UNIVERSITY OF AMSTERDAM

Explainable Machine learning using mathematical optimization Rule based methods

June 27, 2024

mentor:
Tabea Röber

Student:
Jakub Waszczak
14081504

Course:
Bachelor's Thesis and Thesis Seminar
Business Analytics

Course code:
6013B0804Y

7450 (errors:2) words

ABSTRACT

This paper examines the performance of rule-based methods using mathematical optimization. As artificial intelligence has grown over the past few years, the demand for more advanced algorithms has also increased. In response, rule-based methods have aimed to improve simplicity and provide clear explanations for complex models. Using mathematical optimization, the main goal is to achieve the highest possible accuracy and interpretability while maintaining low complexity. My study compares different algorithms across a variety of datasets, highlighting the conditions under which each model performs very well and those in which it lacks. I picked frameworks that vary in architecture, rule generation process, and model evaluation to ensure the results provide a comparison between models that are considerably different from each other.

1 Introduction

In recent years, explainable AI (XAI) has gained a lot of attention due to advancements in computing technology and the need for highly interpretable models. Companies nowadays store large volumes of data, and some of the algorithms used in the past for classification purposes may already be outdated and perform poorly (Belcastro et al., 2022). To meet such demands in the IT industry, many new approaches have been developed, tested, and compared to conventional models. However, one of the main issues that often occurs is the accuracy-interpretability trade-off. Having a framework that is very accurate usually leads to a lack of interpretability, whereas a model that is simple to understand often lacks accuracy (Hulsen, 2023). The objective of ensuring transparency and clarity in AI systems has emerged as a fundamental framework in the fields of machine learning, underscoring the need for clarity in the decision-making processes of AI (Zhang et al., 2020). The rule-based approaches, which are a component of XAI, have reached substantial growth due to their high interpretability (Wang et al., 2021). Moreover, most of these models provide satisfying accuracy, which is also a great indicator of an efficient framework to use. Rule methods can be particularly important in areas where understanding the decision logic is essential, such as healthcare or finance. For instance, in healthcare, a simple "if-then" rule might state: "If a patient's blood pressure exceeds 140/90, then prescribe antihypertensive medication." These methods provide clear and useful insights by making data-driven rules. A group of these interpretable models includes optimization-based rule methods. They often use various techniques that have the same goal of providing a well-balanced trade-off between accuracy and interpretability. In the past couple of years, several methods have been proposed, with the main differences usually being the way loss functions, rule mining approaches, or rule classification are used. These methods address different aspects of the model's performance and explainability, reflecting the diverse needs of various applications.

Depending on the task, whether it's a binary or multi-class classification problem, it can be hard to pick the best-performing model without prior knowledge. Testing all the models and selecting one might be a highly time-consuming task, as is usually the case, as well as computational efficiency. The objective of my study is to compare several methods that vary significantly in the manner in which they obtain outcomes. In order to ensure that the results are reproducible, reliable, and unbiased, I will utilize data sets of different sizes and diverse feature types. First, I will measure the classification performance of each model using a few well-known algorithms as a baseline, and next, I will focus on reporting the interpretability metrics, which will assess both the model's performance and the clarity of its outputs. By evaluating the aforementioned metrics, I aim to provide a comprehensive comparison that highlights the strengths and weaknesses of each method.

.

2 Related work

It is very crucial to find a balance between model correctness and interpretability in the field of Explainable Artificial Intelligence(XAI). Mathematical optimization provides a framework for designing models that are both precise and comprehensible. To optimize the model well, this means setting decision variables, a goal function, and constraints (Bertsimas Dunn, 2017). The optimization problem can be formulated as:

$$\min_{R \in \mathcal{R}} \ \ell_D(R) + \lambda\Omega(R)$$

where:

- $R \in \mathcal{R}$: Represents the set of all possible rules.
- $\ell_D(R)$: Denotes the total loss, which measures the discrepancy between the predicted and actual values.
- $\Omega(R)$: Denotes the complexity of the model, which can include factors such as the number of rules or predicates in the rule set.
- λ: A regularization parameter balancing loss and complexity (Rudin, 2019).

This formulation aims to achieve equilibrium between accuracy and maintaining an understandable output. The influence of these two variables and the regularization parameter is crucial in terms of the correct interpretation and explanation of the model. Building upon this, the Interpretable Decision Sets algorithm uses mathematical optimization to construct rule sets that are simple and understandable. To achieve such results, authors implement non-negative λ parameters that control the weights of various components defined from f_1 to f_7. Parameters $\lambda_1 ... \lambda_7$ are estimated using a validation set, which is created from 5% of the entire data set. Cordinate ascent is used to explore the parameter space, determine optimal values, and provide a decision set that achieves the highest AUC. This process also ensures that the interpretability metrics remain within pre-defined bounds. The objective function is expressed as follows:

$$\arg \max_{R \subseteq S \times C} \sum_{i=1}^{7} \lambda_i f_i(R)$$

where each $f_i(R)$ represents a specific parameter λ controls:

- $f_1(R)$: Rewards decision sets with a smaller number of rules, promoting simplicity.
- $f_2(R)$: Rewards decision sets with fewer predicates in the rules, encouraging shorter and more interpretable rules.
- $f_3(R)$: Penalizes overlap between rules predicting the same class, to ensure clearer decision boundaries.
- $f_4(R)$: Penalizes overlap between rules predicting different classes, to further improve clarity.
- $f_5(R)$: Ensures that each class in the dataset is predicted by at least one rule.
- $f_6(R)$: Encourages precision by favoring rules with small incorrect-cover sets.
- $f_7(R)$: Encourages recall by ensuring that each data point is correctly covered by at least one rule.

All of the mathematical optimization techniques mentioned in this paper have one thing in common. Their main target is to increase the interpretability and transparency of the algorithm without additional frameworks. In other words, these algorithms can be referred to as **glass box models**. They provide a high level of transparency in their decision-making process while at the same time having a relatively small number of internal elements (Rai, 2019). Decision trees, Bayesian classifiers, linear regressions, rule-based models, and other machine learning algorithms produce interpretable models because each of their constituent parts, such as a path in the decision tree, the weight of a feature in the linear model, or an already defined rule, can be investigated to understand predictions generated by the algorithm. On the other hand, over the preceding decade, marked by the evolution of novel technologies and the maturation of AI systems, notable progress has been achieved in the refinement of new algorithms. In light of these advancements, several new large language models have been released that have revolutionized AI usage in day-to-day life. In conjunction with those factors, algorithms became significantly more complex and accurate than glass-box solutions. In response, **black box models** emerged, leveraging complex algorithms and deep learning techniques to handle the intricate patterns and nuances within the data.

However, in order to achieve the mentioned accuracy, models needed to be considerably more advanced. Therefore, we must establish a trade-off between these two metrics. To achieve this precision, there was a noticeable decrease in their transparency, which caused a drawback in how the data was processed. A constrained understanding of the internal decision-making procedures makes algorithms more prone to potential interruptions that could cause them to malfunction. From an ethical standpoint, decisions made through opaque systems are considered unacceptable, particularly regarding the evaluation of human resources (Lampathaki et al., 2020). To handle the trade-off between accuracy and interpretation of black boxes as well as deep learning models, post-hoc explanations became useful in handling given data. This XAI method takes a trained AI model as an input, and afterwards, it produces comprehensible representations in the form of rule sets, visualizations, decision boundaries, or numerous others (Guidotti et al., 2018). To have a reliable and trustworthy prediction, some of the models create feature importance rankings (Chang et al., 2024), which aid in determining which characteristics most significantly affect forecasts.

Within the context of post-hoc explanations, there are two distinct approaches: model-specific and model-agnostic procedures. These approaches provide valuable insights into the internal mechanisms of machine learning models. Model-agnostic explanations are often used to interpret black-box models; however, they are not inherently correlated with them. It's main task is to describe ML models without relying on the details of the model structure or parameters. These explanations focus on understanding the model's behavior solely based on its input-output relationship (Trumbelj et al., 2010), indicating that by observing the input variables, one can directly observe how the model reacts. LIME or SHAP are two of the most widely used methods. Contrarily to agnostic interpreters, model-specific explanations may require access to internal details of the model, such as architecture, parameters, or weights. They explain how specific algorithms work by utilizing reverse engineering, they deliver explanations on how particular algorithms work (Padmanabhuni, 2022). The benefits of specific models include a better understanding of the choice since we know how the model operates internally. However, the disadvantage of these models is that they require a thorough examination of the entire model structure, which can compromise performance because the model must be recreated. One of the most widely recognized examples includes decision tree induction and feature importance analysis for linear models. These days, making AI estimates for crucial tasks is not enough anymore. Models must possess a high degree of explainability and interpretability, ensuring that the processes by which they operate are transparent and comprehensible. XAI techniques can be divided into two categories: global ones, which provide a broader understanding of a model's behavior across the entire dataset, and local explanations, which aim to describe the specifics of a prediction that requires clarification (Mahya, 2023). While local explanations focus on particular instances or predictions, global ones aim to describe the overall logic of the model. Their primary focus is on explaining each prediction within the context of the entire dataset (Saleem et al., 2022). One of the primary examples, which will later be used as a baseline in method comparison, are decision trees. Based on their model architecture and parameters, users can split input features at each node and later analyze particular decision paths, while also understanding why specific decisions were made.

2.1 Rule-based methods

In the realm of machine learning and computational systems, rule-based methods have developed as essential solutions to highly complex problems. The fundamental concept behind these methods involves the use of explicit sets of logical rules to guide decision-making processes, rendering them particularly appealing for applications where interpretability and transparency are critical.

The subject of interpretable models has an extensive background. One of the earliest approaches dates back to 1975, when Professor Lotfi A. Zadeh invented fuzzy rule-based systems (Türkşen, 2010). Specifically, Mamdani-type FRBS, as they are commonly referred to, have two primary elements: **the fuzzy system and fuzzy knowledge base (KB)**. The fuzzy system processes the inputs using fuzzy logic to generate the output. It ought to be pointed out that fuzzy logic is a form of logic that bases its reasoning on approximation, with values ranging between 0 and 1. On the other hand, knowledge base corresponds to the knowledge about the concept that is being solved. KBs use IF-THEN rules made up of linguistic variables (Cordón, 2010), which are variables described by words rather than numerical values. Its implementation allows humans to easily understand systems based on their simple structure and clear output definition.

However, recent endeavors over the past decades have aimed to develop machine learning models that provide not only interpretability but also very high accuracy. A prominent example of such a model is the **decision tree classifier**. It is a tree-like method in which each path starting from the root is defined by a sequence of data divisions until a Boolean result is reached at the leaf node (Jijo et al., 2021). Each node represents a decision that has been made based on a given feature, and each leaf node refers to a prediction or classification. It is one of the models that can logically interpret results through their **if-then** rules (Mahbooba, 2021), which are easily interpretable by humans without the need for complex equations, aligning them closely with rule-based methods. Due to their ability to provide clear explanations, decision trees are appropriate models to use in situations where there is a demand for an explanation of the prediction rather than the prediction itself.

The **CART** algorithm is one of the various forms of classification tree algorithms. Based on statistical measures, it operates by iteratively dividing the data into subsets until a specified criterion is satisfied. As input, CART takes a dataset whose measurement vectors can include variables with values such as boolean, integer, or real (Crawford, 1989). The process of constructing the tree starts at the root node, which involves the whole sample of the set. The CART algorithm scans for the best possible instance to divide one node into two separate child nodes. Several techniques are employed to minimize the search time needed to explore all potential divisions. The "Gini" index is one of the most well-known splitting functions, which measures data purity (Lewis, 2000). The goal is to minimize the Gini index at each node and select the split that reduces impurity the most.

Another approach is the **C4.5** algorithm, which, like the CART algorithm, splits data into subsets to improve decision-making. Like CART, C4.5 starts at the root node and recursively partitions the dataset into child nodes based on feature values. While CART uses the Gini index to split the tree, C4.5 uses the **information gain ratio**, which is also one of its main advantages (Dai et al., 2014). This method prevents the model from selecting attributes with many different values, such as age or customer number. Because of its efficient classification and great precision, people commonly use the C4.5 algorithm (Sharma et al., 2013).

Nevertheless, researchers have made numerous improvements to the described model, resulting in an extensive amount of variation. One of them is the proposed **VFC4.5** (Cherfi et al., 2018). The author adopts a mean and a median as *cut-point* criteria for how the data set should be split. It can be very beneficial when dealing with imbalanced data sets. Especially when extreme values are significant, particularly because they represent a minor class, applying the mean can guarantee that these values are taken into account throughout the split.

Building on these developments, a notable accomplishment in decision tree algorithms is **C5.0**, an improved iteration of the extensively utilized C4.5. Developed by Ross Quinlan, C5.0 includes numerous new features compared to its previous version, making it a very efficient tool for classification tasks. This improvement is due to the fact that fewer trees need to be generated in its newer version. Overall, C5.0 has lower error rates (Pandya et al., 2015), higher accuracy, and better processing time. However, the **Chi-squared Automatic Interaction Detector (CHAID)** is another way to classify data, especially categorical data, and C5.0 makes big improvements to decision tree algorithms. Compared to the previously mentioned classifiers, which are based on information gain ratio, this one uses the chi-squared statistic to determine the best split for categorical variables. The primary goal behind the CHAID algorithm is to divide the samples in the best way possible based on the target variable and the chosen feature index. By evaluating the significance of the chi-square test, it automatically makes decisions regarding the partitioning process (Yang et al., 2023).

Decision trees and rule-based methods are intrinsically connected, as both depend on a systematic approach to decision-making through explicit rules. In a decision tree, each path from the root to a leaf node represents an intersection of conditions (or rules) that lead to a final classification. On the other hand, rule methods use IF-THEN statements without the hierarchical constraints of a tree, allowing for a more versatile and sometimes more efficient decision-making process. This transition from tree-based models to rule-based methods highlights the primary principle of providing clear and interpretable insights, which are crucial in domains that need a high level of transparency and comprehension of decision-making logic. In aiding interpretation, rule lists, defined as rules connected using IF-THEN-ELSE statements (Yang et al., 2022), usually offer simplicity when it comes to understanding the given output. Their structure allows for potentially fast evaluation, as it stops at the first matching rule. Moreover, the decision-making process is quite simple when the order of rules is important. Nevertheless, there are a few disadvantages to rule lists. The initial point to be noted is the sequential evaluation. If there are multiple rules that could potentially apply to a specific scenario, using rule sets would be a more efficient approach. More about the pros and cons of rule sets will be described in the later part of my work.

One of the commonly used induction algorithms that simplifies rule list usage is **CN2**. This method, albeit original, is based on fundamental concepts drawn from the ID3 and AQ algorithms (Clark et al., 1989). Thus, it signifies a progression rather than a drastic deviation from these established techniques. The model is based on two primary components: an algorithm that does the search to identify the most optimal rules, and a control algorithm that executes the search process periodically. It is recommended to assess the quality of the rules found by checking the accuracy of the training data. However, one of the issues associated with this metric is that they have the propensity to favor highly complex rules that cover only a small number of examples. This is because the probability of finding rules with high accuracy on the training data improves as the rules become more specific.

While CN2 can be defined as a significant improvement in rule-based methods, it is not really effective when dealing with complex rules. To target these issues, researchers have developed numerous algorithms that can handle such obstacles. One of the proposed approaches is **CORELS(Certifiably Optimal Rule Lists)**. A special branch-and-bound algorithm creates rule lists for categorical data, ensuring optimal performance. Before CORELS starts the main process, it pre-mines frequent item sets from the data set, which guarantees that the algorithm will start working with a pre-defined set of rules. Only then will CORELS initiate the process of searching for the best rules. First, it creates an empty rules list and places it in the priority queue. In the next steps, CORELS repeatedly generates new lists and calculates a lower bound for the objective function. The lower bound, in this case, acts as a threshold, which says if the rule list should be kept or pruned.

The priority queue serves as a storage space to store and manage the exploration of already-created rule lists. When the lower bound of a particular rule list surpasses the objective value of the best rule list, that rule list is pruned. This process is repeated until all rule lists have been checked and the best one is defined as the optimal one. Moreover, CORELS provides a certificate of optimality, ensuring that it is the best possible solution.

The model proposed by Dash et al. (2018) introduces a novel way to enhance rule-based methodologies. It utilizes Boolean decision rules defined as an integer program (IP) and solves them using column generation (CG). The objective of this strategy is to find a balance between classification accuracy and the simplicity of rules. The approach starts with an integer program that aims to reduce Hamming loss, considering both false positives and false negatives. Constraints guarantee that the level of complexity in the rule set does not surpass a predetermined limit. The column generation framework proceeds by iteratively generating rules in order to enhance the solution. This particular process starts by selecting a limited number of rules, solving a simplified master problem, and subsequently including additional rules that reduce the objective value. This procedure continues until no other improvements can be made.

However, there are a few limitations associated with algorithms generating rule lists. The main drawback that makes them inefficient in a wide range of applications is their architecture. In decision lists, each subsequent rule is contingent on the failure of all preceding rules. Such that later rules may not consider all relevant information, which might lead to biased predictions. Moreover,

as the number of rules in a list grows, interpreting them all becomes a challenging task, potentially leading to misinterpretations. While both rule sets and rule lists are generally interpretable, decision sets can occasionally offer greater clarity in specific circumstances. The structure of rule sets facilitates comprehension of how different rules interact and contribute to the final decision. More specifically, the process of analyzing a rule in a decision list necessitates understanding all of the preceding rules as well, because each rule is applied only when none of the preceding rules have been triggered. Decision sets, do not have this issue and hence provide better interpretability.
One of the suggested algorithms is so-called **IDS(interpretable decision sets)** created by Lakkaraju et al. (2020). Their approach relies on sets of classification rules, all of which are unordered. While decision lists may appear very similar in structure, it is important to note that the rules are not connected by "ELSE" statements, which makes each rule an independent predictor capable of assigning its own label without considering other rules. This method discovers sets that are globally close to optimal and, on average, almost six times smaller than a decision list trained on the same data set. To achieve such a score and ensure comprehensive understanding as well as interpretability, numerous factors must be considered when constructing a model. One of the key factors influencing the model's understandability is its complexity, which has a significant impact afterwards. The author highlights the main metrics to define interpretability: size, length, cover, and overlap. Firstly, **size** defining the number of rules in the set. A decision set, which is characterized by clarity and ease of understanding, should ideally consist of the smallest possible number of rules. Second, we look at the **length** of the rules in a decision set. Logical functions are typically easy for humans to grasp, as Donald A. Norman discusses in his exploration of complexity (Norman, 2010). While these functions can simplify item sets, having too many predictors in a rule can make it harder to understand. This issue is highlighted by the concept of complexity bias discussed on Farnam Street (2024), which suggests that people prefer a moderate level of complexity—too much can be confusing and reduce clarity. We refer to the measure of a rule's size as its length. To evaluate the conciseness of a decision set, the author focuses on two key properties, which are essential components of interpretability. Moreover, for a decision set to be interpretable, it must clearly delineate the decision boundaries for each class within the dataset. One of these properties measures which points in the data set are covered. This refers to the set of data points in the dataset that satisfy the conditions specified by each rule, irrespective of their observed labels, and measuring **cover** helps determine how the dataset is partitioned by the rules. This assessment is performed on a per-rule basis, determining which data points satisfy the criteria set by each rule. In addition to cover, **overlap** is another important property that affects interpretability. It measures whether decision boundaries are clearly defined by inspecting rule overlap. When rules overlap each other at a single data point, a tie-breaking function must be used to make a prediction, which reduces interpretability. Therefore, it is essential to pick rules that overlap as little as possible, ensuring that the user can clearly understand the boundaries created by the model.

On the other hand, model proposed by Yang et al. (2022) utilizes a submodular optimization technique to obtain interpretable decision rule sets. The main task is characterized as a problem of selecting a subset, where the goal is to choose a subset of potential rules that maximizes a combined measure of accuracy and interpretability. Such that, the objective function for selecting the best rules is expressed as a difference of two submodular functions. This formulation, known as a difference of submodular (DS) optimization, allows the problem to be approximately solvable by existing DS optimization algorithms. The optimization process involves two main tasks: first, selecting features to create rules, and second, selecting rules to compose the final rule set. Rather than depending on a predetermined set of rules, the approach dynamically generates rules while optimizing. This technique of rule generation ensures that the learning algorithm is able to explore a diverse set of rules and optimize the main function with more efficiency.

As the complexity of the dataset rises, models might become overly complex and difficult for humans to interpret. The **PRISM** algorithm attempts to bypass these issues by using an alternative induction strategy to induce modular rules (Cendrowska, 1987). It employs a rule-based approach to identify predictive rules and subsequently specifies subgroups of data points with unique characteristics or traits. These rules help partition the data into meaningful subsets. After the rule generation process, PRISM evaluates each rule based on its performance and statistical significance. Only those rules that meet predefined criteria for predictive accuracy and significance are retained for further analysis. In essence, PRISM's main target is to produce interpretable and transparent outputs by generating patterns within the data. This method aligns well with the concept of rule sets, as it generates a collection of rules that can be applied independently to classify data points, enhancing interpretability and manageability in datasets with a lot of features.

Building on the advancements in model interpretability, another development in the realm of rule-based methods is the rule-based optimization method called **RuleOpt** for classification introduced by Lumadjeng et al. (2023). Scalability, interpretability, and fairness are the main metrics that the authors aim to improve with the implementation of this rule-based algorithm. Furthermore, the core feature of the framework is its adaptability to integrate rule costs and other constraints while ensuring scalability. In order to effectively manage large datasets, algorithms use linear programming (LP) combined with column generation (CG). It formulates the problem as an LP, with each rule being represented as a column in the LP model. The main objective is to minimize the total classification loss as well as the overall cost of utilizing the rules. The CG protocol iteratively generates new rules by solving a restricted master problem (RMP) and a pricing subproblem (PSP). The reason for solving the PSP is to identify columns with negative reduced costs. Such columns can improve the objective function value when they are added to the column pool. When the PSP stops returning rules with negative reduced costs, it means that the optimal solution to the master problem has been obtained with the current set of rules. Theoretically, having a minimal number of rules should be better and easier to understand when using a model based on global interpretability (Linardatos et al., 2021). However, this assumes that a single set of rules can explain all decisions, which is not suitable for individualized cases. For example, people who were denied loan approval may need personalized reasons for rejection to understand how to improve their credit score. Furthermore, students receiving grades might benefit from detailed feedback related to their performance, allowing them to prioritize areas needing improvement. Regarding this matter, RuleOpt is mainly focused on local interpretability due to its dual-variable implementation and heuristic approach. Dual variables obtained from solving RMP allow us to identify the most impactful rules for every individual sample based on the importance of each rule. Furthermore, decision tree-based heuristics help in generating rules that are relevant to specific data points, enhancing the ability to provide localized explanations.

The described algorithm offers numerous significant advantages in terms of interpretability by assigning optimal weights to rules. It also includes fairness measures to ensure all protected groups are treated equally, where protected groups are characterized by attributes such as race, religion, age, and more. To achieve even treatment among all groups, authors introduced **Disparate Mistreatment per Class (DMC)**, which balances error rates within each class across different groups. Moreover, **Overall Disparate Mistreatment(ODM)** ensures balanced miss-classification rates among all groups, which leads to providing overall fairness. Despite challenges like solving the PSP and balancing accuracy with fairness, RuleOpt's scalability and flexibility make it a valuable tool for generating interpretable and fair classification rules.

One significant difference between IDS and other approaches like RuleOPT, which will be introduced later, is the approach to minimizing rule overlap. While the main focus of IDS is on creating non-overlapping rules to achieve greater interpretability, RuleOpt optimizes the overall decision set for accuracy and interpretability using hierarchical structure, which is implemented through weights assigned to particular rules. This feature can be very helpful in the decision-making process with complex and large datasets by ensuring that more general rules are applied first, followed by more specific ones.

Handling a large amount of data can be a significant bottleneck for many algorithms. One of the approaches for multiclass classification that addresses this issue was proposed by Proenca and van Leeuwen (2019) and is based on **Minimum Description Length principle(MDL)**. By minimizing the description length, the model compresses the data, highlighting only the most valuable patterns. This process is supervised using the greedy search CLASSY algorithm, which returns a probabilistic rule list. First, it drops redundant patterns to compress the dataset. The rule list starts with the default rule as a baseline. CLASSY repetitively searches for the best rules to add to improve overall compression and stops its iteration where there is no rule that contributes to the compression as a positive gain. Once the process ends, the model produces a rule list. One of the main benefits is its high interpretability across complex and noisy datasets, because the MDL principle compresses both the rule list and the data it explains. Moreover, due to the model's structure, which includes probabilistic distribution, it provides insights into the certainty of the produced rules which, for instance, IDS does not provide. In addition, MDL outperformed IDS on numerous data sets that included a large number of instances. However, sometimes models provided relatively small set of rules compared to the size of the data set, which some users may consider a drawback. Nevertheless, MDL is a tuning-free algorithm. It does not require any hyperparameter optimization, making it a user-friendly model for users without access to efficient computational power.

2.2 Evaluation Criteria

2.2.1 Interpretability and completeness

The difficulty in developing explainable AI comes from providing explanations that strike a balance between interpretability and completeness (Gilpin et al., 2019). The objective of completeness is to accurately represent the functioning of a system. However, it is a challenging task to provide a model that is complete and interpretable at the same time. It should be unethical to provide a simplified representation of an intricate problem with the intention of fostering trust, especially if users are unable to comprehend the constraints of the reduced explanation. In order to prevent one from falling into this pitfall, it is important for explanations to strike a balance between being understandable and complete. Systems should prioritize giving detailed and thorough explanations, even if it means sacrificing interpretability.

2.2.2 Explainability

Whereas interpretability is preoccupied with elucidating the way in which an internal process was arrived at, explainability's primary objective is to generate straightforward outputs that enable seamless human interaction (Vilone et al., 2021). At the outset of each examination, explanations should be divided into two dimensions: local versus global explanations, as previously discussed in the paper, and those dependent on time. Explanations based on time should be distinguished into three phases: prior (before the forecast), contemporaneous (during the prediction), and post (following the prediction) (McDermid et al., 2021).

2.2.3 Fairness

In discussing fairness within the statistical framework, it is essential to distinguish between two fundamental concepts: fairness at the group level and fairness at the individual level (Alikhademi et al., 2021). According to its statistical (group) idea, it is essential to ensure equal treatment among smaller and bigger groups as a collective entity. However, the concept of statistical fairness is constrained by its requirement for unambiguous and well-defined groups (Binns, 2019). In other words, it is necessary to clearly define categories such as demographic groupings (such as race and gender) or outcomes (such as approved versus denied loan applications) in order to determine if fairness standards are being met. When categories are unclear or overlap, it becomes difficult to accurately use statistical fairness measures.
Conversely, individual fairness guarantees that individuals who are deemed 'similar' in relation to the categorization task will be assigned comparable outcomes. Decisions made by AI should be unbiased (Vilone et al., 2021) and treat all individuals equitably, irrespective of their history or any other personal characteristics.

2.2.4 Accuracy

Overall, the accuracy levels of the systems play a crucial role in determining user trust. As the accuracy increases, so does the level of trust from the user (Papenmeier et al., 2019). It assesses the extent to which the explanations align with the processes or features that influence the model's outputs, enabling users to trust and comprehend the behavior of the AI system.
The fidelity of explanations has also had a significant impact. Depending on the accuracy of the model, for systems with intermediate accuracy, a high-fidelity explanation does not negatively reduce user trust, whereas a low-fidelity explanation does. This is because high-fidelity explanations

demonstrate a higher level of accuracy. On the other hand, in a system that has a high level of accuracy, any kind of explanation, regardless of whether it is extensive or simplified, leads to a loss of trust.

3 Methods

In this section, I describe the metrics I will use to evaluate the models included in my study. At the beginning of my research, I had collected nine algorithms. However, after numerous attempts to implement them on the datasets, I could not include some of them in my study. Algorithms from the papers by Yang et al. (2022) and Dash et al. (2018) had numerous bugs in the source code that were unsolvable without the developers' expertise. Ultimately, I decided to use RuleOpt, MDL, IDS, and CORELS.

IDS The Interpretable Decision Sets (IDS) framework relies on a collection of unordered rules. To ensure proper understanding and great interpretability, I computed my study using interpretability metrics provided by the IDS framework, taking into account the previously mentioned size, length, cover, and overlap measures.

- **Fraction Overlap**: This metric captures the extent of overlap between every pair of rules in a decision set. Smaller values indicate higher interpretability.

$$Fraction\ Overlap(\mathcal{R}) = \frac{2}{|\mathcal{R}| \cdot (|\mathcal{R}| - 1)} \sum_{r_i, r_j \in \mathcal{R}, i<j} \frac{overlap(r_i, r_j)}{N}$$

- **Fraction Uncovered**: This metric measures the fraction of data points not covered by any rule in the decision set.

$$Fraction\ Uncovered(\mathcal{R}) = 1 - \frac{1}{N} \sum_{r \in \mathcal{R}} cover(r)$$

- **Average Rule Length**: This metric calculates the average number of predicates that must be evaluated to apply a rule in a decision set.

$$Average\ Rule\ Length(\mathcal{R}) = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} length(r)$$

- **Fraction of Classes**: This metric measures the fraction of class labels that are predicted by at least one rule in a decision set.

$$Fraction\ of\ Classes(\mathcal{R}) = \frac{1}{|C|} \sum_{c \in C} (\exists\, (r, s, c) \in \mathcal{R} \mid c = c')$$

where:

- **N**: The total number of data points in the dataset.
- **r**: A single rule in the rule set $\mathcal{R}$.
- $\mathcal{R}$: The entire set of rules in the decision set.
- **C**: The set of all classes in the dataset.

MDL Due to the structure of rules generated by the Minimum Decision Length model, I was able to use interpretability metrics provided by the IDS authors. In addition, MDL does not require any additional hyperparameter tuning, meaning that all obtained values are considered final, excluding classification performance scores. Since the model is built using the *sklearn* library, I was able to tune those metrics using GridSearchCV.

RuleOPT The performance evaluation of this method is determined using relatively distinct measures compared from those used in IDS and MDL. Nevertheless, all of the investigated algorithms have two common metrics, such as rule set length and average rule width. At the same time, RuleOPT has the unique feature of assigning weight to each rule, providing a clear indication of how valuable a rule is. I did not include this metric in my study due to its absence in other algorithms; however, it is a useful feature to mention. Below are all the remaining metrics the algorithm generates to evaluate performance:

- **Number of Rules (NoR)**: The total number of rules generated by the model. This metric helps in assessing the complexity and manageability of the rule set.
- **Average Rule Length (Avg.RL)**: The average number of conditions in the rules. This metric is used to evaluate the simplicity of the rules generated.

To perform rule classification, RuleOPT uses RUGClassifier and so on it implements plenty of different parameters to effectively regulate the process.

- **Max Depth**: Controls the maximum depth of the decision trees. Common values used are {3, 5, 7, 9, 11, 15}. This parameter is critical for managing the trade-off between model complexity and performance.
- **Penalty Parameter (pen_par)**: Values used are {0.1, 1.0, 10.0} to control the penalty during rule generation. This parameter helps in balancing the accuracy of the model with the simplicity of the rules.
- **Maximum RMP Calls (max_RMP_calls)**: The number of iterations for the RUG algorithm, with values {5, 15, 30}. This parameter ensures the computational feasibility of the rule generation process.
- **Complexity Parameter (C)**: Used for controlling the complexity of the model, with values set around the reported mean for the dataset. This parameter influences the sparsity of the generated rules.
- **Running Time Limit**: Set to 300 seconds for all methods to ensure computational tractability.

CORELS Since this algorithm is a bit outdated and does not use modern metrics, I could only generate a rule list and evaluate metrics related to performance classification.

Baseline As a baseline for my computational study, I have used the RandomForest, C4.5, and CART algorithms. Each of them was implemented with the default parameter settings without any modifications. For more information, please refer to my repository.

3.1 Data

For the evaluation of rule-based methods, a diverse range of datasets was selected. These datasets vary in terms of the number of data points, characteristics, and levels of imbalance. Binary data sets main characteristic is using only two different values in a target variable. Moreover, I decided to pick multi-class classification datasets with variety of classes in prediction column to have a more reliable and precised computation. Imbalance level refers to the disproportion of classes in a dataset, where one class has significantly more instances than others. This can make models ignore the minority class, hence addressing the imbalance is necessary to make accurate forecasts. For more detailed information, please refer to Table 1, which contains all the required information about the datasets

Binary Classification Datasets:

- compas
- titanic
- tictactoe
- ionsphere
- sonar
- cells
- liver

Multi-Class Classification Datasets:

- glass
- doctor
- cars
- beans

Dataset	# of Data points	Features	Classes	Imbalance Level
compas	7214	12	2	0.452
titanic	1761	3	2	0.34
glass	213	9	6	0.31
doctor	714	14	4	0.5
cars	1728	6	4	0.66
tictactoe	958	9	2	0.31
ionsphere	315	34	2	0.28
beans	13611	16	7	0.22
sonar	207	60	2	0.072
cells	699	9	2	0.31
liver	345	104	2	0.16

Table 1: Dataset Information

3.2 Implementation

In this section, I detail the process and methods used to implement and evaluate the rule methods. This includes the train/test split procedure, hyperparameter tuning techniques, hardware and software specifications, time constraints for methods, and the specific parameters tuned for each algorithm.

3.2.1 Hardware and Software

The experiments are conducted using the following hardware and software setup:

Hardware

- CPU: Apple M1
- RAM: 8 GB
- GPU: -

Software

- Operating System: macOS Monterey 12.6.3
- Programming Language: Python 3.10.14
- Development environments: Visual Studio Code, Google Colab

3.2.2 Train/Test Split

To evaluate the performance of chosen models, I divided the data set into training and test set with the ratio of 80/20. Meaning that, as a training set, I picked 80% of the original data set.

3.2.3 Hyperparameter Tuning

To tune the algorithms, I decided to use **Grid Search Cross-Validation**. Several parameters work particularly well with rule-based methods and can significantly increase the performance and accuracy of the model.

RuleOPT

- rule_cost: [Gini(), Length()]
- *max_rmp_calls:* [3, 6]
- *max_depth:* [2, 4, 10]
- *min_samples_split:* [2, 10]
- *min_samples_leaf:* [1, 4]
- *threshold:* [1e-4, 1e-2]
- *max_features:* ['auto', 'sqrt', 'log2']
- *max_leaf_nodes:* [None, 10, 20]
- *min_impurity_decrease:* [0.0, 0.2]
- ccp_alpha: [0.0]

IDS As previously mentioned, Interpretable Decision Sets is based on tuning parameters in the range of λ_1 to λ_7. By default, the algorithm searches for the best parameters between 1 and 1000. To reduce computational time, I decided to reduce the search space to 400, which significantly improved the runtime.

4 Results

In this section, I have collected and described the results of my computational study when comparing rule-based methods. Results are summarized in the tables, including performance and interpretability metrics. Additionally, models that could be hyper-tuned were optimized using predefined parameters.

Table 2 provides a classification of the performance metrics for selected models and benchmark algorithms. In most of the data sets, the Random Forest Classifier consistently scored the highest among other methods, even without hyper-tuning its parameters. However, when considering models excluding baselines, RuleOPT, CORELS, and MDL tended to score comparable results. Moving forward, Table 3 presents the AUC ROC curve scores as part of the performance metrics. The AUC ROC Curve offers insights into the trade-off between true positive and false positive rates for different models across various datasets.

Next, I examined the interpretability metrics in Table 4, which provide detailed information about the values obtained from the generated rule sets for each method. The first tested model was a method using the IDS algorithm. The best results for fraction overlap were obtained on a multi-class classification data set with a 50% imbalance level. Other data sets, both small and medium, performed similarly. Dataset "beans" could not be implemented due to limited computational power. Among most of the data sets, the value of one was obtained in the fraction classes metric, which is desirable as it indicates that all classes are represented by at least one rule in the rule set. Exceptions to this were the *cars* and *glass* datasets. On the other hand, using *glass* and *tictactoe* datasets, IDS generated the longest rules on average. It tends to generate more complex rules as the data set contains more data points. When taking into consideration how many rules are included in the set, the number of instances did not have that much influence. In an alternating manner, small and medium-sized datasets generated a larger number of rules in the rule set.

MDL was the next model to be evaluated using interpretability metrics, and the results can be seen in Table 5. Based on fraction overlap values, larger data sets scored better, indicating that overlap between the rules is higher. Moreover, the percentage of instances not covered by any rule was the lowest in the largest of the tested data sets. During the study, there was an issue with computing fractions of classes, so I decided to implement only the results I am confident are unbiased. The next metric is the average rule width. As expected, larger datasets generated more complex rules due to the greater number of features.

Metrics that allow comparison of all three models simultaneously are average rule width and rule set length and are presented in Table 1 and Table 2. IDS could not run on the largest dataset due to insufficient virtual RAM to process the code. Interestingly, RuleOPT was able to generate fewer rules compared to the MDL method. On average, RuleOPT's rule set size varied somewhat across different dataset sizes. Due to the principles of the MDL model, it created the smallest rule sets, except for the beans dataset. When considering average rule width, each of the three models exhibited similar variation in the length of the rule, regardless of the dataset size.

After evaluating interpretability metrics, I performed parameter tuning on selected algorithms. Table 6 summarizes the lambda values obtained after tuning, and Table 7 shows how these values affected the interpretability metrics of the IDS model. In three out of four cases, fraction overlap was reduced after tuning. Likewise, in all cases, the length of the rule set decreased by almost 50% or more. However, the fraction of uncovered data points by any rule increased significantly in each dataset. The average rule width did not change significantly after tuning, and the fraction of classes metric improved in only two datasets.

Table 8 demonstrates the parameters obtained after tuning by RuleOPT. The highest value for levels in the tree was obtained by the *cars* dataset, while only the *cells* dataset achieved an integer value of 10 for the maximum number of leaf nodes. The threshold was found to be optimal for each dataset at the same value. The overall best score was highest for the *cells* dataset.

Metric	Dataset	MDL	RuleOPT	COR	CART	C4.5	RNF
Accuracy	cells	0.94	0.94	0.92	0.94	0.92	**0.97**
	cars	0.91	0.82	-	**0.96**	0.86	**0.96**
	glass	0.61	0.66	-	0.72	0.34	**0.83**
	tictactoe	0.97	0.96	0.76	0.93	0.95	**0.98**
	titanic	0.84	**0.85**	0.76	0.84	0.84	0.84
F1	cells	0.93	0.92	0.90	0.95	0.91	**0.96**
	cars	0.79	0.78	-	**0.87**	0.60	**0.87**
	glass	0.41	0.67	-	0.69	0.21	**0.86**
	tictactoe	**0.97**	0.95	0.84	0.89	0.86	0.95
	titanic	0.71	0.74	**0.84**	0.71	0.70	0.71
Precision	cells	0.94	0.94	0.83	0.95	0.91	**0.96**
	cars	0.82	0.77	-	**0.87**	0.61	0.85
	glass	0.39	0.64	-	0.72	0.26	**0.91**
	tictactoe	**0.98**	0.94	0.74	0.92	0.86	0.96
	titanic	0.78	**0.82**	0.73	0.77	0.77	0.77
Recall	cells	0.93	0.91	**1.00**	0.94	0.91	0.96
	cars	0.83	0.81	-	0.90	0.62	**0.92**
	glass	0.48	0.71	-	0.72	0.25	**0.84**
	tictactoe	0.96	0.95	**0.97**	0.88	0.86	0.94
	titanic	0.68	0.71	**0.99**	0.67	0.68	0.68

Table 2: Performance comparison across datasets with accuracy, F1 score, precision, and recall

Method	**AUC ROC**		
	cells	**glass**	**cars**
IDS	**0.98**	0.63	0.84
MDL	0.92	0.62	**0.93**
RuleOpt	0.85	0.89	**0.98**
CART	0.92*	0.89	0.95*

Table 3: AUC ROC comparison across datasets

** results were calculated using weighted average across all the classes*

	Frac. Overlap	**Frac. Uncovered**	**Frac. Classes**	**Avg. Rule width**	**Rule Set Length**
titanic	0.025	0.01	1.0	2.15	13
sonar	0.0067	0.3810	1.0	2.0	**25**
cells	0.1583	0.2429	1.0	2.89	18
doctor	0.0	**0.9790**	1.0	1.95	22
cars	0.07	0.44	0.5	2.13	15
glass	**0.0281**	0.7442	0.5	**2.93**	14
ionsphere	0.07	0.58	1.0	1.89	19
tictactoe	0	0.64	1.0	3	17

Table 4: IDS model interpretability metrics

	Frac. Overlap	**Frac. Uncovered**	**Frac. Classes**	**Avg. Rule width**	**Rule Set Length**
titanic	**0.47**	0.30	1.0	1.6	5
cells	0.30	0.64	-	1.33	6
cars	0.20	**0.70**	-	**4.25**	16
glass	0.18	0.5748	-	1.6	5
tictactoe	0.08	0.44	-	2.7	10
beans	0.20	0.10	1.0	2.43	**30**

Table 5: MDL model interpretability metrics

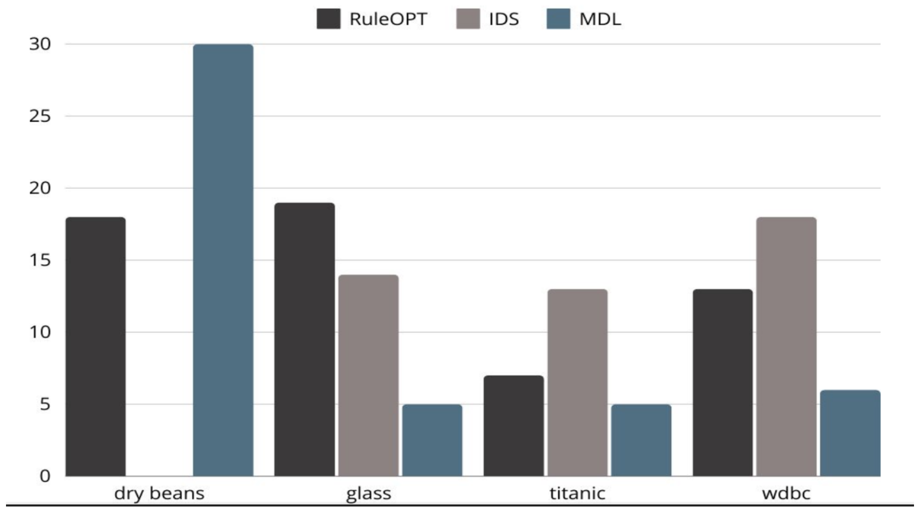

Figure 1: Rule set length comparasion

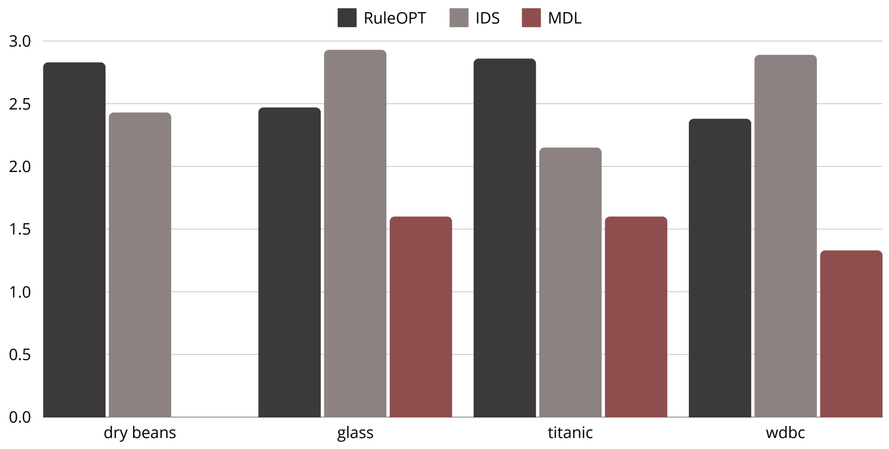

Figure 2: Average rule width comparasion

	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
cells	322	111	371	97	111	39	137
cars	74	260	129	271	144	305	336
glass	64	205	58	199	40	176	355
titanic	58	18	133	62	296	306	125

Table 6: IDS Lambda parameters obtained after tuning

	cars	**glass**
From	0.07 — 0.44 — 0.5 — 2.13 — 15	0.02 — 0.74 — 0.5 — 2.93 — 14
To	0.01 — 0.52 — 0.25 — 2.0— 8	0.0 — 0.97 — 0.33 — 3.0 — 6
	titanic	**cells**
From	0.02 — 0.01 — 1.0 — 2.15 — 13	0.15 — 0.24 — 1.0 — 2.89 — 18
To	0.0 — 0.18 — 1.0 — 2.5— 4	0.39 — 0.37 — 1.0 — 2.77 — 9

Table 7: IDS Metrics improvement after parameter tuning

Parameters	**cells**	**glass**	**titanic**	**cars**
ccp_alpha	0.0	0.0	0.0	0.0
max_depth	4	2	2	10
max_features	sqrt	sqrt	sqrt	sqrt
max_leaf_nodes	10	None	None	None
max_rmp_calls	3	6	3	3
min_impurity_decrease	0.0	0.0	0.0	0.0
min_samples_leaf	1	4	1	1
min_samples_split	2	2	2	2
rule_cost	Length	Gini	Gini	Gini
threshold	0.0001	0.0001	0.0001	0.0001
Best Score	0.95	0.64	0.84	0.79

Table 8: RuleOPT Best Parameters and Scores after tuning

5 Discussion

This study sought to investigate and compare various rule methods using mathematical optimization. The findings are the strengths of each tested algorithm; however, it is difficult to highlight which one outperforms the others, as each of them excels under specific conditions. RuleOPT and MDL work well with large data sets while maintaining good performance metrics. However, for some users, the Minimum Description Length (MDL) algorithm may yield a very small number of rules, which might be seen as a disadvantage for those seeking longer lists of rules. On the other hand, RuleOPT consistently produced a comparable number of rules, regardless of the size of the size of the dataset.. Moreover, each rule is assigned a weight, which indicates its importance, which I consider to be a really valuable feature. Furthermore, IDS provided very clear interpretability metrics and a highly understandable framework, even for individuals not well-versed in the Python programming language. Additionally, there were few, if any, requirements for preprocessing the datasets, which made using IDS a very efficient process. The author included numerous Jupyter notebooks in the repository, which were also helpful. It should be noted that IDS ran out of memory while attempting to process larger datasets, which led to the need to restart the environment. In contrast, RuleOPT and MDL did not encounter this issue. Going forward, CORELS performed adequately; however, it is somewhat outdated compared to other algorithms tested, given the advancements in currently developed algorithms. There is very little information available online, and there is absolutely no information about rule generation at all. Nevertheless, computational issues were a significant problem for IDS and MDL. Problems with package dependencies, environments, outdated libraries, or building wheels were pretty common and required considerable time to identify and resolve. The knowledge available online is limited to the dedicated paper and the GitHub repository, which some may consider a drawback as well. To sum up, I would definitely implement IDS, MDL, and RuleOPT in real-life cases that require the use of one of these methods. At the beginning of my study, I tested nine different algorithms, and only these, including CORELS, worked smoothly without any need to fix the source code.

6 Appendix

In Figure 3, the AUC-ROC curve for the *tictactoe* dataset demonstrates the model's performance in distinguishing between classes. The area under the curve (AUC) is 0.96, indicating great model performance, as it is close to 1.0. With great classification accuracy, the curve demonstrates a good capacity to distinguish between the real positive rate and the false positive rate.

Figure 4 shows the AUC-ROC curve for the *glass* dataset, displaying the model's performance across multiple classes. The AUC values for the different classes are 0.66, 0.61, 0.54, and 0.63, indicating moderate to poor performance in classifying these groups. The fact that the curves approach the diagonal line implies that the model may not be particularly good at appropriately classifying these classes.

The model's performance in a multi-class categorization situation is illustrated in Figure 5 by the AUC-ROC curve for the *cars* dataset. Excellent score in classifying between the various classes is indicated by the AUC values of 0.95, 0.93, 0.94, and 0.91 for each. The curves are significantly above the diagonal, reflecting the model's strong capability to separate true positives from false positives across the car classes.

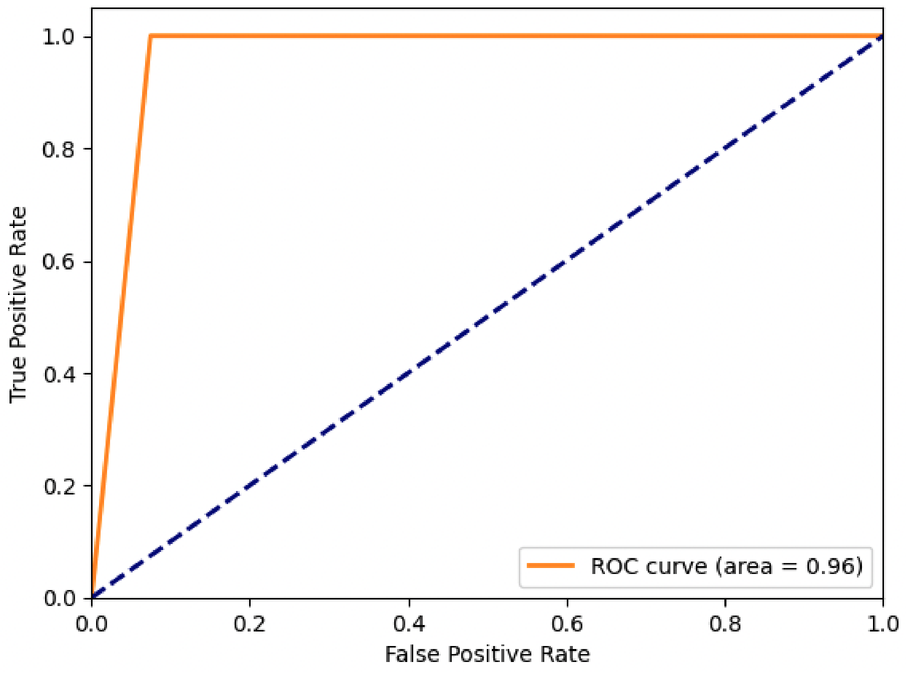

Figure 3: AUC-ROC Curve, tictactoe

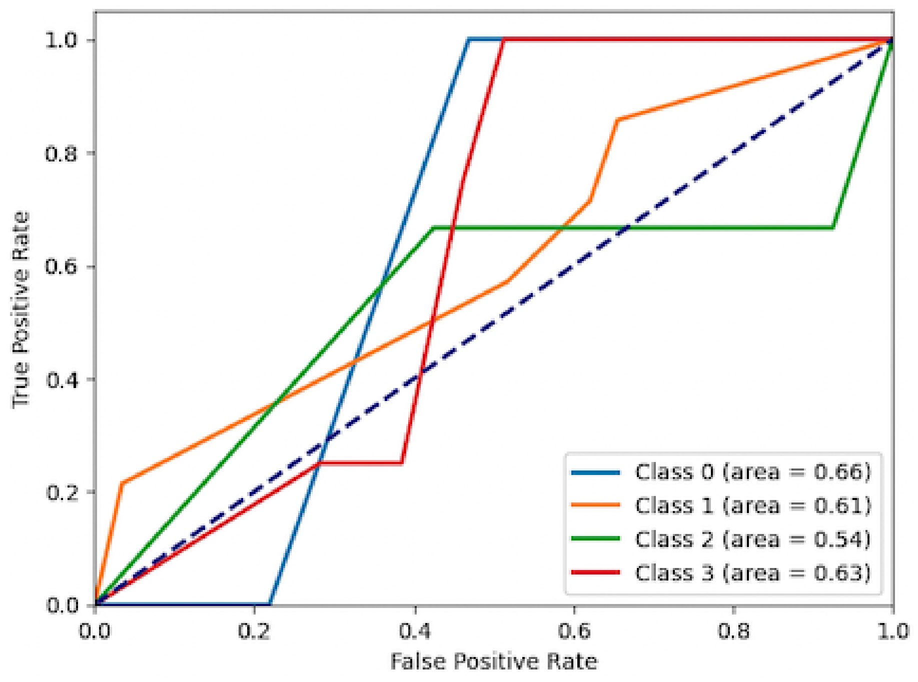

Figure 4: AUC-ROC Curve,glass

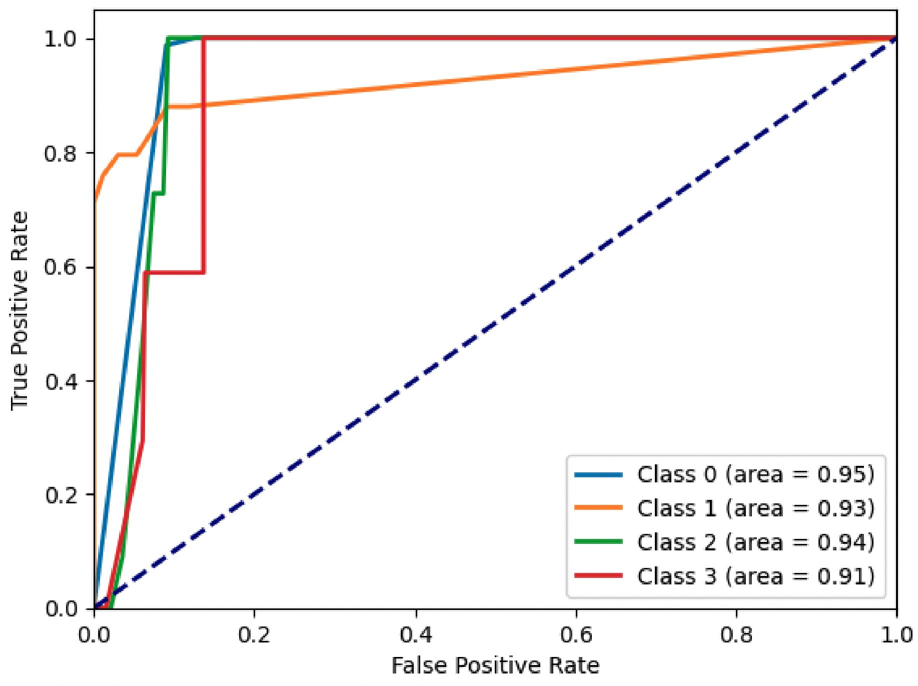

Figure 5: AUC-ROC Curve, cars

7 Repository

https://github.com/QBKFD/XAI_Thesis

8 References

Alikhademi, K., Richardson, B., Drobina, E., Gilbert, J. E. (2021). *Can Explainable AI explain unfairness? A framework for evaluating explainable AI.* arXiv preprint arXiv:2106.07483. Retrieved from https://arxiv.org/abs/2106.07483

Belcastro, L., Cantini, R., Marozzo, F. et al. (2022). *Programming big data analysis: principles and solutions. J Big Data*, 9, 4. https://doi.org/10.1186/s40537-021-00555-2

Bertsimas, D., Dunn, J. (2017). *Optimal classification trees. Machine Learning*, 106(7), 1039-1082.

Binns, R. (2019). *On the apparent conflict between individual and group fairness. Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency (FAT)**. University of Oxford, Department of Computer Science.

Cendrowska, J. (1987). *PRISM: An algorithm for inducing modular rules. International Journal of Man-Machine Studies*, 27(4), 349-370.

Chang, T. L., Xia, H., Mahajan, S., Mahajan, R., Maisog, J., Vattikuti, S., Chow, C. C., Chang, J. C. (2024). *Interpretable (not just posthoc-explainable) medical claims modeling for discharge placement to reduce preventable all-cause readmissions or death.*

Clark, P., Niblett, T. (1989). *The CN2 induction algorithm. Machine Learning*, 3(4), 261-283. https://doi.org/10.1007/BF00116835

Confalonieri, R., Weyde, T., Besold, T. R., Moscoso del Prado Martín, F. (2021). *Using ontologies to enhance human understandability of global post-hoc explanations of black-box models. Artificial Intelligence*, 296, 16. https://doi.org/10.1016/j.artint.2021.103471

Cordón, O. (2010). *A historical review of evolutionary learning methods for Mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems. International Journal of Approximate Reasoning.*

Crawford, S. L. (1989). *Extensions to the CART algorithm. Advanced Decision Systems.*

Dash, S., Günlük, O., Wei, D. (2018). *Boolean Decision Rules via Column Generation.* arXiv. https://arxiv.org/abs/1805.09901

Dai, W., Ji, W. (2014). *A MapReduce implementation of C4.5 decision tree algorithm. International Journal of Database Theory and Application*, 7(1), 49-60.

Farnam Street. (2024). *Complexity Bias: Why We Prefer Complicated to Simple.* Retrieved from https://fs.blog/tag/complexity-bias/

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., Kagal, L. (2019). *Explaining explanations: An overview of interpretability of machine learning. Proceedings of the 2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA), IEEE. Massachusetts Institute of Technology, Cambridge*, MA.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D. (2018). *A survey of methods for explaining black box models.* ACM Computing Surveys, 51(5), Article 93. https://doi.org/10.1145/3236009

Hugo M. Proença, Matthijs van Leeuwen (2019). *Interpretable multiclass classification by MDL-based rule lists.* arXiv. https://doi.org/10.48550/arXiv.1905.00328

Jijo, B. T., Abdulazeez, A. M. (2021). *Classification based on decision tree algorithm for machine learning.* Vol. 02, No. 01, pp. 20 – 28. ISSN: 2708-0757.

Lampathaki, F., Agostinho, C., Sesana, M., Glikman, Y. (2020). *Moving from 'black box' to 'glass box': Artificial Intelligence in Manufacturing*, with XMANAI.

Lewis, R. J. (2000). *An introduction to classification and regression tree (CART) analysis.* Presented at the 2000 Annual Meeting of the Society for Academic Emergency Medicine in San Francisco, California.

Li, H., Sheu, P. C. Y. (2022). *A scalable association rule learning and recommendation algorithm for large-scale microarray datasets.* J Big Data, 9, 35. https://doi.org/10.1186/s40537-022-00577-4

Linardatos, P., Papastefanopoulos, V., Kotsiantis, S. (2021). *Explainable AI: A Review of Machine Learning Interpretability Methods.* Entropy, 23(1), 18. https://doi.org/10.3390/e23010018

Mahbooba, B., Timilsina, M., Sahal, R., Serrano, M. (2021). *Explainable Artificial Intelligence (XAI) to enhance trust management in intrusion detection systems using decision tree model.*

Mahya, P., Fürnkranz, J. (2023). *An empirical comparison of interpretable models to post-hoc explanations.* AI, 4(2), 426-436. https://doi.org/10.3390/ai4020023

McDermid, J. A., Jia, Y., Porter, Z., Habli, I. (2021). *Artificial intelligence explainability: the technical and ethical dimensions.* Philosophical Transactions of the Royal Society A, 379, 20200363. https://doi.org/10.1098/rsta.2020.0363

Norman, D. A. (2010). *Living with Complexity.* MIT Press.

Optimization Wiki. (n.d.). *Column generation algorithms.* Cornell University Computational Optimization Open Textbook.

Padmanabhuni, S. (2022, March 25). *Model agnostic vs model specific explainability.*

Pandya, R., Pandya, J. (2015). *C5.0 algorithm to improved decision tree with feature selection and reduced error pruning. International Journal of Computer Applications*, 117(16).

Papenmeier, A., Englebienne, G., Seifert, C. (2019). *How model accuracy and explanation fidelity influence user trust in AI.* Proceedings of the IJCAI 2019 Workshop on Explainable Artificial Intelligence (xAI). GESIS - Leibniz Institute of the Social Sciences, Germany; University of Twente, Netherlands.

Rai, A. (2019). *Explainable AI: From black box to glass box.* Academy of Marketing Science.

Rudin, C. (2019). *Stop explaining black box machine learning models for high stakes decisions and use interpretable models instea*d. Nature Machine Intelligence, 1(5), 206-215.

Saleem, R., Yuan, B., Kurugollu, F., Anjum, A., Liu, L. (2022). *Explaining deep neural networks: A survey on the global interpretation methods.* Neurocomputing.

Salzberg, S. L. (1994). *C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc. Machine Learning*, 16, 235–240. https://doi.org/10.1007/BF00993309

Sharma, S., Agrawal, J., Sharma, S. (2013). Classification through machine learning technique: C4.5 algorithm based on various entropies. International Journal of Computer Applications, 82(16).

Shen, Y., Sun, Y., Li, X., Eberhard, A., Ernst, A. (2022). *Enhancing Column Generation by* *a Machine-Learning-Based Pricing Heuristic for Graph Coloring. Proceedings of the AAAI Conference on Artificial Intelligence*, 36(9), 9926-9934. https://doi.org/10.1609/aaai.v36i9.21230

Štrumbelj, E., Kononenko, I. (2010). *An efficient explanation of individual classifications using game theory.* Journal of Machine Learning Research, 11, 1-18.

Türkşen, I. B. (2010). *A review of developments in fuzzy system models: Fuzzy rule bases to fuzzy functions.* Scientia Iranica, Transactions D: Computer Science Engineering and Electrical Engineering.

Vilone, G., Longo, L. (2021). *Notions of explainability and evaluation approaches for explainable artificial intelligence.* Information Fusion. Retrieved from https://www.elsevier.com/locate/inffus

Wang, M., Lyu, L., Zhang, Z., Chen, H., Li, J., Wang, J. (2021). *Scalable Rule-Based Representation Learning for Interpretable Classification.* arXiv preprint arXiv:2109.15103.
https://arxiv.org/abs/2109.15103