N., Nichol, A., 2021. Rasa open source.
- [7] 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*, pp. 456–461.

- [8] Dandl, S., Molnar, C., Binder, M., Bischl, B., 2020. Multi-objective counterfactual explanations, in: Back, T., Preuss, M., Deutz, A., Wang, H., Doerr, C., Emmerich, M., Trautmann, H. (Eds.), *Parallel Problem Solving from Nature – PPSN XVI*, pp. 448–469.
- [9] d’Avila Garcez, A.S., Besold, T.R., Raedt, L., Foldiak, P., Hitzler, P., Icard, T., Kuhnberger, K.U., Lamb, L.C., Miikkulainen, R., Silver, D.L., 2015. Neural-symbolic learning and reasoning: Contributions and challenges, in: *The 2015 AAAI Spring Symposium Series*.
- [10] d’Avila Garcez, A.S., Broda, K., Gabbay, D.M., 2001. Symbolic knowledge extraction from trained neural networks: A sound approach. *Artificial Intelligence* 125, 155–207.
- [11] De Raedt, L., Kimmig, A., 2015. Probabilistic (logic) programming concepts. *Machine Learning* 100, 5–47.
- [12] Dubois, D., Prade, H., 1990. Rough fuzzy sets and fuzzy rough sets. *International Journal of General System* 17, 191–209.
- [13] 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)*, pp. 80–89.
- [14] Grau, I., Hernandez, L.D., Sierens, A., Michel, S., Sergeysse, N., Froyen, V., Middag, C., Nowe, A., 2021. Talking to your data: Interactive and interpretable data mining through a conversational agent,

- in: Proceedings of the 33th Benelux Conference on Artificial Intelligence and the 30th Belgian Dutch Conference on Machine Learning (BNAIC/BENELEARN 2021), pp. 1–3.
- [15] Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., Giannotti, F., 2018. Local rule-based explanations of black box decision systems. URL: <https://arxiv.org/abs/1805.10820v1>.
- [16] Hatwell, J., Gaber, M.M., Azad, R.M.A., 2020. CHIRPS: Explaining random forest classification. *Artificial Intelligence Review* 53, 5747–5788.
- [17] Hoitsma, F., Knobon, A., Espinosa, M.L., Nápoles, G., 2020. Symbolic explanation module for fuzzy cognitive map-based reasoning models, in: Bramer, M., Ellis, R. (Eds.), *Artificial Intelligence XXXVII*, pp. 21–34.
- [18] Inuiguchi, M., Wu, W., Cornelis, C., Verbiest, N., 2015. Fuzzy-rough hybridization, in: Kacprzyk, J., Pedrycz, W. (Eds.), *Springer Handbook of Computational Intelligence*, pp. 425–451.
- [19] Jayaram, B., Mesiar, R., 2009. On special fuzzy implications. *Fuzzy Sets and Systems* 160, 2063–2085.
- [20] Kumar, I.E., Venkatasubramanian, S., Scheidegger, C., Friedler, S., 2020. Problems with shapley-value-based explanations as feature importance measures, in: III, H.D., Singh, A. (Eds.), *Proceedings of the 37th International Conference on Machine Learning*, pp. 5491–5500.
- [21] Kuźba, M., Biecek, P., 2020. What would you ask the machine learning model? identification of user needs for model explanations based on

- human-model conversations, in: ECML PKDD 2020 Workshops, pp. 447–459.
- [22] Lipton, Z.C., 2018. The mythos of model interpretability. *Communications of the ACM* 61, 36–43.
- [23] Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.I., 2020. From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence* 2, 56–67.
- [24] Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions, in: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 4768–4777.
- [25] Manhaeve, R., Dumancic, S., Kimmig, A., Demeester, T., De Raedt, L., 2018. Deepproblog: Neural probabilistic logic programming, in: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*.
- [26] Montavon, G., Samek, W., Muller, K.R., 2018. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing* 73, 1–15.
- [27] Mothilal, R.K., Sharma, A., Tan, C., 2020. Explaining machine learning classifiers through diverse counterfactual explanations, in: Hildebrandt, M., Castillo, C., Celis, E., Ruggieri, S., Taylor, L., Zanfir-Fortuna, G. (Eds.), *FAT* '20: Conference on Fairness, Accountability, and Transparency*, pp. 607–617.

- [28] Murdoch, W.J., Singh, C., Kumbier, K., Abbasi-Asl, R., Yu, B., 2019. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences of the United States of America* 44, 22071–22080.
- [29] Nápoles, G., Koumeri, L.K., 2021. A fuzzy-rough uncertainty measure to discover bias encoded explicitly or implicitly in features of structured pattern classification datasets. *Pattern Recognition Letters* URL: <https://arxiv.org/abs/2108.09098>.
- [30] Pagouropoulos, P., Tzimopoulos, C., Papadopoulos, B., 2020. A method for the detection of the most suitable fuzzy implication for data applications. *Evolving Systems* 11, 467–477.
- [31] Plumb, G., Molitor, D., Talwalkar, A.S., 2018. Model agnostic supervised local explanations, in: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 2520–2529.
- [32] Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., Bie, T.D., Flach, P.A., 2020. FACE: feasible and actionable counterfactual explanations, in: Markham, A.N., Powles, J., Walsh, T., Washington, A.L. (Eds.), *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society*, New York, NY, USA, February 7-8, 2020, pp. 344–350.
- [33] Ribeiro, M.T., Singh, S., Guestrin, C., . Anchors: High-precision model-agnostic explanations, in: McIlraith, S.A., Weinberger, K.Q. (Eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intel-*

- ligence (IAAI-18), and the 8th AAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), pp. 1527–1535.
- [34] Ribeiro, M.T., Singh, S., Guestrin, C., 2016. Why should i trust you?, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144.
- [35] Roscher, R., Bohn, B., Duarte, M.F., Garcke, J., 2020. Explainable machine learning for scientific insights and discoveries. *IEEE Access* 8, 42200–42216.
- [36] Stepin, I., Alonso, J.M., Catala, A., Pereira-Farina, M., 2021. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access* 9, 11974–12001.
- [37] Tekol, Y., contributors, 2020. PySwip v0.2.10. URL: <https://github.com/yuce/pyswip>.
- [38] 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, 841.
- [39] Zhang, Y., Song, K., Sun, Y., Tan, S., Udell, M., 2019. ”why should you trust my explanation?” understanding uncertainty in lime explanations, in: Proceedings of the ICML 2019 AI for Social Good Workshop.