SIMPLIFYING RANDOM FORESTS THROUGH POST-HOC RULE EXTRACTION

by

LAUREN BRAKKE

(Under the Direction of Frederick Maier)

ABSTRACT

Despite the high accuracy of black-box models, a significant challenge remains: their decision-making processes are often too complex for humans to easily understand. In response, there has been a renewed attention to explainable and interpretable artificial intelligence, a field dedicated to making the decision-making processes of models more understandable. Building upon prior work and using the Random Forest model as a basis, we create a rule extraction framework which seeks to produce a more understandable model that retains predictive performance. Through the use of post-hoc rule extraction methods, we extract rules from the original ensemble, reduce the size of the ruleset, and thus improve the overall explainability.

INDEX WORDS: Explainable Artificial Intelligence, Interpretable Artificial Intelligence

SIMPLIFYING RANDOM FORESTS THROUGH POST-HOC RULE EXTRACTION

by

LAUREN BRAKKE

B.A. Cognitive Science, University of Georgia, 2022

B.A. Philosophy, University of Georgia, 2022

A Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2024

SIMPLIFYING RANDOM FORESTS THROUGH POST-HOC RULE EXTRACTION

by

LAUREN BRAKKE

Major Professor:	Frederick Maier
Committee:	Kimberly Van Orman Khaled Rasheed

Electronic Version Approved:

Ron Walcott
Vice Provost for Graduate Education and Dean of the Graduate School
The University of Georgia
May 2024

ACKNOWLEDGEMENTS

I’d like to thank Dr. Maier and the members of my committee for their guidance. I also want to thank my sister, Hailey; my partner, Tanner; my mom; and my dad- all of whom have provided me with support and encouragement throughout my academic career.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

CHAPTER 1

BACKGROUND

1.1 Introduction

Concerns over the lack of transparency of 'black-box' models have led to an increased interest in the field of interpretable and explainable artificial intelligence (AI). Often considered a black-box due to its complex structure of numerous decision trees, the Random Forest model often proves difficult for humans to interpret. Recognizing the importance of simplifying these models for improved transparency and understanding, this thesis focuses on employing post-hoc methods to simplify the output of random forests while preserving their high predictive accuracy.

1.2 Background

The field of machine learning has been transformed by the emergence of systems capable of exceptional accuracy, commonly known as ‘black-box' models [1]. These models are considered black-boxes because their lack of transparency and complex internal processes render humans incapable of comprehending the reasoning behind specific results [1]. Commonly seen in the form of *neural networks* (NNs) or d*eep neural networks* (DNNs), these models display remarkable predictive capabilities. However, their lack of transparency, often referred to as ‘the black-box problem,’ means that humans are unable to understand their inner workings.

The black-box problem in machine learning can be addressed through the use of both interpretable AI and explainable AI [15]. The terms *interpretability and explainability* are often used interchangeably in the context of understanding machine learning models, however many authors argue the importance of distinguishing between them [7, 9, 13, 15]. Interpretable ML involves designing inherently clear models, whereas explainable ML aims to provide explanations for opaque, complex black-box models [15].

Interpretable models provide clear rationale for their outputs and can therefore be easily understood by humans [14]. Inherently interpretable, or "white-box," models include decision trees, rule-based models, and linear models [1]. Despite their transparency, these models often fall short in predictive accuracy compared to their black-box counterparts. Therefore, an alternative approach involves using post-hoc methods to explain the workings of black-box models without affecting their internal mechanisms [14]. Post-hoc methods prioritize model accuracy and separately generate explanations, offering insights into the decision-making processes of black-boxes without sacrificing predictive power [11].

In our work, we focus on the use of post-hoc methods to simplify the output of the random forest model in order to preserve its high accuracy, while also making its output more understandable. In addition, we compare our approach against several directly interpretable models in order to evaluate the trade-offs between accuracy and complexity inherent to each method.

Despite the growing amount of research dedicated to these machine learning methods, the field still lacks universally accepted measures for evaluating the quality of ML explanations [3, 8, 9]. Doshi-Velez and Kim [14] present a classification system for interpretability evaluation methods, identifying three key categories: application-grounded, human-grounded, and functionally-grounded. Application-grounded and human-grounded evaluation methods focus on conducting human-subject experiments to assess the quality of explanations; meanwhile, functionally-grounded evaluation methods leverage mathematical definitions of interpretability to assess the quality of a model [14].

In this thesis, we focus on functionally-grounded evaluation methods. In this evaluation method, a formal definition of interpretability serves as a proxy to evaluate the quality of an explanation, such as the depth of a decision tree [16]. However, establishing suitable measurement criteria and metrics is a challenging task, due to the inherent difficulties in quantifying interpretability and explainability, and continues to be a contentious issue.

1.3 Motivation

As machine learning models continue to replace human decision-making in traditional areas, the need to comprehend the reasoning behind these decisions becomes more pressing [7]. Despite black-box models demonstrating the ability to achieve high accuracy, the lack of transparency inherent in these systems can lead to serious consequences [1, 6]. For instance, one controversy caused by opacity in machine learning models involves an Amazon.com case in which a model inadvertently excluded minority neighborhoods from free same-day delivery even though neighboring areas qualified [1]. Machine learning models often make decisions based purely on patterns recognized in the training data, without comprehending the rationale behind these patterns [5]. Understanding the reasoning behind an algorithm’s decision is particularly critical in domains like healthcare and finance, where errors could result in severe repercussions, such as improper patient care or the denial of loans [2, 3, 4].

In response to the rise of ethical concerns and lack of user trust, there has been a renewed attention to explainable and interpretable artificial intelligence, a field dedicated to making the decision-making processes of models understandable [8]. These systems cultivate user trust and give humans the ability to comprehend the underlying mechanisms that contribute to a model's decisions. Below is a synthesized list from a few sources [5, 11, 12] which highlight the main motivators for explaining machine learning systems:

1. **Building Trust**: The development of interpretable AI is driven by the need to trust models, and ensure that their decisions are justifiable [5, 12]. Transparent models allow users to understand their internal processes and outcomes, which is essential given AI often produces biased results due to patterns in the training data. By providing clear justifications for their decisions, interpretable AI not only strengthens user trust but also informs thoughtful and responsible use of AI technology [5, 11].
2. **Enhancing Control**: Explainability is crucial for identifying and correcting system errors, particularly in contexts like debugging [11]. Deeper insights into a system's behavior uncover vulnerabilities and flaws, enabling better system control and error management.
3. **Facilitating Improvement**: Explainable AI enables ongoing model refinement. User comprehension behind how and why models arrive at specific outputs allows for informed improvements, resulting in more intelligent and effective AI systems [11].
4. **Promoting Ethical Decision-Making**: Black-box models can inadvertently perpetuate biases present in the training data, such as unintentional gender-biased word embeddings [5], leading to discriminatory algorithmic outputs. Decisions produced by machine learning

algorithms should align with ethical standards, and by understanding the rationale behind a model's output, interpretable AI can help us ensure that the model is unbiased [10].

The motivations for developing explainable and interpretable artificial intelligence, as summarized above, directly influence the direction of our work. By utilizing post-hoc methods, we aim to simplify the output of the random forest model, thus enhancing our understanding of the model's underlying decision-making processes. This approach not only preserves the accuracy of this model, but also provides outputs which are more understandable, addressing key concerns such as building user trust and promoting ethical decision-making. Therefore, our work facilitates better control over AI systems, allows for continuous improvements, and ensures that AI decisions are aligned with ethical standards.

1.4 Our Contribution

The random forest model, typically classified as a black-box due to its complex structure, can be simplified through the use of post-hoc methods. Unlike typical black-box models where the individual components– such as neurons in a neural network– have no meaning, random forests are composed of numerous white-box models, namely decision trees [17, 18]. While each tree is individually interpretable, their aggregation into a forest, often comprising thousands of deep trees, renders the overall model a black-box. Interpreting a random forest, therefore, involves effectively synthesizing the extensive outputs from a multitude of trees within the forest [17].

The focus of this thesis is to apply post-hoc strategies to the random forest model, generating global explanations that render the model's collective output more transparent while preserving its high predictive accuracy. Various post-hoc strategies can be utilized to interpret the output of a random forest, such as *rule extraction*, *size reduction,* and *local explanation.* In our work we focus on rule extraction methods, which involve identifying decision paths from the root to leaf nodes within the trees, and transforming them into a comprehensible set of rules or a dimensionally reduced set, thereby enhancing the model's interpretability. There are several relevant works which focus on utilizing rule extraction methods, such as *Node Harvest* [22] which leveraged shallower sections of trees to create rules, as well as *inTrees* [23] which extracts and prunes interpretable rulesets. In our work, we propose several modifications to the *ExtractingRuleRF* algorithm [26], a post-hoc method which produces global explanations for the random forest algorithm through a refined ruleset. In our view, this algorithm has several limitations, and by addressing them we can create a less complex model.

ExtractingRuleRF [26] adopts a greedy approach to obtain a reduced ruleset from a random forest. The algorithm operates in two phases: rule integration and rule extraction. The rule integration phase involves ranking rules derived from each tree based on several criteria in a sequential manner, and then integrating them by removing redundant conditions, covered rules, and duplicate rules to achieve a simplified ruleset. Subsequently, the rule extraction phase involves extracting rules either through a bottom-up or top-down scheme in order to obtain an optimal final ruleset. To further refine the ruleset while preserving predictive performance, we propose RandomForestRuleExtractor (RAFREX), which addresses what we feel are limitations to ExtractingRuleRF (ERRF). Our modifications include removing the rule ranking scheme to improve training time without significantly affecting model performance, and adding additional rule integration techniques in order to reduce redundancy and address conflicting rules. Our analysis reveals that the best-performing RAFREX variant achieves an average performance comparable to the RF model, without statistically significant differences, and with an average complexity reduction rate of 67.48%. Meanwhile, the average accuracy differences between ERRF and RF are statistically significant, and ERRF yields a lower average complexity reduction rate of 62.97%.

1.5 Outline of the Thesis

The structure of the remainder of this thesis is as follows: Chapter 2 provides a comprehensive review of prior works, discussing different types of interpretability, various approaches to interpretable models, and the metrics and techniques used for assessing the interpretability and explainability of a model. In Chapter 3, we provide a discussion of our methodology, detailing the modifications made to the ExtractingRuleRF algorithm, as well as an overview of the experimental setup. Chapter 4 presents our experimental analysis and their results. Chapter 5 concludes by encapsulating our findings and providing suggestions for future research in this domain.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter discusses research contributions in the field of interpretable and explainable AI, focusing on those which explain the random forest model through techniques such as size reduction, rule extraction, and local explanation. It also explores prior work in model agnostic approaches, which explain the behavior of any black-box model, in addition to directly interpretable models. Furthermore, the chapter examines various metrics and techniques for assessing model complexity, offering insights into their interpretability and functionality.

Reference	Global vs. Local	Direct vs. Post-Hoc	Model Specific vs. Model Agnostic	Year
Meinshausen [22]	G	PH	S	2010
Deng [23]	G	PH	S	2019
Wang et al. [24]	G	PH	S	2020
Sirikulviriya and Sinthupinyo [25]	G	PH	S	2011
Thi et al. [26]	G	PH	S	2015
Obregon et al. [27]	G	PH	S	2019
Obregon and Jung [28]	G	PH	S	2023
Zhang and Wang [29]	G	PH	S	2009
Bernard et al. [30]	G	PH	S	2009
Latinne et al. [31]	G	PH	S	2001
Van et al. [32]	G	PH	S	2007
Gibbons et al. [33]	G	PH	S	2013
Welling et al. [34]	L	PH	S	2016
Moore et al. [35]	L	PH	S	2018

Mollas et al. [36]	L	PH	S	2019
Ribeiro et al. [37]	L	PH	A	2016
Ribeiro et al. [38]	L	PH	A	2018
Lundberg and Lee [39]	L	PH	A	2017
Zhou and Hooker [40]	G	PH	A	2016
Dhurandhar et al. [41]	L	PH	A	2018
Luss et al. [42]	L	PH	A	2019
Breiman [43]	G	D	S	1984
Cohen [44]	G	D	S	1995
Frank and Witten [45]	G	D	S	1998
Gaines and Compton [46]	G	D	S	1995
Dash et al. [47]	G	D	S	2018
Lakkaraju et al. [48]	G	D	S	2016

Table 2.1: Related works

2.2 Random Forest Explainability Approaches

Work on what we now know as the random forest (RF) model began in the 1990s, with Brieman’s 2001 work being a significant example [19]. The random forest model is an ensemble learner composed of numerous decision trees known for its high predictive accuracy [17, 18]. While the construction process of individual trees within the forest is intuitive and straightforward, their standalone accuracy is not as competitive as other classification approaches [18]. However, the random forest model overcomes this limitation by aggregating the outputs of many decision trees, often enhancing predictive performance [17].

Although each tree within the forest is often considered to be a white-box model, the large number and depth of the decision trees it comprises leads to it often being classified as a black-box model [17, 18]. Extracting and interpreting the collective knowledge from these trees presents a significant challenge, making the overall model less transparent, despite the interpretability of its individual components. Consequently, while each tree in a random forest is interpretable, the aggregated knowledge and decision-making process of the entire forest become opaque, solidifying its status as a black box model [17]. However, unlike neural networks, they do have the benefit that decisions at any given node do not directly involve the values of input attributes.

To address the opaque nature of the random forest model, various post-hoc strategies can be utilized to simplify its output. These strategies include *rule extraction*, *size reduction,* and *local explanations*. Rule extraction methods involve identifying decision paths from the root to leaf nodes within the trees, and transforming them into a comprehensible set of rules or a dimensionally reduced set, thereby enhancing the model's interpretability [17, 18]. Size reduction techniques focus on condensing the forest into a more manageable size without losing predictive accuracy [18]. Additionally, local decomposition methods provide detailed analysis of individual predictions, offering insights into the model's specific decision-making processes [17].

2.2.1 Rule extraction

Our work focuses on the utilization of rule extraction strategies to extract and subsequently simplify the rules produced by a random forest to produce a more understandable model. This involves translating each decision tree in the forest into an equivalent set of rules, representing the same knowledge in a different form [20]. For every leaf node, a specific rule is derived, corresponding to the path from the root of the decision tree to the respective leaf node [20].

Meinshausen [22] created Node Harvest, a method designed to improve both the accuracy and interpretability of predictions by leveraging the shallower sections of tree ensembles. This approach involves two key phases; the initial phase focuses on maintaining a clear and simple model structure by extracting the simpler, shallow parts of the trees while discarding their deeper, more complex counterparts. Subsequently, the extracted shallow trees are combined to form an ensemble that performs effectively on the training dataset. In the final stage, selected rules are weighed to ensure a balanced and interpretable rule ensemble.

The inTrees algorithm, developed by Deng [23], is designed to extract and refine interpretable information from random forests. It begins by extracting and categorizing rules from the forest, and then prunes them to remove irrelevant or noise-generating rules. Following this, a concise set of significant, non-redundant rules is selected and key interactions between them are identified. The final stage involves the creation of a learner, which leverages the refined information from the chosen rules to make predictions on new data, ensuring the model's predictive efficacy and interpretability.

Wang et al. [24] developed the Improved Random Forest-based Rule Extraction (IRFRE) method, which extracts rules from a RF for breast cancer diagnosis. Their approach identifies the optimal combination of rules through a multi-objective evolutionary algorithm (MOEA) that considers both accuracy and interpretability. This involves techniques such as non-dominated sorting, uniform crossover, and flip bit mutation to find the Pareto optimal front, a series of accuracy and interpretability trade-offs which correspond to the evolved, optimal rulesets.

Sirikulviriya and Sinthupinyo [25] proposed a method that involves extracting rules from each tree in the forest and integrating them using several techniques. First, redundant conditions, or more general conditions which appear in the same rule with more specific conditions, are removed from each rule. Then, for every pair of decision trees, redundant, conflicting, and overly specific rules are removed; additionally, rules that overlap are combined to reduce redundancy. This integration process continues if the accuracy of the new rules on the validation set is still improved.

Thi et al. [26] adopted a greedy approach to obtain a reduced ruleset from a RF that balances high accuracy and acceptable coverage, called ExtractingRuleRF. The algorithm operates in two phases: rule integration and rule extraction. The rule integration phase involves ranking rules derived from each tree based on several metrics, and then integrating them by removing redundant conditions, covered rules, and duplicate rules to achieve a simplified ruleset. Subsequently, the rule extraction phase involves extracting rules either through a bottom-up or top-down scheme in order to obtain an optimal final ruleset.

Moreover, Obregon et al. [27] developed RuleCOSI, which extracts decision rulesets from a boosting ensemble of binary decision trees. Their method employs a combination matrix to effectively merge the predictor spaces of each decision tree, considering their respective weights. Additionally, a pessimistic error approach is used to prune unnecessary conditions from the rules. Obregon and Jung [28] later modified their approach to create RuleCOSI+, which is capable of running much faster for ensembles with hundreds of trees. Additionally, it can be applied to both bagging and boosting ensembles.

The overall goal and techniques employed in these methodologies lay the groundwork for the direction of our work later in this thesis. By using similar strategies to those in the literature, such as weighting rules [22], removing irrelevant rules [23], addressing rule conflicts [25], and extracting the top-performing rules to form the final ruleset [26], we aim to achieve a reduced ruleset from the RF, but with similar predictive power.

2.2.2 Size Reduction and Local Explanation

The other two post-hoc explanation techniques include size reduction and local explanation. Size techniques reduce the complexity of a random forest through removing and simplifying the trees within the forest. Meanwhile, local explanation techniques explain the relationship between specific input-output pairs, rather than globally explaining the whole model [12]. Although we do not utilize these techniques in our work, they represent alternative approaches that can be used to create a more understandable model and provide insights behind individual predictions.

Zhang and Wang [29] developed a methodology to reduce the number of trees in the RF while preserving, or possibly improving, the prediction accuracy. They use three methods for tree removal: evaluating each tree's contribution to the overall accuracy to identify expendable ones, removing trees that are structurally similar, and removing trees with similar predictions. They evaluate various sub-forests to find the most efficient subset, choosing smallest sub-forest that maintains the highest performance level, effectively identifying an optimal size of the reduced forest.

Bernard et al. [30] explored tree selection methods to obtain a subset of trees that outperform the original forest. Rather than identifying the optimal subset of individual trees, they instead focus on how reducing the forest to a particular subset of trees can improve its accuracy. They utilized two sub-optimal selection methods: Sequential Forward Selection (SFS), which gradually adds trees that boost performance, and Sequential Backward Selection (SBS), which removes the less accurate trees.

Latinne et al. [31] used the McNemar test of significance on the tree predictions to reduce the number of RF trees. This method involves comparing the predictions from a larger RF and a RF reduced in size. If the test finds no significant difference in predictions, it suggests that the reduced forest is sufficient for achieving the same accuracy.

Van et al. [32] presented a method to learn a single decision tree that approximates the decisions made by an entire tree ensemble. The construction of the tree involves selecting the most informative tests from the ensemble, based on their ability to accurately predict class distributions, and using them to build the new tree. This process starts at the root and iteratively adds branches based on these selected tests and uses pre-pruning to prevent the tree from becoming too large or complex.

Similarly, Gibbons et al. [33] obtained an interpretable individual decision tree that maintained the high predictive accuracy of a tree ensemble. They first generate a large, artificial dataset that imitates the original data's distribution. They then grow a single tree based on this artificial dataset in order to closely reproduce the output of the random forest, with the dataset's size helping reduce the sensitivity of the tree to minor changes. Furthermore, to make the tree more interpretable, they prune the tree to a more understandable depth.

In terms of local explanation techniques, the Forest Floor algorithm, introduced by Welling et al. [34], is one which determines the impact of each variable on the model's individual predictions. This algorithm depicts prediction breakdowns within 2D or 3D feature spaces; however, it may have limitations in cases of high-dimensional feature spaces, in which the inherent complexity could render such visual interpretations less effective [2].

Moore et al. [35] produce local explanations for the RF model based on a list of features and their ranges, ranked based on their contribution. They track changes in tree nodes' outputs before and after an instance passes through, and calculate each feature's impact based on these changes for each prediction. However, [36] notes that when the list of features is long with very narrow ranges, the interpretation becomes less reliable, as minor variations in the features could render the interpretation useless.

In response, Mollas et al. [36] introduced LionForests, a method which provides natural language explanations for individual RF predictions. Utilizing unsupervised learning methods such as association rules and k-medoids clustering, their approach simplifies paths and features within RF models, effectively reducing the feature count and broadening the feature ranges to yield more robust interpretations.

2.3 Model Agnostic Approaches

In addition to model specific approaches, model agnostic methods can also be used to provide retrospective insights into otherwise opaque models [21]. These methods are typically post-hoc, meaning they apply techniques to generate explanations for uninterpretable, black-box models without altering or fully comprehending the original model's inner workings. This can involve a variety of techniques such as natural language explanations, visualizations of learned models, and example-based explanations [20]. Furthermore, they are capable of explaining a range of black-box models, such as neural networks, random forests, and support vector machines. Although our work primarily focuses on the use of post-hoc strategies to simplify the random forest model, the overall objective is the same as that of other model-agnostic techniques: to enhance the understandability of complex models.

One of the most commonly used methods for locally approximating black-box models is LIME (Local Interpretable Model-Agnostic Explanations), proposed by Ribeiro et al. [37]. LIME creates simulated data points around a specific input instance to see how changes affect predictions. A simpler, more transparent model is then trained on these new instances, which helps interpret the original model's decisions for that specific input.

Ribeiro et al. [38] advanced this concept with the development of high-precision rules, called anchors. When applied, anchors ensure the prediction remains unchanged despite any alterations to the other feature values of that instance. The formation of each anchor starts with an empty rule applicable to all instances, which is incrementally refined to increase precision and coverage.

Lundberg and Lee [39] introduced SHAP (SHapley Additive exPlanations), leveraging Shapley values to determine how each feature influences model predictions at a local level. SHAP evaluates all possible combinations of features to determine their individual impact. Each SHAP value is calculated by averaging the changes in prediction when a feature is added, weighted by the number of ways a feature can be added.

Zhou and Hooker [40] developed a method to simplify complex models into single decision trees. They introduced a splitting method that focuses on the asymptotic differences of Gini indices in order to stabilize tree structures. Zhou and Hooker specifically focus on simplifying a RF into a single DT in their study.

Furthermore, Dhurandhar et al. [41] introduced the Contrastive Explanations Method (CEM) for identifying necessary and absent features for a prediction. Given an input and its associated prediction, the method identifies not only the features that need to be minimally and sufficiently present to produce a specific prediction, but also those that must be minimally and necessarily absent. Luss et al. [42] adapted CEM for image-based applications by incorporating monotonic functions.

2.4 Directly Interpretable Models

In this thesis, we intend to evaluate our proposed model against directly interpretable models, known for their balance of accuracy and simplicity, to assess how effectively our model provides explanations without sacrificing accuracy. This comparison seeks to gauge where our model stands relative to the interpretability these models provide. Specifically, we compare our model against CART [43], RIPPER [44], PART [45], RIDOR [46], and BRCG [47]. Direct explanation models are inherently understandable and provide the most straightforward approach to achieving interpretable artificial intelligence [18]. As noted in [1, 9, 10, 11, 12], there's a limited selection of models within current research recognized for their inherent interpretability:

1. Decision Trees: Represented as a hierarchical graph structure, internal nodes function as tests on specific attributes or features and leaf nodes serve as class labels. Each branch from root

to leaf represent decision rules, allowing if-then logic to assign class labels based on specific criteria.

2. Rule-based Models: These models create decision rules mapping observations to actions, usually in an “if-then” format. involving conditions and outcomes. [1]. The "if" part constitutes a *clause*, while the “then” part forms the rule’s outcome. Rules can be structured in *Disjunctive Normal Form* (DNF), a disjunction of conjunctions, or *Conjunctive Normal Form* (CNF), a conjunction of disjunctions. Additionally, decision sets are comprised of independent classification rules, each functioning independently, without reliance on others in the set [1].
3. Linear Models: These models evaluate and visually display the importance of features through *feature importances.* These provide insights into the model’s decision-making process by illustrating the impact of feature values on the output.

Several proposals are based in these directly explainable methods to create direct explanation models. Classification and Regression Trees (CART), developed by Breiman et al. [43], is a decision tree algorithm designed for both classification and regression tasks. CART systematically partitions data to enhance homogeneity in each subset by evaluating attributes and their potential split points for optimal division. Impurity in these divisions is measured using the Gini index, which assesses the likelihood of misclassification based on the class distribution within a subset. CART's partitioning process continues until subsets are perfectly homogeneous or a specific stopping criterion, like maximum tree depth, is reached.

The Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm, is a rule-based classification algorithm developed by Cohen [44] in 1995. RIPPER employs the Minimum Description Length (MDL) principle to create and refine rulesets, aiming for models (or in this case, rules) that concisely describe data without significantly compromising accuracy, through an assessment of each rule's length and associated errors. Starting with a ruleset covering all training data, RIPPER iteratively prunes and tests rules using a statistical significance test to improve predictive accuracy. This process, evaluated against a validation dataset, continues until a stopping condition is met or no further accuracy improvement is observed.

Introduced by Frank and Witten [45], Partial Decision Trees (PART) merge the decision tree logic of C4.5 with RIPPER's rule induction approach. PART creates rulesets by sequentially segmenting data, starting with a single rule that best classifies a data subset. After applying each rule, the covered instances are removed, and PART identifies the next rule for the remaining data. Through this iterative process, PART constructs and prunes a partial decision tree to extract the most generalized rule, represented by the leaf node with the most coverage, and then discards the remainder of the tree. This approach, similar to C4.5’s attribute-based division and entropy refinement, also strategically prunes branches to enhance rule accuracy and generalization.

Gaines and Compton [46] presented the RIDOR (RIpple-DOwn Rule) algorithm, which offers an incremental technique for rule induction in classification tasks. The methodology for handling rule exceptions involves identifying the most prominent exception to a pre-existing default rule, and then formulating this exception as its own rule. RIDOR then finds exceptions to this new rule, thereby generating a chain of rules in a "ripple-down" fashion. Rules are generated for each subsequent exception until no more exceptions can be identified. This ripple-down approach is highly efficient, as rules are formulated to handle specific exceptions rather than attempting to cover the entire instance space.

Furthermore, Dash et al. [47] introduced Boolean Rule Column Generation (BRCG), a technique for creating globally interpretable models using Boolean rules in DNF or CNF. BRCG combines column generation, an optimization technique in linear programming, and Boolean rule learning to iteratively improve the model's objective function, typically focused on minimizing classification error. To address the computational challenges of a large rule-space, the method employs an approximate algorithm with randomization for efficient search.

2.5 Evaluation Measures and Metrics

Despite the growing amount of research dedicated to creating interpretable machine learning methods, the field still lacks universally accepted measures for evaluating the quality of these explanation methods [11, 16]. This issue is often attributed to the innately subjective nature of explainability, which is contingent on numerous factors including individual users, the explanation itself, and the specific data the user is seeking [11, 16]. Furthermore, explanations are typically specific to a domain; there is no 'one-size-fits-all' explanation that suits every situation [16]. Doshi-Velez and Kim [14] address this by presenting a classification system for interpretability evaluation methods, identifying three key categories: application-grounded, human-grounded, and functionally-grounded.

2.5.1 Application-grounded Evaluation

Application-oriented evaluation involves conducting end-user experiments within real-world applications to determine how well the explanations assist domain experts in specific tasks [14]. This evaluation method measures the system's effectiveness in achieving its intended objectives, which in turn provides evidence of the success of the explanations. For example, Williams et al. [41] evaluated their homework-hint system based on improved student post-test performance, a method that, while lacking a standard metric, directly assesses the system's designed objectives.

2.5.2 Human-grounded Evaluation

The human-grounded evaluation approach is similar to the application-grounded evaluation, but focuses on simpler human-subject experiments without requiring domain experts [14]. The primary goal is to evaluate the quality of the explanation itself, rather than its suitability for specific applications [14]. Experimental designs like binary forced choice, forward prediction, and counterfactual simulation are used to gauge explanation quality [14]. These experiments test participants' understanding of the explanation, irrespective of the prediction's accuracy or the model type, offering insights into the effectiveness of different explanatory approaches [14].

2.5.3 Functionally-grounded Evaluation

Functionally-grounded evaluation eliminates the need for human involvement by leveraging mathematical definitions of interpretability to assess the quality of an interpretability method [14]. In this evaluation method, a formal definition of interpretability serves as a proxy to evaluate the quality of an explanation, such as the depth of a decision tree [16]. However, establishing suitable measurement criteria and metrics is a challenging task, due to the inherent difficulties in quantifying interpretability and explainability, and continues to be a contentious issue. Given that model size can reflect the complexity of models, metrics such as number of rules, length of rules, and depth of trees are used to measure the quality of model interpretability [16]. A basic yet widely used metric for complexity in the literature [24, 47, 49, 50, 51, 52] is defined as the sum of the rules in the ruleset + the total number of conditions in those rules. Additionally, to assess the complexity and interpretability of post-hoc explanation models, *rule reduction rate* can be used for quantifying the percentage of the reduction in size between the post-hoc model and original ensemble [28].

These metrics, however, do not take into account redundancy within the ruleset. It is possible for models to have rulesets with redundant rules, in that they share identical content or functionality. In such cases, the actual complexity of the model, when compressed, may be considerably lower than initially perceived. Therefore, to address possible redundancy in the ruleset, the Minimum Message Length (MML) principle can be utilized to compress both the data and the model.

Minimum Message Length, introduced by C.S. Wallace and D.M. Boulton [35] in 1968, is a foundational concept in statistical inference and machine learning. The "message" in MML consists of a model statement and a concise encoding of data based on that model. By minimizing the overall length of the compressed message, this metric can provide a more accurate estimation of the model’s true complexity.

CHAPTER THREE

METHODOLOGY

3.1 Overview

A significant challenge encountered with current white-box models is their often inferior classification performance compared to black-box models. This fact leads us towards innovating a solution that leverages the high accuracy of black-box models as a foundation for developing a more effective and understandable rule-based classification model. As such, there is an opportunity to apply rule extraction techniques to the random forest model, in order to produce a refined ruleset from the original model. We propose a series of enhancements to the ExtractingRuleRF algorithm, designed to optimize the algorithm's efficacy in producing a more interpretable ruleset without sacrificing accuracy. In this chapter, we discuss the ExtractingRuleRF algorithm, its limitations, and our proposed solutions to said limitations. The goal of these modifications is to further refine the algorithm, enabling it to produce a more concise ruleset derived from the random forest algorithm, while maintaining the integrity of its classification accuracy. This involves reducing the computational complexity of the algorithm, expanding the scope of rules that can be integrated, and adding additional techniques to integrate rules. Furthermore, we discuss the experimental setup, including the algorithms, datasets, and metrics used in the experiments .

3.2 ExtractingRuleRF Overview

The ExtractingRuleRF algorithm [26] refines rules produced by the RF model, aiming for a more comprehensible set of classification rules. Rules are extracted from each tree within the RF to produce the initial set of rules. Once the rules are extracted from the original RF, the algorithm follows a greedy approach with two phases: rule refinement and rule extraction. The refinement phase involves ranking, refining, and weighting rules, while still preserving the equivalent classification power [26]. Subsequently, the extraction phase considers the rank of each rule to identify a reduced final ruleset. This ruleset is used for prediction, in which the weight of the rule serves as its voting strength, similar to the voting process in RFs.

3.2.1 Rule Refinement

The rule refinement stage begins by setting the weight of every rule to one and consists of two stages: rule ranking and rule integration. The first stage involves assigning a rank to each rule based on several metrics. Instead of ranking the rules derived from entire forest collectively, Thi et al. [26] opt to rank the rules derived from each individual tree. This choice is made due to the subsequent integration phase. During the integration phase, rules with the same rank in different trees are considered for integration, with the aim being to maintain predictive performance in the integrated rule. Redundant conditions, covered rules, and duplicate rules are removed in order to reduce redundancy while preserving the overall predictive power [26]. As rules are integrated, their weights are updated. This weight signifies the number of corresponding paths across all trees from which the rule is derived [26].

3.2.1.1 Rule Ranking

During the rule ranking process, rules derived from each tree are ranked sequentially based on their priority. The original ExtractingRuleRF algorithm determines a rule's priority according to the following criteria:

```
Input: a raw ruleset derived from an original random forest
Output: a resulting interpretable refined ruleset
Phase 1: Rule Refinement
1.1. Initializing a weight of each rule to 1
1.2. Rule ranking
   - Rank based on criteria in sequence:
       i. Accuracy
       ii. Coverage
       iii. Number of important attributes in a tree
       iv. Number of important attributes in a rule
   - Prioritize rule with higher metric if two rules
     have equal previous metric (e.g., higher
     coverage if accuracy is equal)
1.3. Rule integration
   - Combine rules with the same rank
   - Refine ruleset by:
       i. Removing redundant conditions
       ii. Removing covered rules
       iii. Removing duplicate rules
   - Update weights post-merger
Phase 2: Rule Extraction
2.1. Rule extraction in bottom-up or top-down scheme
   - Bottom-up:
       i. Retains top-ranked rules
       ii. Stops when a rule falls below the accuracy
           threshold
   - Top-down:
       i. Iteratively discard lowest accuracy rules
       ii. Continues until accuracy falls below that of
           the raw ruleset and uses ruleset from
           previous iteration
2.2. Return a resulting interpretable refined ruleset
```

Figure 3.1: ExtractingRuleRF (ERRF) algorithm

1. Rule Accuracy: This is defined as to the percentage of new instances correctly classified by the rule [26], and is calculated for each rule using the Out-Of-Bag (OOB) data. This criterion has the highest priority to rank a rule. Rules with higher accuracy receive a higher rank.

2. Rule Coverage: This is defined as the percentage of new instances that satisfy all conditions of the rule [26], and is calculated on the OOB data. If multiple rules have the same accuracy, this criterion is used to rank the rules. Rules with higher coverage receive a higher rank.

3. Number of important attributes in a tree ($VI^t(X_i)$ where X_i is an attribute in a tree t of the forest): It is defined in [26] as the permutation importance by comparing the prediction accuracy of a tree before and after random permutation of X_i on the out-of-bag (OOB) data

[26]. The higher $VI^t(X_i)$ value, the more important X_i is in t. A higher value indicates greater importance, and if rules have equal coverage, this criterion is used to rank the rules. The higher number of the important attributes, the higher rank a rule has.

4. Number of the important attributes in a rule ($VIRT^t(X_i, R)$ where X_i is an attribute in a rule R extracted from a tree t of the forest): It is defined below using the frequency of X_i ($Freq(X_i)$) in a rule R that contains m attributes and the inverse frequency of X_i ($IFreq(X_i)$) in the entire ruleset RuleTree of the tree t [26]. In $Freq(X_i)$, count(X_j, R) is the number of occurrences of attribute X_j in R, also equal to the number of conditions that contain attribute X_j for j = 1..m. The higher $VIRT^t(X_i, R)$ value, the more important X_i is in R. If rules have the same number of significant attributes in a tree, this criterion is used to rank the rules. The higher number of the important attributes, the higher rank a rule has.

The result of this process is a ruleset derived from the random forest, composed of ranked subsets, corresponding to the rules derived from each tree in the forest [26]. The ranked subsets identify the most influential rules for accurately classifying test instances, and set the groundwork for the subsequent stages of the algorithm [26].

However, a drawback of this approach is that it takes up a significant amount of the time complexity, specifically for one of the lower ranked metrics. ‘Variable importance in a tree’ is calculated based on the change in predictive accuracy on the out-of-bag (OOB) data before and after randomly assigning observations to a variable’s child nodes in the tree. A greater change signifies higher importance of the variable. The need to calculate this metric for each variable for each tree in the forest significantly increases the computational workload of the algorithm, especially when dealing with larger datasets. Furthermore, this metric is positioned lower in the ranking scheme, indicating less importance in determining the rank of a rule, and yet it takes a significant amount of time to compute. The lower placement of this metrics in the hierarchy, despite the computational demands, suggests its limited impact compared to accuracy and coverage metrics, making a case for potential exclusion.

3.2.1.2 Rule Integration

During rule integration, rules of similar significance are combined to reduce redundancy while preserving the overall predictive power [26]. This involves removing redundant conditions, covered rules, and duplicate rules. When rules are integrated, their weights are updated- if two rules are merged, the new rule's weight is an aggregate of the weights of the individual rules. In order for two rules to be considered for integration they need two be derived from two different trees in the forest and have the same rank in their respective trees. The aim of this approach is to maintain predictive accuracy in the integrated rule.

However, this approach fails to consider the difference in predictive accuracy across different trees in the forest. As discussed in [53], every decision tree in the forest is constructed using a randomly selected subset of both the training data and features. This reduces the correlation between trees [53], thus making the forest diverse. Therefore, while the rule ranking scheme may successfully rank rules within a tree, this ranking is inadequate when it comes to comparing rules derived from different trees. A top-ranked rule in one tree may not have the same predictive accuracy as a top-ranked rule in another tree. Additionally, two rules with similar predictive power may not be integrated because they differ in rank. Adopting a new criteria to determine if two rules should be integrated, opposed to the rank metric utilized in [26], can result in more rules being integrated and help to better maintain predictive power in the resulting rule. There is also the possibility to integrate all rules, without the need to meet a specific criteria, which is demonstrated in multiple other approaches [54, 55, 56, 57, 58].

Moreover, Thi et al. [26] state that their goal is to reduce the number of rules in the final set, and yet they use very few integration techniques. Their integration scheme involves removing redundant conditions, covered rules, and duplicate rules:

1. **Removing redundant conditions in each rule**: Removing redundant conditions involves removing a condition if every data instance that satisfies that condition inevitably satisfies another condition in the rule [26]. After removing redundant conditions, the weight of the rule remains the same but the rule is no longer unnecessarily complex. Given R_1 with a redundant condition: "`b >= 4.05`":

   R_1 before refinement with a weight w: `IF b >= 4.05 AND a >= 0.9 AND a < 4.2 AND b >= 6.15 THEN 1`

   R_1 after refinement with an unchanged weight w: `IF a >= 0.9 AND a < 4.2 AND b >= 6.15 THEN 1`

2. **Removing covered rules**: Given two rules that lead to the same prediction and share all but one of their conditions, if the unique condition of the second rule encompasses that of the first, then the first rule is considered redundant and is removed from the set. In this scenario, the weight or importance of the second rule is increased by one. This adjustment is made because the second rule provides a prediction for not only itself but also for those previously

covered by the removed rule. Given R_1 and R_2 with a covered condition in R_1: “`a >= 2.3`”:

R_1 before removal: `IF a >= 2.3 AND a < 4.2 AND b >= 6.15 THEN 1`

R_2 with weight $w + 1$: `IF a >= 0.9 AND a < 4.2 AND b >= 6.15 THEN 1`

3. **Removing duplicate rules**: If two or more rules are identical in their conditions and outcomes, they are considered duplicates. Such duplicates can be removed, as they do not add any value in terms of interpretability or predictive accuracy. As with covered rules, when duplicates are encountered, one rule is eliminated from the set, and the remaining rule's weight is incremented by one to account for the vote of its duplicate. Given R_1 and R_2 which are have the same antecedent and consequent:

   R_1 before removal: `IF a >= 0.9 AND a < 4.2 AND b >= 6.15 THEN 1`

   R_2 with weight $w + 1$: `IF a >= 0.9 AND a < 4.2 AND b >= 6.15 THEN 1`

The ruleset can be further refined through modifying their current methods and applying new integration techniques. A ruleset is redundant if it can be reduced in size by removing at least one rule to produce a new ruleset that is still equivalent to the initial one [59]. Therefore, rules that have the same outcome can be merged into a single, equivalent rule, thus reducing the number of rules without compromising predictive performance [59, 60]. It is “obvious” that redundant rules should be removed, according to Ligeza, as they are a potential source of inconsistency and occupy more space, thus making the ruleset less transparent.

The differences in the rules establishes the degree to which they can be combined [60]. For example, if two rules predict the same outcome but differ by only one condition, the rule with the broader condition can replace the more specific one [26]. However, this approach overlooks potential cases in which multiple conditions are covered. In studies [55, 56, 58], "coverage" is defined more broadly, allowing for the integration of rules that predict the same outcome, share an equal number of initial conditions, and include identical attributes, but specify different values for one *or more* of these attributes. These rules can then be merged into a single, new rule.

Another case of rule redundancy is rule subsumption, where one rule is fully contained within another [60] Subsumed rules can be removed as they produce weaker, or fewer, results and require stronger conditions to be satisfied [59]. The subsuming, more general rule, offers the ability to classify more instances with fewer premises. Therefore, any results produced by the subsumed rule can also be produced by the subsuming rules [59].

There can also be cases in which two or more rules apply to the same instances, but reach different predictions [60]. In cases were two rules directly conflict, meaning the antecedent of the rules is exactly the same, the simplest approach to address them, as discussed in [54], is to ignore the conflict, as they do not contribute to the predictive accuracy and can therefore be removed.

Furthermore, another form of conflict involves cases where the ranges of the rules overlap, leading to two different predictions [61]. In this type of conflict, two rules predict different outcomes and there exists a non-empty set of training data that matches both rules [57]. Several strategies can be employed to address overlap between classes: Ligęza [59] uses a conflict resolution strategy to execute one rule; Nalepa [62] applies the concept of rule priority to each set of conflicting rules; Hall et al. [55, 56] create new sub-rules based on the conflicting rules; and Andrzejak et al. [63] resolve conflicts by assigning the class with the greatest probability. Despite the numerous strategies discussed in the literature that aim to reduce redundancy address conflict, Thi et al. only remove redundant conditions, rules where one condition is covered, and duplicate rules.

Moreover, Thi et al. opt to perform one phase of rule integration, meanwhile many similar schemes [54, 55, 56, 57, 58] perform more than one iteration of rule integration. In [58], rules from each pair of decision trees are combined as long as the accuracy on the validation set is improved. Similarly, [55, 56] adopt a comparable methodology by first removing duplicate rules and resolving conflicts; they then repeat their first step and eliminate any redundant rules that may have been created in the process of removing conflicts.

Therefore, although several strategies exist for removing redundancy and resolving conflicts, Thi et al.'s method primarily focuses on a single phase of rule integration with few integration techniques, and no use of conflict resolution strategies, unlike other approaches in the literature [54, 55, 56, 57, 58, 62, 63].

3.2.2 Rule Extraction

The second phase of the algorithm, rule extraction, extracts the best rules from the ruleset produced by the previous phase. Rule extraction can be approached in two ways: bottom-up and top-down.

1. **Bottom-up rule extraction**: In the bottom-up scheme, the algorithm retains the most accurate rules, starting from the highest ranked rules and gradually descending to the lower ranked ones. The process is driven by a user-specified accuracy threshold, which can often yield a smaller, highly accurate ruleset that applies to a specific subset of the data due to an acceptable compromise on the overall coverage of the rules in the final set [26].

2. **Top-down rule extraction**: Conversely, the top-down scheme operates by discarding rules with the lowest accuracy, beginning with the lowest ranked rules and moving upwards. This process is repeated iteratively until the predictive accuracy achieved by the remaining rules is lower that of the initial set. The final set is then the set from the previous iteration, which will have an accuracy that is equal to or higher than the original set. In comparison to the ruleset generated by the bottom-up approach, this final set from the top-down approach covers a broader range of cases, though it may have either the same or slightly lower accuracy and includes a greater number of rules [26].

Whether utilizing a bottom-up or top-down scheme, the rule extraction phase leads to a ruleset with better prediction power than the one produced in the rule refinement phase. In the final ruleset, each rule has an associated weight that serves as its vote during prediction, mimicking the procedure of a random forest. In cases where there is a tie between classes, the prediction is determined by priority, in which the class with the highest priority is chosen. Each class's priority is determined by the accuracy and coverage of the rules that predict the class, the importance of the related attributes, and the number of such rules and attributes [26].

3.3 Modifications

To address what we believe are limitations associated with the methodology utilized in [26], we propose RandomForestRuleExtractor (RAFREX), which features several modifications to the ERRF algorithm. These modifications involve removing the rule ranking scheme, integrating rules based on accuracy rather than rank, and utilizing more integration techniques. Through these modifications, we aim to reduce the complexity of both the model and the resulting ruleset.

Our first modification involves removing the rule ranking scheme. In the approach discussed in [26], each rule in each tree receives a ranking based several metrics in a sequential manner. However, $VI^t(X_i)$ in particular is computationally expensive to compute. Additionally, the actual ranking that each of the rules receive is rather arbitrary, as it only reflects its rank in an individual tree, and does not accurately represent its importance in the overall ruleset. We propose the removal of the rule ranking scheme in order to improve the efficiency of the algorithm, and to remove the restriction the ranking system imposes on the rule integration scheme.

Therefore, given the removal of the rule ranking scheme, instead of integrating rules based on rank, we integrate them based on accuracy. In order for two rules to be integrated, they must meet a user-defined accuracy threshold. Additionally, the resulting rule must also meet a user-defined accuracy threshold in order to replace the integrated rules. If the new rule does not meet the accuracy threshold, then the two original rules are used rather than the resulting rule. This approach ensures that the predictive power of the integrated rules is better preserved in the resulting rule, and that the ruleset remains highly accurate.

Furthermore, the rule integration scheme as described in [26] is updated in our version in order to further refine and reduce redundancy within the ruleset. We do so through the modification of one of the existing modification techniques described in [26], as well as the inclusion additional integration techniques:

```
Input: a raw ruleset derived from an original random forest
Output: a resulting interpretable refined ruleset
Phase 1: Rule Refinement
1.1. Initializing a weight of each rule to 1
~~1.2. Rule ranking~~
1.2. Rule integration
  - ~~Combine rules with the same rank~~
  - Refine ruleset by:
     i. Removing redundant conditions
     ii. Removing covered rules- updated
     iii. Removing subsumed rules with same targets
     iv. Removing conflicting rules
     v. Addressing overlapping rules with different
        targets
     vi. Addressing encompassing rules with
         different targets
     vii. Removing duplicate rules
  - Update weights post-merger
Phase 2: Rule Extraction
2.1. Rule extraction in bottom-up or top-down scheme
  - Bottom-up:
     i. Retains top-ranked rules
     ii. Stops when a rule falls below the accuracy
         threshold
2.2. Return a resulting interpretable refined ruleset
```

Figure 3.2: RandomForestRuleExtractor (RAFREX) algorithm

1. **New methodology for integrating covered rules**: To modify the integration technique used in [26], we can use techniques discussed in [55, 56, 58] to combine more rules and further reduce redundancy. There may be rules which have the same number of conditions and predict the same class, but have different values for the conditional tests [55, 56]. These rules can be merged into one. In [55, 56], they "scope" continuous attributes by finding all pairs of rules which have the same number of antecedent conditions and have one *or more* attributes that are the same, but the have different continuous values chosen for the test. When the attribute test is '>' then the smaller of the two values should be used, and when the attribute test is '<' then the larger of the two values should be used in the combined rule [55, 56]. This technique is also used in

[58] in which they describe that they combine rules with ranges of the same attribute by extending the range of continuous conditions into the largest one.

Therefore, in our approach, given two rules that lead to the same prediction and share the same conditions, a new rule can be created so that the ranges of the same attribute can be combined to encompass the largest one. This approach addresses the limitations of removing ‘covered’ rules discussed in [26], by recognizing that a rules might encompass more than one covered condition. Furthermore, our approach differs from that of [55, 56, 58] as we will follow the methodology in [26] and combine the weights of the original rules to form the weight of the new rule, as it provides a prediction them both. Given ruleset **R** containing distinct rules R_1 and R_2 with covered condition “`a ≥ 2.3`” in R_1 and “`b ≥ 6.15`” in R_2:

R_1 before removal: `IF a ≥ 2.3 AND a < 4.2 AND b ≥ 6.03 THEN 1`

R_2 before removal: `IF a ≥ 0.9 AND a < 4.2 AND b ≥ 6.15 THEN 1`

R_3 with w of $R_1 + R_2$: `IF a ≥ 0.9 AND a < 4.2 AND b ≥ 6.03 THEN 1`

2. **Removing subsumed rules:** To address an aspect of redundancy that Thi et al. fail to consider, we remove all subsumed rules from the ruleset. For this type of redundancy, two rules R_1 and R_2 which predict the same class are considered to be redundant if one rule is more specific than the other [57]. In other words, the more general rule is contained within the more specific rule. The subsuming, more general rule, offers the ability to classify more instances with fewer premises than the subsumed rule. Therefore, the subsumed rule can be removed without any impact on the predictive accuracy, because any results produced by the

subsumed rule can also be produced by the subsuming rules. Authors in [58] follow this approach in their work. When comparing rules derived from two different decision trees, if one is more specific than another, they choose to remove the more specific rule from the ruleset and retain the more general one. Saidani et al. [57] choose to address this form of redundancy by iteratively examining each pair of rules to see if one is more specific than another. If so, the more generic rule is removed and the features of the more specific rule are removed iteratively as long as the classification accuracy of the updated rule is increased. This technique aims to remove non-meaningful features from the more specific rule in order to maintain its generalization capability on new data.

In our approach, we opt to use a methodology similar to that discussed in [58]. Given two rules which predict the same class and one is more specific than another, we remove the more specific rule from the ruleset. Unlike in [58], we utilize the weighting mechanism discussed in [26], and increment the weight of the more general rule, as it provides a prediction for both itself and the removed subsuming rule. Given ruleset **R** containing distinct rules R_1 and R_2 in which R_1 is subsumed by R_2:

R_1 before removal: `IF a ≥ 2.3 AND a < 4.2 AND b ≥ 6.03 AND b < 7.11 THEN 1`

R_2 with $w + 1$: `IF a ≥ 2.3 AND a < 4.2 AND b ≥ 6.03 THEN 1`

3. **Removing conflicting rules**: Another major issue associated with rulesets that [26] fails to address is conflicting rules. Conflicting rules are those which have the same premise but the

conclusions contradict [61]. That is, there exists instances that can trigger both rules, but the rules predict different targets [54, 60]. According to [60], these rules should "for obvious reasons" be detected and resolved. The simplest approach to handling conflict, as discussed in [54], is to ignore it and return a "Null decision". As such, we can effectively remove the conflicting rules to simplify the ruleset without affecting the predictive accuracy, as done in [58]. Given **R** containing R_1 and R_2 which have the same antecedents but different consequents:

R_1 before removal: `IF a ≥ 2.3 AND a < 4.2 AND b ≥ 6.03 THEN 0`

R_2 before removal: `IF a ≥ 2.3 AND a < 4.2 AND b ≥ 6.03 THEN 1`

R_3: —

4. **Addressing overlapping conflicting rules**: While directly conflicting rules can be solved by being removed from the ruleset, there exists less extreme cases of conflicting rules that can be addressed through other methods. Another form of conflict involves cases where ranges of the rules overlap, leading to two different predictions. A common approach discussed in [55, 56, 58] involves creating new sub-rules based on the conflicting rules in order to address conflict. We opt to use the approach discussed in [58], in which the overlapping ranges are removed from each of the rules, so that only the non-overlapping regions of the feature space are classified by each rule. This approach is similar to the previous integration strategy, in which conflicting rules which apply to the same instances are removed. Therefore, given two rules that overlap but have different targets, the two rules are modified to create new rules

which no longer overlap and successfully classify the data. This modification aims to improve the accuracy by avoiding situations in which two rules that result in different outcomes can be applied to the same instance. Given **R** containing R_1 and R_2 which predict different outcomes and have overlapping ranges:

R_1 before removal: `IF a ≥ 2.3 AND b > 4.2 THEN 0`

R_2 before removal: `IF a ≥ 2.3 AND b < 6.03 THEN 1`

New R_1: `IF a ≥ 2.3 AND b ≥ 6.03 THEN 0`

New R_2: `IF a ≥ 2.3 AND b ≤ 4.2 AND THEN 1`

5. **Addressing encompassing conflicting rules**: Yet another form of conflict involves rules which are completely encompassed by another. This form of conflict is not as common, but may be seen in cases where one class is contained within another. Here, we can utilize an approach similar to the one discussed in [62], and apply the concept of rule priority to sets of conflicting rules. In our scheme, this involves incrementing the weight of the more specific rule, so that it is correctly classified, rather than receiving an incorrect classification based on the more general rule. Given **R** containing R_1 and R_2 which predict different outcomes and R_1 completely encompasses R_2:

   R_1: `IF a <= 30.76 AND a > -13.58 AND b <= 17.23 AND b > -33.7 THEN target = 0`

New R_2 with incremented weight w: `IF a <= 30.35 AND a > 10.58 AND b <= 0.23 AND b > -28.14 THEN target = 1`

3.4 Overview of Experiments

To assess the proposed modifications, we compare our algorithm to ERRF, RF, along with several directly interpretable algorithms: BRCG, RIPPER, PART, RIDOR, and CART. This section details an overview of the datasets, algorithms, metrics, and evaluation methods we use in our experiments.

3.4.1 Datasets

We utilize several publicly available datasets from the UCI Machine Learning Repository, namely Banknote Authentication, Pima Indian Diabetes, Hepatitis, Indian Liver Patient (ILPD), Ionosphere, Haberman's Survival, Statlog, Blood Transfusion, and Breast Cancer Wisconsin (WDBC). In addition, we also test model performance on two synthetic datasets: exclusive-or (XOR) and concentric rings, visualized in Figure 3.3. The datasets obtained from the UCI repository are commonly used in studies about ruleset classifiers [19, 27, 28, 37, 39]. Several of the datasets from the UCI repository contained missing values, which we dropped during the data preprocessing phase.

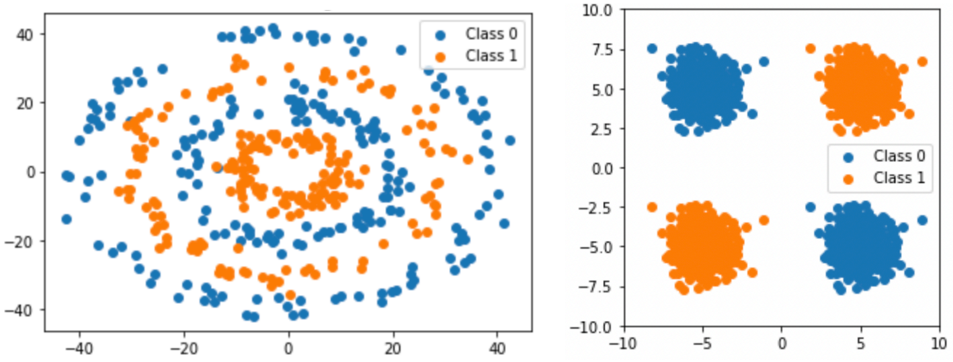


Figure 3.3: Concentric Rings and XOR synthetic datasets

3.4.2 Algorithms

Several rule-based algorithms are used in the experiments, including BRCG [cite], RIPPER [44], RIDOR [46], and ERRF [26]. We also use two tree-based algorithms, CART [43] and PART [45], which were converted into rulesets for evaluation purposes. Furthermore, we use the RF algorithm [19] as a benchmark for accuracy, and similarly convert the forest into a ruleset. We opted to use the 'RuleMaxAcc' version of ERRF, which uses bottom-up extraction, over the 'RuleMaxCover' version, which uses top-down extraction, because it often produces much fewer rules comparatively. Each model was implemented in Python, BRCG and RIPPER were implemented using the AIX360 package [64], and all others were implemented using scikit-learn. Additionally, we evaluate several variations of our model, 'Random Forest Rule Extractor' (RAFREX). The baseline version, 'Integration Threshold + One Rule Ranking Iteration' (TOR), enforces a minimum accuracy threshold rules must meet to be considered for rule integration and completes only one phase of rule refinement; the next version 'Integration Threshold + Multiple Rule Ranking Iterations' (TMR), also employs an accuracy threshold, but performs multiple iterations of rule refinement, provided that the accuracy improves on the validation set or remains the same while having fewer rules. Two other variants, 'No Integration

Threshold + One Rule Ranking Iteration' (NTOR) and 'No Integration Threshold + Multiple Rule Ranking Iterations' (NTMR), do not have an accuracy threshold, and consider all rules for integration; one of these two variants (NTOR) completes only one phase of rule refinement, while the other (NTMR) completes multiple iterations of rule refinement given that the accuracy improves on the validation set or remains the same while having fewer rules. Furthermore, the last variation 'ExtractingRuleRFNoRanking' (ERRFNR) has the same logic as the original ERRF algorithm, but does not employ a rule ranking scheme, and instead uses an accuracy requirement for rule integration rather than rank.

3.4.3 Metrics

The performance of all the models is assessed in terms of both classification and interpretability. The classification performance are assessed using four main measures: accuracy, precision, recall, and F1-score. To assess whether the difference in accuracy scores are statistically significant, we use the Wilcoxon signed rank test.

In terms of interpretability, we use a few different metrics. These measures include the number of rules + the number of conditions in each of the rules, MML, and complexity reduction rate (REDUC). As a proxy for MML as a measure of complexity, we utilize the 'lzma' module from Python's sci-kit learn library. The Lempel-Ziv-Markov chain Algorithm, also known as LZMA, is a recognized data compression technique distinguished by its high compression ratio. The model description is defined as model's total size and the size of the ruleset it produces, and the data description is represented by the size of the misclassified instances. To determine the reduction in the complexity, we base our formula on one from [28], which measures rule reduction rate (REDU). To determine the reduction in number of rules *Nrules* between the original ensemble *H* and the post-hoc model ruleset *R,* the equation is defined as [28]:

$$redu(R|H) = 1 - \frac{Nrules(R)}{Nrules(H)} \tag{1}$$

To better assess how much the post-hoc model simplifies the original ensemble, we opt to determine the reduction in complexity based on the number of rules *Nrules* and the number of conditions in each rule *Nconditions*. Therefore, the equation to calculate REDUC is defined as:

$$reduc(R|H) = 1 - \frac{Nrules(R) + Nconditions(R)}{Nrules(H) + Nconditions(H)} \tag{2}$$

3.4.4 Hyperparameter tuning and model evaluation

The experiments were performed using a stratified 3 × 10 fold cross-validation (CV), scheme, which involves a 10-fold CV repeated three times. For each fold, a 3-fold CV grid search was used to in order to determine the optimal hyperparameters. The search space of the hyperparameters for each algorithm is presented in Table 4.2. The optimal hyperparameters are then used in 10-fold CV, in which results are averaged over 10 iterations in an effort to obtain the most accurate model performance estimates. Results are validated based on the Wilcoxon signed rank test. Individual dataset pairwise comparison is performed to determine whether two algorithms exhibit equivalent performance across multiple iterations. The null hypothesis assumes that the mean difference in performance between any pair of algorithms is zero, implying no significant difference. A significance level of 0.05 is used to determine whether the observed differences are statistically significant. If the adjusted p-value is less than 0.05, we reject the null hypothesis and conclude that there is a significant difference between the two algorithms being compared; otherwise, we conclude that there is no significant difference. To mitigate the risk of Type I errors, falsely detecting a difference when there is none, we use the Holm's alpha correction and multiply the p-value obtained from each Wilcoxon test by the total number of comparisons made.

Algorithm	**Parameters**
CART	criterion ∈ {gini, entropy}, max_depth ∈ {None, 10, 20}, min_samples_split ∈ {2, 5, 10}, min_samples_leaf ∈ {1, 2, 4}
PART	criterion ∈ {gini, entropy}, max_depth ∈ {None, 10, 20}, min_samples_split ∈ {2, 5, 10}, min_samples_leaf ∈ {1, 2, 4}
RIDOR	max_depth ∈ {5, 10, 20}, min_samples_split ∈ {2, 5, 10}, min_samples_leaf ∈ {1, 2, 4}
RIPPER	d ∈ {32, 64, 128}, k ∈ {1, 2, 3, 4}, pruning_threshold ∈ {10, 20, 30}
BRCG	lambda0 ∈ {0.01, 0.001}, lambda1 ∈ {0.01, 0.001}, iterMax ∈ {50, 100, 500}, genMax ∈ {50, 100, 500}, K ∈ {5, 10, 20}
ERRF	extraction_accuracy_threshold ∈ {0.85, 0.90, 0.95}
TOR, TMR, NTOR, NTMR, ERRFNR	integration_accuracy_threshold ∈ {0.85, 0.90, 0.95}, retain_accuracy_threshold ∈ {0.85, 0.90, 0.95}, extraction_accuracy_threshold ∈ {0.85, 0.90, 0.95}
RF	criterion ∈ {gini, entropy}, max_depth ∈ {None, 10, 20}, min_samples_split ∈ {2, 5, 10}, min_samples_leaf ∈ {1, 2, 4}

Table 3.1: Hyperparameter values explored for each algorithm

CHAPTER FOUR

EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Experimental Results Overview

This section details the experimental results and their analyses. We first assess the impact the removal of the rule ranking system has on the results of the ERRF algorithm. We then compare the classification performance of all the models to assess how our several variations of RAFREX compare to ERRF, the directly interpretable models, and to RF. We evaluate the the complexity of each algorithm using two metrics: the number of rules + the number of conditions in each rule and MML based on model compression. Additionally, we assess the reduction of complexity in all the post-hoc algorithms based on the REDUC metric. Furthermore, we evaluate evaluate the accuracy-complexity tradeoffs of all the models.

4.2 Model Variant Results

To determine whether the removal of the ranking scheme has an impact on the ERRF algorithm, we compare ERRF to ExtractingRuleRFNoRank (ERRFNR), which follows the exact logic as ERRF apart from the rule ranking scheme. The results are presented in Table 4.1 and Table 4.2. For each dataset, the ERRFNR algorithm has a much shorter training time compared to ERRF. The MML complexity measure is also reduced for each of the datasets, and there is little impact on the number of rules + number of conditions.

	ERRF	ERRFNR
Banknote	486.67	3.15
Diabetes	7630.65	8.95
Hepatitis	2.86	0.19
ILPD	3519.33	0.25
Ionosphere	131.97	1.92
Statlog	195.98	2.15
Survival	3114.83	5.22
Transfusion	25313.86	22.31
WDBC	459.54	6.11
XOR Clusters	3606.65	5.46
Concentric Rings	489.32	2.13

Table 4.1: Training time comparisons for ERRF and ERRFNR

To statistically compare the accuracy results of the two models, we used the Wilcoxon signed rank test and determined there are no statistically significant differences in classification performance. Therefore, the removal of rule ranking system has no negative impact on the classification performance and positively impacts the MML complexity and training time.

4.3 Classification Results

To assess the classification performance of the models we consider four measures: accuracy, precision, recall, and F1-score. The results of the accuracy comparisons are presented in Table 4.3, Figure 4.1, and Figure 4.2. In Table 4.3, an arrow next to each result indicates whether the difference in accuracy is statistically significant compared to ‘Integration Threshold + One Rule Ranking Iteration’ (TOR), with no arrow indicating no statistical significance. The baseline version, TOR, was able to maintain a statistically similar performance to both the original RF ensemble for ten out of the eleven datasets. TOR, along with the other model variations, were outperformed by BRCG and RIDOR on the Transfusion dataset. However, it’s worth noting that these models also outperformed the original RF ensemble, therefore explaining the performance of our models.

		ERRF			ERRFNR	
	Accuracy	MML	Rules + conditions	Accuracy	MML	Rules + conditions
Banknote	99.23	21075.68	595.50	99.15	17257.36	584.21
Diabetes	74.23	64292.00	2761.03	74.23	50600.56	2761.03
Hepatitis	85.50	6552.37	104.48	86.25	5780.36	107.75
ILPD	69.97	49987.60	2627.37	70.58	38298.20	2626.33
Ionosphere	92.48	16123.24	527.01	92.68	13135.92	513.90
Statlog	80.04	24534.08	1048.07	81.48	19341.92	1040.64
Survival	68.85	33970.64	1638.30	70.07	26634.80	1639.51
Transfusion	74.55	59173.60	1978.60	74.55	59173.60	1978.32
WDBC	96.76	16464.60	743.94	96.20	12666.08	756.85
XOR Clusters	100.00	18684.40	488.50	100.00	15267.20	442.60
Concentric Rings	93.20	26136.20	812.64	93.00	22360.31	825.12

Table 4.2: Accuracy and complexity comparisons for ERRF and ERRFNR based on average accuracy, MML, and number of rules + number of conditions for 3 × 10 fold CV

	TOR	TMR	NTOR	NTMR	ERRF	RF	BRCG	RIPPER	PART	RIDOR	CART
Banknote	99.16 (0.76)	99.03 (0.80)	98.72 ↓ (1.05)	96.54 ↓ (3.67)	99.23 (0.72)	**99.24** (0.79)	97.73 ↓ (1.29)	98.32 ↓ (1.10)	98.08 ↓ (1.15)	96.60 ↓ (1.40)	98.18 ↓ (1.17)
Diabetes	74.48 (5.01)	**74.61** (4.55)	74.19 (4.79)	73.61 (4.39)	74.23 (4.99)	74.59 (4.72)	74.15 (4.74)	62.40 ↓ (10.23)	71.01 ↓ (5.01)	74.10 (4.66)	72.48 (6.83)
Hepatitis	**87.75** (9.68)	85.75 (11.99)	85.38 (10.91)	86.50 (13.19)	85.50 (11.82)	87.50 (11.46)	84.25 (11.94)	79.00 ↓ (14.88)	83.75 (13.98)	79.07 ↓ (13.83)	84.88 (12.16)
ILPD	69.86 (5.57)	69.76 (5.95)	70.24 (5.76)	**70.69** (5.77)	69.97 (5.59)	69.81 (6.01)	69.95 (5.34)	69.79 (6.82)	65.54 ↓ (6.22)	66.91 ↓ (5.55)	62.86 ↓ (7.51)
Ionosphere	92.68 (4.78)	**92.94** (4.61)	92.56 (4.15)	92.93 (3.58)	92.48 (4.74)	92.91 (3.64)	91.25 ↓ (4.51)	90.46 ↓ (4.52)	88.61 ↓ (5.63)	90.03 ↓ (5.02)	89.09 ↓ (5.17)
Statlog	80.89 (7.34)	79.85 (7.08)	79.66 (6.99)	80.89 (7.33)	80.04 (7.54)	**81.81** (7.51)	75.88 ↓ (9.51)	78.21 ↓ (8.28)	77.35 ↓ (8.28)	75.78 ↓ (8.64)	76.82 ↓ (5.77)
Survival	71.34 (8.78)	71.30 (8.60)	**71.66** (9.37)	69.72 ↓ (8.56)	68.85 ↓ (7.62)	71.50 (8.06)	70.39 (7.88)	71.98 (6.90)	67.05 ↓ (8.00)	71.16 (7.74)	67.46 ↓ (8.72)
Transfusion	75.60 (4.60)	74.70 (4.57)	75.04 (4.34)	74.19 (5.45)	74.55 (4.88)	76.58 (4.39)	**78.14** ↑ (4.34)	73.05 (5.37)	75.64 (5.42)	77.62 ↑ (4.91)	75.89 (4.49)
WDBC	96.68 (2.14)	96.57 (2.12)	95.97 (3.64)	95.91 (4.17)	**96.76** (2.00)	96.68 (2.24)	95.73 ↓ (2.58)	94.41 ↓ (2.82)	94.42 ↓ (2.50)	95.31 ↓ (2.33)	94.32 ↓ (2.88)
XOR	99.90 (0.29)	99.87 (0.32)	99.91 (0.33)	99.95 (0.21)	100.00 ↑ (0.00)	**100.00** ↑ (0.05)	98.55 ↓ (5.33)	99.66 ↓ (0.39)	99.32 ↓ (0.77)	92.07 ↓ (16.46)	99.25 ↓ (0.74)
Concentric Rings	92.03 (4.18)	87.95 ↓ (5.47)	88.80 ↓ (5.08)	93.13 (4.67)	**93.20** ↑ (3.69)	92.62 (3.99)	69.30 ↓ (7.48)	85.90 ↓ (6.34)	92.15 (4.43)	78.38 ↓ (7.98)	92.00 (4.14)

Table 4.3: Accuracy and standard deviation comparisons based on averages over 10 iterations of 10 fold CV. Arrows indicates the result of the Wilcoxon signed rank test comparing each variation to 'Integration Threshold + One Rule Ranking Iteration (TOR)', i.e., ↑: better, ↓: worse, and no mark: not statistically different (p-value > .05). **Bold:** Best overall.

Figure 4.1 presents the statistical comparison of the models' accuracy results. The Friedman test rejected the null hypothesis that the algorithms had the same performance, with F_f = 46.814 and p ≈ 0. Therefore, we subsequently performed the Wilcoxon signed rank test with Holm's alpha correction to determine which differences in the algorithms are statistically significant. Thick horizontal lines connect algorithms whose performance differences are not statistically significant, and the number next to each algorithm represents the average rank over all experiments. The original ensemble, RF, achieved the highest average ranking, closely followed by TOR. Figure 4.2 shows these two algorithms are connected by a thick line, signifying that TOR's accuracy is not statistically different from that of the RF model. The accuracy differences among the lower-ranked models—BRCG, CART, RIPPER, PART, and RIDOR—are not statistically significant. BRCG, the top-ranked directly interpretable model, is connected by a thick line to NTMR, indicating that it achieved an overall accuracy performance similar to this RAFREX variant. Furthermore, the performance differences among the post-hoc explanation models were not statistically significant, with the exception of NTMR, the lowest-ranked variant, and TOR, the highest-ranked.

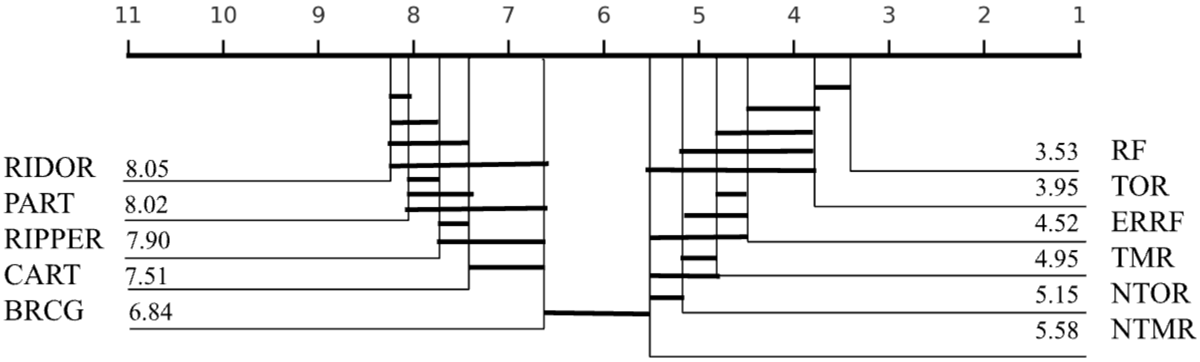


Figure 4.1: Critical difference diagram of accuracy based on Wilcoxon-Holm post-hoc procedure (initial Friedman test result F_f = 46.814, p ≈ 0)

Figure 4.3 presents box-and-whisker plots, displaying the distribution of several datasets for each of the algorithms (see Appendix A). An interesting takeaway is that TOR, the highest-performing RAFREX variant, bears the closest resemblance to ERRF, which ranks third, ahead of the other variants. The two RAFREX variants, NTOR and NTMR, which do not require rules to meet given criteria for integration, are the two lowest ranked variants. This suggests that the ERRF methodology, which restricts which rules can be integrated, does in fact help maintain predictive performance in the integrated rule. While ERRF's strategy aimed at maintaining this performance through a ranking requirement, our method, which utilizes accuracy over a less meaningful rank metric, is not only more intuitive but also more aligned with practical application.

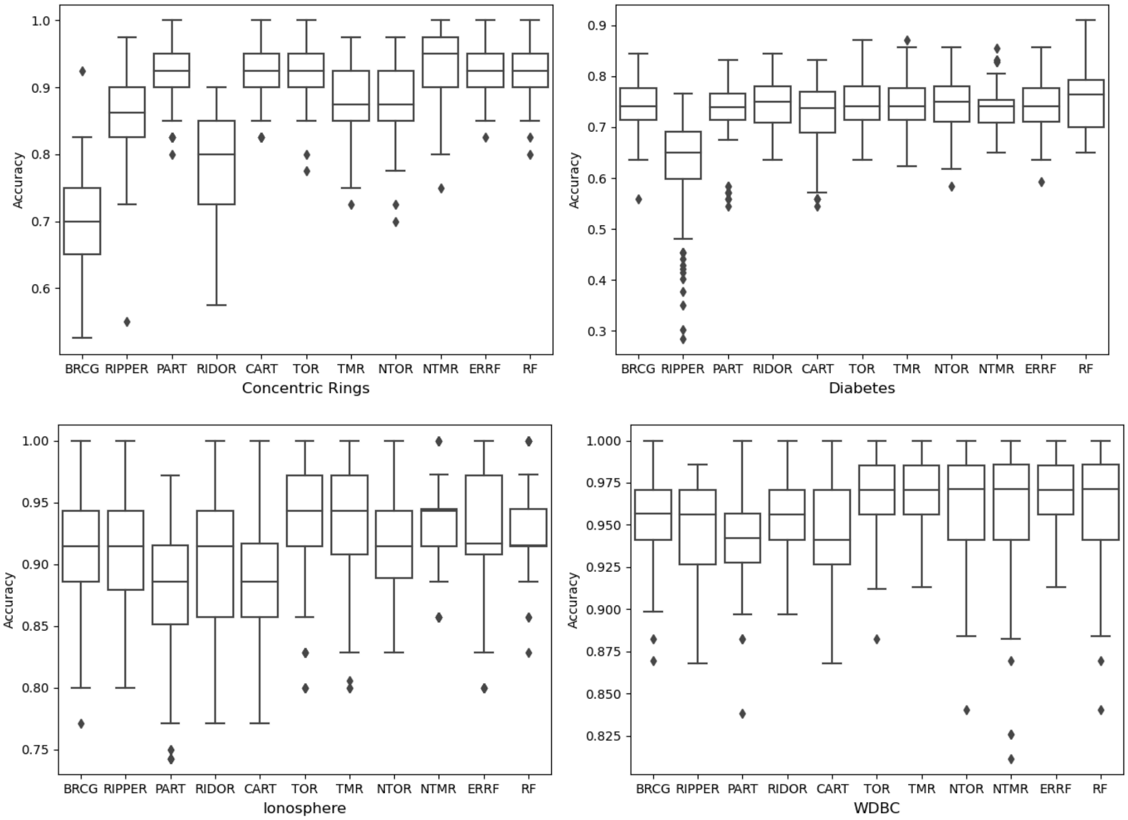


Figure 4.2: Box-and-whisker plots based on accuracy over 10 iterations of 10 fold CV

	TOR	TMR	NTOR	NTMR	ERRF	RF	BRCG	RIPPER	PART	RIDOR	CART
Banknote	99.05 ± 0.87	98.91 ± 0.89	98.56 ± 1.20 ↓	95.87 ± 4.79 ↓	99.14 ± 0.82	99.05 ± 1.05	97.29 ± 1.44 ↓	98.10 ± 1.23 ↓	97.82 ± 1.30 ↓	96.07 ± 1.68 ↓	**99.32** ± 0.77 ↑
Diabetes	59.45 ± 8.48	59.93 ± 7.40	59.23 ± 8.04	58.16 ± 7.83	59.00 ± 7.26	**61.71** ± 8.55	57.87 ± 8.55	47.12 ± 11.89 ↓	56.59 ± 7.29 ↓	59.78 ± 8.94	58.43 ± 9.54
Hepatitis	44.50 ± 44.10	41.47 ± 44.84	34.42 ± 42.83	45.37 ± 45.97	36.07 ± 43.59	**50.72** ± 43.04	47.96 ± 40.64	29.86 ± 39.73	47.00 ± 41.91	38.82 ± 37.95	47.20 ± 41.20
ILPD	80.14 ± 4.38	80.02 ± 4.60	80.29 ± 4.39	80.71 ± 4.40	80.43 ± 4.17	79.88 ± 4.58	**81.93** ± 3.87 ↑	81.25 ± 5.23	75.99 ± 5.45 ↓	78.40 ± 5.16 ↓	73.27 ± 6.79 ↓
Ionosphere	88.99 ± 7.86	**89.63** ± 6.58	88.61 ± 7.30	89.39 ± 5.55	88.91 ± 6.96	89.33 ± 5.77	86.52 ± 7.79 ↓	86.00 ± 7.17 ↓	83.61 ± 8.85 ↓	85.01 ± 7.74 ↓	83.55 ± 9.30 ↓
Statlog	83.03 ± 6.93	81.79 ± 7.83	81.72 ± 7.21	83.02 ± 6.85	82.26 ± 7.15	**83.58** ± 8.23	78.85 ± 9.07 ↓	80.55 ± 11.10	78.94 ± 8.85 ↓	78.62 ± 8.04 ↓	79.53 ± 5.94 ↓
Survival	81.02 ± 6.90	81.23 ± 6.50	**82.17** ± 7.51 ↑	79.96 ± 6.90 ↓	79.51 ± 5.81	81.65 ± 6.08	80.53 ± 5.98	82.07 ± 5.16	77.34 ± 6.60 ↓	80.76 ± 5.80	77.66 ± 7.17 ↓
Transfusion	33.22 ± 10.33	32.73 ± 11.01	33.90 ± 11.29	33.14 ± 10.65	33.53 ± 11.08	39.11 ± 10.40 ↑	41.34 ± 10.36 ↑	8.21 ± 8.04 ↓	36.64 ± 11.61 ↓	**41.87** ± 11.99 ↑	35.98 ± 9.86 ↓
WDBC	95.25 ± 3.15	95.04 ± 3.08	94.90 ± 3.67	94.80 ± 4.56	**95.32** ± 2.96	95.20 ± 3.71	93.74 ± 3.89 ↓	91.73 ± 4.32 ↓	91.83 ± 3.73	93.17 ± 3.58 ↓	91.70 ± 4.22 ↓
XOR	99.91 ± 0.28	99.87 ± 0.31	99.80 ± 0.46 ↓	99.95 ± 0.20	100.00 ± 0.00 ↑	**100.00** ± 0.05 ↑	98.62 ± 4.78 ↓	99.66 ± 0.40 ↓	99.32 ± 0.77 ↓	91.21 ± 20.34 ↓	99.25 ± 0.73 ↓
Concentric Rings	91.99 ± 4.34	87.88 ± 5.87 ↓	88.82 ± 5.19 ↓	93.18 ± 4.64	**93.21** ± 3.83 ↑	92.46 ± 4.13	67.04 ± 8.79 ↓	86.09 ± 6.41 ↓	92.05 ± 4.61	76.47 ± 9.69 ↓	91.83 ± 4.52

Table 4.4: F1-score (μ ± σ) comparisons based on averages over 10 iterations of 10 fold CV. Arrows indicate the result of the Wilcoxon signed rank test comparing each variation to 'Integration Threshold + One Rule Ranking Iteration (TOR)', i.e., ↑: better, ↓: worse, and no mark: not statistically different (p-value > .05). **Bold:** Best overall.

The results of the experiments involving F1-score, precision, and recall comparisons are presented in Table 4.4, Table 4.5, and Table 4.6. Unlike the accuracy results, which were quite stable—having found only two instances where the directly interpretable models achieved statistically better performance than the post-hoc models and RF—the recall results vary a bit more. Directly interpretable models achieved better recall results for a few of the datasets. These results suggest that although the post-hoc models are able to correctly predict a large portion of the outcomes, in some cases they may miss some true positive instances. However, for the majority cases, the post-hoc models achieve precision, recall, and F1-scores which are statistically better than that of the directly interpretable models.

	TOR	TMR	NTOR	NTMR	ERRF	RF	BRCG	RIPPER	PART	RIDOR	CART
Banknote	98.73 (1.44)	98.69 (1.25)	98.50 (1.62)	98.22 ↓ (2.00)	98.88 (1.36)	99.32 (1.04)	97.41 ↓ (2.22)	98.61 (1.33)	98.28 ↓ (1.54)	97.80 ↓ (2.91)	**99.33** ↑ (0.98)
Diabetes	66.44 (9.54)	66.71 (9.77)	65.96 (9.73)	64.87 (9.46)	66.58 (9.38)	66.71 (11.49)	**66.85** (10.45)	51.41 ↓ (13.54)	59.24 ↓ (8.40)	64.49 (9.75)	60.68 ↓ (10.23)
Hepatitis	**90.17** (27.19)	86.50 (33.14)	80.50 (38.01)	85.50 (34.13)	83.75 (34.16)	82.33 (36.50)	65.67 ↓ (40.20)	57.27 ↓ (45.10)	70.83 (41.32)	47.25 ↓ (42.12)	68.17 ↓ (40.21)
ILPD	75.43 (5.73)	75.47 (6.54)	75.94 (6.01)	75.90 (6.35)	75.05 (6.20)	**76.01** (6.55)	71.66 ↓ (5.99)	72.26 ↓ (6.55)	75.81 (6.35)	73.12 ↓ (5.97)	74.46 (7.27)
Ionosphere	92.81 (8.32)	93.21 (6.89)	92.82 (9.04)	**93.80** (7.35)	92.53 (8.19)	93.61 (6.59)	91.49 (9.45)	88.60 ↓ (9.21)	84.77 ↓ (11.09)	90.71 (9.08)	86.89 ↓ (10.33)
Statlog	**82.14** (9.28)	80.57 (9.89)	80.86 (8.22)	82.39 (9.71)	81.23 (9.80)	82.02 (10.57)	77.53 ↓ (10.11)	78.37 ↓ (10.55)	81.07 (11.63)	77.66 ↓ (10.14)	78.55 ↓ (8.10)
Survival	77.67 (9.37)	77.18 (9.04)	75.04 ↓ (9.43)	76.61 (9.45)	76.37 (7.75)	76.83 (8.51)	77.33 (7.95)	76.75 (7.47)	77.51 (8.42)	**78.69** (7.46)	77.51 (8.32)
Transfusion	47.94 (15.44)	44.54 (14.16)	46.02 (15.42)	43.34 (13.64)	45.32 (16.39)	51.94 ↑ (10.31)	**58.45** ↑ (14.38)	37.74 ↓ (35.56)	49.13 (17.20)	54.86 ↑ (15.08)	49.43 (13.89)
WDBC	**96.15** (3.83)	95.71 (3.78)	95.28 (5.51)	94.88 (6.70)	95.98 (3.86)	94.56 (4.12)	94.32 (4.50)	93.04 ↓ (4.87)	92.27 ↓ (4.82)	92.81 ↓ (5.01)	93.35 ↓ (5.34)
XOR	99.91 (0.42)	99.86 (0.46)	99.76 ↓ (0.60)	99.90 (0.39)	100.00 ↑ (0.00)	**100.00** ↑ (0.00)	98.69 ↓ (5.95)	99.68 ↓ (0.57)	99.33 ↓ (0.98)	92.04 ↓ (17.98)	99.31 ↓ (0.89)
Concentric Rings	90.85 (6.61)	87.24 ↓ (8.13)	87.62 ↓ (7.96)	91.38 (6.78)	91.64 (6.34)	**92.94** ↑ (5.83)	72.33 ↓ (10.31)	84.03 ↓ (9.57)	92.18 (6.50)	82.54 ↓ (11.05)	92.05 (5.88)

Table 4.5: Precision and standard deviation comparisons based on averages over 10 iterations of 10 fold CV. Arrows indicate the result of the Wilcoxon signed rank test comparing each variation to 'Integration Threshold + One Rule Ranking Iteration (TOR)', i.e., ↑: better, ↓: worse, and no mark: not statistically different (p-value > .05). **Bold:** Best overall.

	TOR	TMR	NTOR	NTMR	ERRF	RF	BRCG	RIPPER	PART	RIDOR	CART
Banknote	99.39 (0.99)	99.14 (1.30)	98.65 ↓ (1.70)	94.06 ↓ (7.99)	**99.41** (1.00)	99.32 (1.04)	97.23 ↓ (2.32)	97.63 ↓ (1.86)	97.40 ↓ (2.13)	94.51 ↓ (2.63)	99.31 (0.96)
Diabetes	54.78 (10.26)	55.28 (8.75)	54.56 (9.53)	53.68 (9.53)	54.00 (9.29)	**58.18** ↑ (8.75)	51.97 ↓ (9.81)	52.30 (23.94)	55.17 (9.60)	56.79 (11.39)	57.21 (11.69)
Hepatitis	46.42 (44.59)	41.47 (44.85)	43.58 (45.18)	49.95 (46.97)	40.50 (45.84)	50.33 (45.76)	**66.08** ↑ (41.44)	42.00 (45.31)	55.58 (43.40)	57.50 (42.47)	59.25 ↑ (43.48)
ILPD	85.90 (5.76)	85.67 (5.68)	85.58 (5.38)	86.63 (5.30)	87.15 (5.30)	84.66 (5.54)	**96.61** ↑ (6.96)	93.24 ↑ (5.45)	75.99 ↓ (7.05)	85.90 (10.63)	72.27 ↓ (6.99)
Ionosphe-re	86.35 (11.00)	**87.16** (10.14)	86.12 (11.03)	86.08 (8.34)	86.30 (9.33)	86.09 (8.85)	83.27 (11.12)	84.61 (10.28)	83.65 ↓ (10.85)	81.37 ↓ (11.84)	81.55 ↓ (12.03)
Statlog	84.97 (9.20)	84.11 (10.30)	83.53 (10.25)	84.88 (9.44)	84.43 (9.21)	**86.44** (10.32)	81.09 ↓ (10.83)	85.50 (12.92)	78.58 ↓ (11.60)	81.25 ↓ (11.55)	81.54 ↓ (8.96)
Survival	85.97 (8.72)	86.90 (8.10)	92.43 ↑ (9.91)	84.98 (8.91)	83.76 ↓ (8.06)	87.95 (7.31)	85.00 (8.76)	**88.87** ↑ (6.46)	78.12 ↓ (9.22)	83.80 (8.72)	78.66 ↓ (9.71)
Transfusion	26.30 (9.01)	26.91 (10.92)	28.08 (10.73)	28.11 (10.98)	27.77 (10.38)	32.57 ↑ (10.31)	33.61 ↑ (10.68)	5.56 ↓ (5.67)	30.65 (11.10)	**35.08** ↑ (11.96)	29.31 (9.84)
WDBC	**96.15** (3.83)	95.71 (3.78)	95.28 (5.51)	94.88 (6.70)	95.98 (3.86)	96.07 (5.32)	93.46 ↓ (5.94)	90.70 ↓ (5.90)	91.71 ↓ (5.71)	93.88 ↓ (5.43)	90.53 ↓ (6.62)
XOR	99.91 (0.36)	99.88 (0.45)	99.84 (0.63)	99.99 (0.10)	**100.00** ↑ (0.00)	99.99 ↑ (0.09)	98.62 ↓ (3.45)	99.65 (0.62)	99.31 ↓ (0.96)	92.26 ↓ (19.70)	99.21 ↓ (1.09)
Concentr-ic Rings	93.63 (5.68)	89.17 ↓ (7.04)	90.74 ↓ (6.73)	**95.43** ↑ (5.17)	95.26 ↑ (3.83)	92.38 ↓ (5.74)	64.16 ↓ (12.75)	89.31 ↓ (7.96)	92.05 ↓ (5.23)	73.19 ↓ (9.69)	92.07 (6.74)

Table 4.6: Recall and standard deviation comparisons based on averages over 10 iterations of 10 fold CV. Arrows indicate the result of the Wilcoxon signed rank test comparing each variation to 'Integration Threshold + One Rule Ranking Iteration (TOR)', i.e., ↑: better, ↓: worse, and no mark: not statistically different (p-value > .05). **Bold:** Best overall.

4.4 Interpretability Results

The results for the complexity comparisons are displayed in Table 4.7 and Table 4.8. The two complexity metrics are MML based on model compression and the total number of rules + the total number of conditions in each rule. The accuracy-complexity tradeoffs for each algorithm for several datasets are shown in Figure 4.4 (see Appendix B). The directly interpretable models are able to achieve a significantly better performance than the post-hoc extraction algorithms and the RF model in terms of both complexity metrics. An interesting result is that although the post-hoc models, ERRF and the RAFREX variants, exhibit a lower complexity than RF across all datasets in terms of number of rules + number of conditions in each rule, the RF model demonstrates lower complexity in terms of MML based on model compression. This suggests that the RF ensemble contains a large amount of redundancy. Therefore, when compressed, the RF model is actually much smaller than it appears when based on number of rules + number of conditions in each rule for measuring model complexity. Also in terms of number of rules + number of conditions in each rule, the RAFREX variants exhibit lower complexity than ERRF across several datasets; but for others such as Diabetes, Ionosphere, and Statlog, ERRF demonstrates comparable complexity. However, similarly to RF, the RAFREX models consistently achieve lower complexity in terms of MML compared to ERRF. This indicates that ERRF produces rulesets which contains more redundancy than those of the RAFREX variants, suggesting that the additional integration techniques do in fact decrease redundancy in the ruleset.

	TOR	TMR	NTOR	NTMR	ERRF	RF	BRCG	RIPPER	PART	RIDOR	CART
Banknote	16900.20 (880.85)	16419.68 (844.01)	15476.32 (870.19)	14688.76 (1033.72)	21075.68 (1145.36)	11102.64 (546.14)	2795.68 (137.13)	17507.80 (90.82)	1908.24 (92.44)	1765.92 (24.93)	**1509.88** (85.69)
Diabetes	52261.60 (1095.65)	51992.36 (1224.33)	51886.76 (1308.30)	50294.16 (1146.96)	64292.00 (1345.27)	33348.32 (706.88)	4827.28 (70.8.60)	4668.84 (480.77)	3350.44 (64.92)	**1727.12** (60.26)	3577.88 (64.75)
Hepatitis	5919.92 (293.92)	5806.12 (356.28)	5870.56 (376.67)	5864.48 (342.14)	6552.37 (426.28)	4193.52 (207.80)	3014.76 (170.90)	2359.60 (149.79)	1446.88 (24.63)	**1162.40** (41.32)	1507.40 (31.71)
ILPD	40045.80 (1022.99)	39955.84 (1388.59)	39450.72 (1001.58)	39284.92 (996.32)	49987.60 (931.98)	20822.08 (415.76)	4497.04 (323.51)	3525.16 (284.15)	3982.84 (94.76)	**1610.16** (104.73)	3353.36 (81.03)
Ionosphere	13846.36 (655.27)	13768.96 (660.69)	13791.84 (649.62)	13689.20 (582.24)	16123.24 (812.78)	9138.00 (381.15)	7970.20 (386.07)	27942.32 (324.10)	1860.28 (47.49)	**1746.96** (44.72)	1952.44 (44.39)
Statlog	20062.76 (756.66)	19878.16 (758.08)	20001.68 (785.44)	20007.12 (785.24)	24534.08 (1133.53)	10705.44 (291.63)	5656.24 (465.54)	2946.56 (257.87)	2093.76 (46.52)	**1621.48** (67.34)	2286.44 (47.65)
Survival	27383.28 (2478.36)	25804.68 (759.66)	21994.48 (4026.60)	27016.76 (2038.94)	33970.64 (1049.25)	12228.04 (319.71)	2711.40 (127.92)	2235.72 (104.89)	2289.00 (44.97)	**1364.20** (49.15)	2436.76 (52.41)
Transfusior	48155.04 (1148.39)	46879.60 (1189.61)	46858.60 (1128.76)	46369.60 (1579.71)	59173.60 (4306.26)	21589.96 (565.23)	2368.92 (67.32)	1553.36 (91.43)	3190.84 (79.27)	**1471.44** (60.61)	3303.28 (78.66)
WDBC	13388.40 (766.79)	13030.36 (679.79)	12927.44 (896.74)	12881.20 (836.65)	16464.60 (885.01)	7163.28 (339.87)	3933.00 (276.46)	2210.40 (122.90)	1747.16 (43.54)	**1474.52** (49.05)	1814.24 (39.91)
XOR	10707.76 (1536.89)	12972.08 (2159.59)	12853.24 (1970.26)	14873.72 (2025.09)	18684.40 (885.01)	9474.00 (911.79)	2174.56 (56.22)	13996.08 (105.26)	1423.24 (81.92)	**1392.60** (145.31)	1535.24 (81.33)
Concentric Rings	21812.12 (1060.76)	20231.76 (1133.46)	20349.44 (1001.70)	22844.04 (996.29)	26136.20 (1382.70)	15055.24 (729.36)	2655.24 (164.43)	7399.44 (232.87)	2419.48 (72.50)	**1863.56** (119.71)	2568.12 (88.05)

Table 4.7: Complexity comparisons based on average MML over 10 iterations of 10 fold CV. **Bold:** Best overall.

	TOR	TMR	NTOR	NTMR	ERRF	RF	BRCG	RIPPER	PART	RIDOR	CART
Banknote	424.69 (39.33)	384.76 (45.99)	271.21 (33.57)	230.19 (137.78)	595.50 (48.26)	1757.69 (138.55)	**12.95** (1.95)	27.50 (3.07)	47.02 (9.10)	198.52 (13.66)	21.01 (8.65)
Diabetes	2759.53 (132.43)	2745.83 (151.66)	2730.27 (149.37)	2472.93 (123.34)	2761.03 (157.35)	11789.02 (416.66)	**26.34** (7.57)	52.06 (18.15)	209.06 (20.32)	291.60 (37.11)	221.91 (18.14)
Hepatitis	102.15 (19.47)	96.75 (21.87)	103.07 (26.72)	101.46 (22.12)	104.48 (21.41)	428.34 (51.78)	11.53 (1.89)	10.34 (5.62)	**5.18** (2.13)	76.07 (15.30)	5.29 (2.28)
ILPD	2595.86 (146.79)	2705.57 (231.91)	2448.99 (134.01)	2459.04 (135.43)	2627.37 (144.99)	6690.40 (308.31)	**4.39** (6.10)	40.09 (12.17)	646.07 (66.20)	288.95 (75.20)	334.20 (33.48)
Ionosphere	526.53 (70.41)	530.72 (67.60)	507.46 (64.83)	523.59 (64.22)	527.01 (64.30)	1655.14 (119.29)	24.94 (3.29)	**22.11** (4.32)	66.75 (9.17)	370.34 (31.42)	54.29 (7.54)
Statlog	1043.29 (79.63)	1040.66 (97.43)	1031.06 (83.67)	1033.91 (86.28)	1048.07 (89.81)	2079.73 (107.13)	48.43 (6.28)	**36.08** (11.11)	66.38 (9.13)	213.84 (41.93)	74.12 (9.12)
Survival	1243.67 (73.94)	1065.28 (67.53)	341.61 (41.32)	2718.72 (724.29)	1638.30 (84.22)	3106.71 (136.49)	**12.51** (5.38)	57.00 (6.83)	173.67 (17.36)	216.79 (39.25)	181.32 (21.93)
Transfusior	1796.62 (108.76)	2552.18 (652.51)	2626.26 (647.96)	3045.27 (350.90)	1978.60 (105.67)	6059.91 (244.63)	8.57 (2.97)	**6.38** (2.54)	131.48 (16.12)	148.81 (33.16)	128.88 (15.75)
WDBC	763.94 (76.60)	718.77 (76.61)	705.50 (90.43)	702.23 (86.66)	743.94 (66.50)	1198.89 (112.53)	**16.61** (2.54)	46.63 (7.27)	64.98 (10.65)	373.65 (49.48)	56.84 (8.33)
XOR	139.69 (30.92)	169.36 (288.93)	117.02 (223.41)	426.30 (354.33)	488.50 (75.37)	1584.57 (253.61)	**6.27** (0.86)	11.86 (3.22)	21.01 (8.65)	123.28 (50.15)	22.91 (8.40)
Concentric Rings	526.21 (38.06)	382.34 (37.16)	386.79 (40.11)	1588.74 (119.12)	812.64 (51.02)	3752.90 (339.12)	**37.32** (8.85)	42.20 (8.79)	155.90 (14.91)	310.28 (64.01)	157.61 (15.87)

Table 4.8: Complexity comparisons based on average number of rules + conditions in each rule over 10 iterations of 10 fold CV. **Bold:** Best overall.

While directly interpretable models were generally not capable of achieving the same statistical performance as post-hoc models—BRCG and NTMR being exceptions—they often come close and use significantly fewer rules. The evaluation of accuracy-complexity tradeoffs for each algorithm is dependent on the specific domain. In fields where accurate predictions are important, such as healthcare, post-hoc models are likely preferred over directly interpretable ones. The Breast Cancer Wisconsin (WDBC) dataset, for instance, is one which the directly interpretable models achieve significantly fewer rules, but the differences in their F1-scores are statistically significant compared to the post-hoc models. Therefore, when the accuracy of positive predictions and the detection of relevant cases are particularly important, post-hoc models, which achieve better predictive results at the cost of increased complexity, are likely the preferred option.

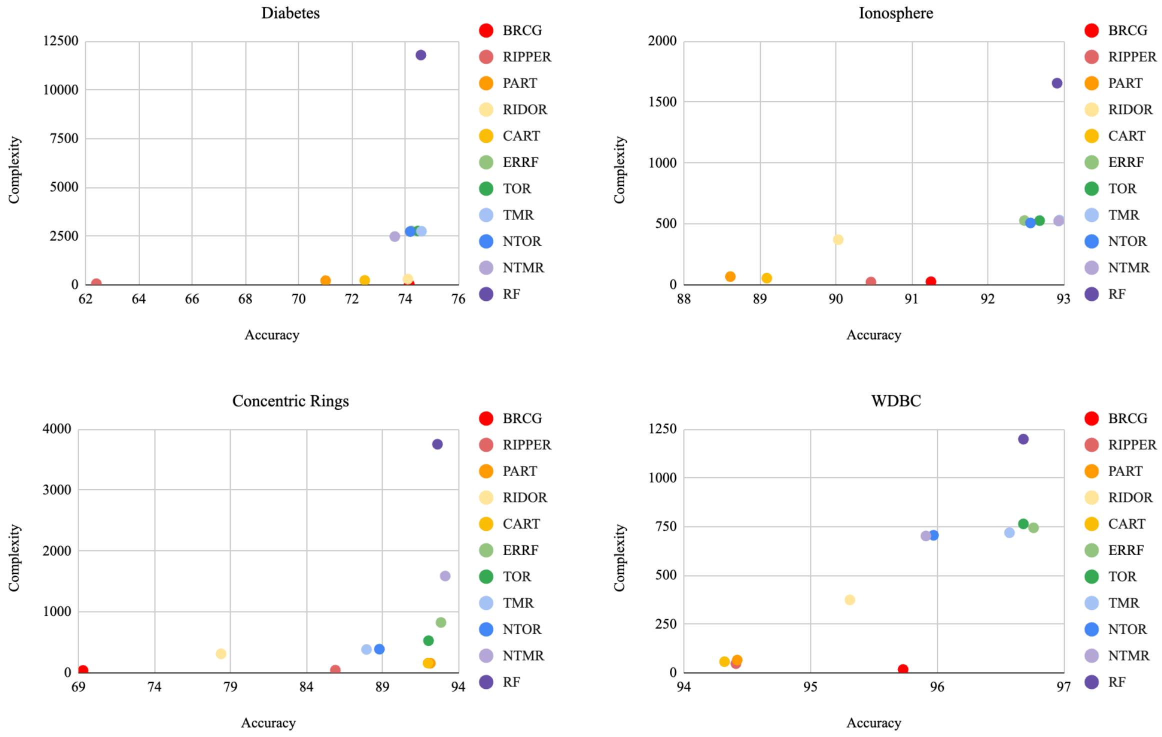


Figure 4.3: Accuracy-complexity tradeoffs based on average accuracy and number of rules + conditions over 10 iterations of 10 fold CV

	TOR	TMR	NTOR	NTMR	ERRF
Banknote	74.19 (3.00)	76.56 (3.63)	83.50 (2.43)	**86.05** (8.13)	63.75 (4.35)
Diabetes	76.56 (1.33)	76.68 (1.33)	76.80 (1.50)	**78.99** (1.29)	76.54 (1.53)
Hepatitis	75.07 (6.19)	**76.58** (5.99)	74.98 (7.07)	75.48 (5.67)	75.17 (5.76)
ILPD	61.13 (2.63)	59.46 (4.06)	**63.31** (2.72)	63.15 (2.90)	60.65 (2.86)
Ionosphere	67.98 (5.21)	67.75 (4.88)	**69.20** (4.30)	68.20 (4.56)	67.93 (4.98)
Statlog	49.72 (4.46)	49.84 (5.28)	**50.31** (4.62)	50.19 (4.53)	49.46 (5.21)
Survival	59.88 (3.13)	65.63 (2.88)	**88.99** (1.35)	12.24 (23.82)	47.11 (3.48)
Transfusion	**70.01** (2.31)	57.88 (10.69)	56.62 (10.77)	49.67 (6.08)	67.29 (2.28)
WDBC	35.67 (9.18)	39.45 (8.99)	41.00 (6.76)	**41.33** (5.92)	37.36 (8.40)
XOR Clusters	85.93 (0.90)	**89.72** (1.42)	89.61 (1.45)	57.32 (5.05)	78.26 (1.34)
Concentric Rings	86.12 (3.64)	88.75 (20.11)	**92.38** (14.74)	71.60 (25.57)	69.20 (8.21)
Average	67.48	68.03	**71.52**	59.47	62.97

Table 4.9: REDUC comparisons based on average complexity reduction rate over 10 iterations of 10 fold CV. **Bold:** Best overall.

Table 4.9 displays the complexity reduction rate (REDUC) comparisons. The results for the TOR model demonstrate that the RF model's complexity, in terms of number of rules + number of conditions in each rule, can be reduced by an average rate of 67.48% and have no statistically significant impact on the average predictive performance of the model. Furthermore, three out of the four variants were able to achieve a high average complexity reduction rate than ERRF, with NTMR's inferior overall performance due to its poor REDUC on the Survival dataset. These results also demonstrate that in most cases only one rule integration phase is often sufficient for effective ruleset refinement. Figure 4.5 and Figure 4.6 presents box-and-whisker plots, displaying the distribution of two datasets for each of the algorithms (see Appendix C).

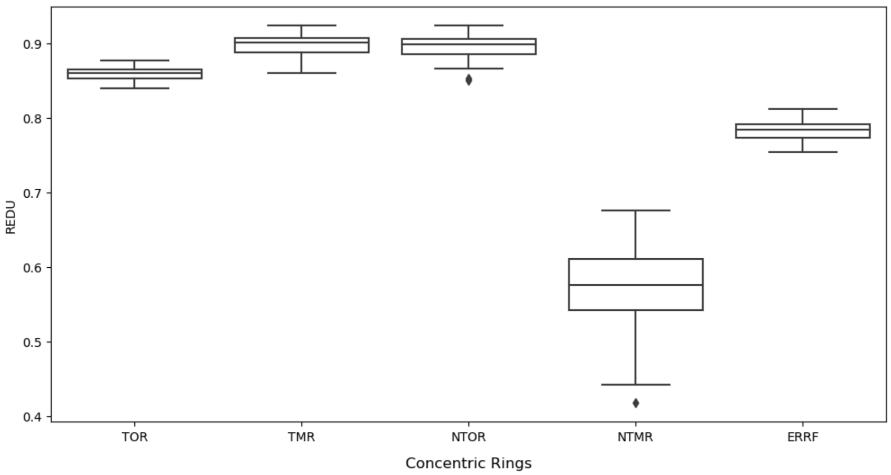


Figure 4.4: REDUC distribution for Concentric Rings dataset based on on average rule reduction rate over 10 iterations of 10 fold CV

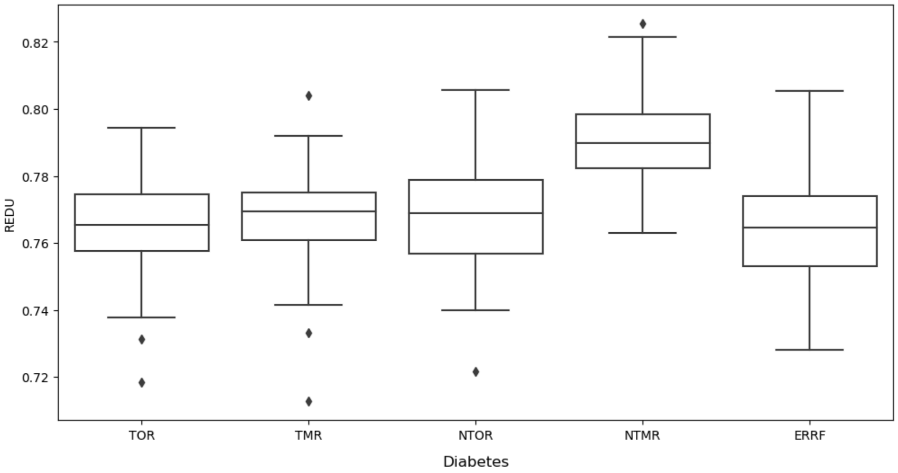


Figure 4.5: REDUC distribution for Diabetes dataset based on on average rule reduction rate over 10 iterations of 10 fold CV

CHAPTER FIVE

CONCLUSION

5.1 Conclusion

Our overall results indicate that the RF model can be greatly reduced in size without adversely affecting predictive accuracy. Additionally, modifications to the ERRF result in significantly shorter training times and less redundancy while also maintaining comparable accuracy. In assessing the impact of the rule ranking scheme on the overall results, we found that ERRFNR reduces training time and MML complexity across datasets without statistically significant differences in accuracy.

The classification performance of each model was assessed through several metrics including accuracy, precision, recall, and F1-score, along with statistical evaluation using the Wilcoxon signed rank test with Holm's alpha correction. The highest-performing RAFREX variant, TOR, was found to achieve an average performance comparable to the RF model, without statistically significant differences, and with an average complexity reduction rate of 67.48%.

Furthermore, complexity comparisons found that the RAFREX models consistently achieved lower complexity based on MML as compared to ERRF. Also, interestingly, RF was found to have lower complexity in terms of MML for each of the datasets, suggesting that the model contains a large amount of redundancy, that when compressed, yields a much smaller result. However, the post-hoc models and RF were outperformed by the directly interpretable models in terms of both complexity metrics. So although the directly interpretable models often fell short in predictive performance, they were able to exhibit lower complexity, making them more understandable. This highlights a fundamental trade-off in these two approaches to model understandability: while directly interpretable models offer simplicity at the expense of lower accuracy, post-hoc models provide superior accuracy at the cost of increased complexity.

5.2 Future Research Directions

The performance of the directly interpretable models in our experiments suggests that achieving a more understandable model is easier if we start with an empty ruleset and methodically add rules, rather than starting with a large ruleset and refining it. One potential method for this approach could involve clustering to naturally derive a set of rules. Traditional rule systems often treat the entire data space monolithically, generating a ruleset based purely on samples, but not considering the inherent structure and nuances in the data. Clustering, however, allows us to observe the samples but also consider their respective groupings. Rather than uniformly generating rules across all samples, it might be more valuable to gather similar rules into meaningful cluster, and then generate rules for each specific cluster. This approach could result in rulesets that better reflect the diversity and heterogeneity within the data, potentially improving both the overall interpretability and predictive performance.

BIBLIOGRAPHY

1. 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 (CSUR)*, *51*(5), 1-42.

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

3. Stella Lowry and Gordon Macpherson. 1988. A blot on the profession. Brit. Med. J. Clin. Res. 296, 6623 (1988), 657.

4. Carolyn Carter, Elizabeth Renuart, Margot Saunders, and Chi Chi Wu. 2006. The credit card market and regulation: In need of repair. NC Bank. Inst. 10 (2006), 23.

5. Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, *16*(3), 31-57.

6. Alex A. Freitas. 2014. Comprehensible classification models: A position paper. ACM SIGKDD Explor. Newslett. 15, 1 (2014), 1–10.

7. Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2018, October). 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). IEEE.

8. Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, *267*, 1-38.
9. Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2020). Explainable ai: A review of machine learning interpretability methods. *Entropy*, *23*(1), 18.
10. Molnar, C., Casalicchio, G., & Bischl, B. (2020, September). Interpretable machine learning–a brief history, state-of-the-art and challenges. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 417-431). Cham: Springer International Publishing.
11. Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access*, *6*, 52138-52160.
12. Mohseni, S., Zarei, N., & Ragan, E. D. (2021). A multidisciplinary survey and framework for design and evaluation of explainable AI systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 11(3-4), 1-45.
13. Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, *8*(8), 832.
14. Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.
15. Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, *1*(5), 206-215.

16. Zhou, J., Gandomi, A. H., Chen, F., & Holzinger, A. (2021). Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, *10*(5), 593.

17. Haddouchi, M., & Berrado, A. (2019, October). A survey of methods and tools used for interpreting random forest. In *2019 1st International Conference on Smart Systems and Data Science (ICSSD)* (pp. 1-6). IEEE.

18. Aria, M., Cuccurullo, C., & Gnasso, A. (2021). A comparison among interpretative proposals for Random Forests. *Machine Learning with Applications*, *6*, 100094.

19. Breiman, L. (2001). Random forests. *Machine Learning*, 45, 5–32.

20. Williams, G. J. (1991). Inducing and combining decision structures for expert systems. Australian National University.

21. Chipman, H. A., George, E. I., & McCulloh, R. E. (1998). Making sense of a forest of trees. *Computing Science and Statistics*, 84-92.

22. Meinshausen, N. (2010). Node harvest. *The Annals of Applied Statistics*, 2049-2072.

23. Deng, H. (2019). Interpreting tree ensembles with intrees. *International Journal of Data Science and Analytics*, *7*(4), 277-287.

24. Wang, S., Wang, Y., Wang, D., Yin, Y., Wang, Y., & Jin, Y. (2020). An improved random forest-based rule extraction method for breast cancer diagnosis. *Applied Soft Computing*, *86*, 105941.

25. Sirikulviriya, N., & Sinthupinyo, S. (2011, May). Integration of rules from a random forest. In *International Conference on Information and Electronics Engineering* (Vol. 6, pp. 194-198).

26. Thi, K. P. L., Thi, N. C. V., & Phung, N. H. (2015, November). Extracting rule RF in educational data classification: from a random forest to interpretable refined rules. In *2015 International Conference on Advanced Computing and Applications (ACOMP)* (pp. 20-27). IEEE.

27. Obregon, J., Kim, A., & Jung, J. Y. (2019). RuleCOSI: Combination and simplification of production rules from boosted decision trees for imbalanced classification. *Expert Systems with Applications*, *126*, 64-82.

28. Obregon, J., & Jung, J. Y. (2023). RuleCOSI+: Rule extraction for interpreting classification tree ensembles. *Information Fusion*, *89*, 355-381.

29. Zhang, H., & Wang, M. (2009). Search for the smallest random forest. *Statistics and its Interface*, *2*(3), 381.

30. Bernard, S., Heutte, L., & Adam, S. (2009, June). On the selection of decision trees in random forests. In *2009 International joint conference on neural networks* (pp. 302-307). IEEE.

31. Latinne, P., Debeir, O., & Decaestecker, C. (2001). Limiting the number of trees in random forests. In Multiple Classifier Systems: Second International Workshop, MCS 2001 Cambridge, UK, July 2–4, 2001 Proceedings 2 (pp. 178-187). Springer Berlin Heidelberg.

32. Van Assche, A., & Blockeel, H. (2007). Seeing the forest through the trees: Learning a comprehensible model from an ensemble. In *Machine Learning: ECML 2007: 18th European Conference on Machine Learning, Warsaw, Poland, September 17-21, 2007. Proceedings 18* (pp. 418-429). Springer Berlin Heidelberg.

33. Gibbons, R. D., Hooker, G., Finkelman, M. D., Weiss, D. J., Pilkonis, P. A., Frank, E., ... & Kupfer, D. J. (2013). The computerized adaptive diagnostic test for major depressive disorder (CAD-MDD): a screening tool for depression. *The Journal of clinical psychiatry*, *74*(7), 3579.

34. Welling, S. H., Refsgaard, H. H., Brockhoff, P. B., & Clemmensen, L. H. (2016). Forest floor visualizations of random forests. *arXiv preprint arXiv:1605.09196*.

35. Moore, A., Murdock, V., Cai, Y., & Jones, K. (2018, June). Transparent tree ensembles. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (pp. 1241-1244).

36. Mollas, I., Bassiliades, N., Vlahavas, I., & Tsoumakas, G. (2019). Lionforests: local interpretation of random forests. *arXiv preprint arXiv:1911.08780*.

37. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1135–1144.

38. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *AAAI Conference on Artificial Intelligence.*

39. S. M. Lundberg and S. I. Lee, ''A unified approach to interpreting model predictions,'' in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 4768–4777.

40. Zhou, Y., & Hooker, G. (2016). Interpreting models via single tree approximation. *arXiv preprint arXiv:1610.09036*.

41. Dhurandhar, A.; Chen, P.Y.; Luss, R.; Tu, C.C.; Ting, P.; Shanmugam, K.; Das, P. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 592–603.

42. Luss, R.; Chen, P.Y.; Dhurandhar, A.; Sattigeri, P.; Zhang, Y.; Shanmugam, K.; Tu, C.C. Generating contrastive explanations with monotonic attribute functions. arXiv 2019, arXiv:1905.12698.

43. Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

44. Cohen, W. W. (1995). Fast effective rule induction. In *Machine learning proceedings 1995* (pp. 115-123). Morgan Kaufmann.

45. Frank, E. & Witten, I. (1998). Generating Accurate Rulesets Without Global Optimization. *Fifteenth International Conference on Machine Learning*, 144-151.

46. Brian R. Gaines, Paul Compton (1995). Induction of Ripple-Down Rules Applied to Modeling Large Databases. *J. Intell. Inf. Syst.* 5(3):211-228.

47. Dash, S., Gunluk, O., & Wei, D. (2018). Boolean decision rules via column generation. *Advances in neural information processing systems*, *31*.

48. Lakkaraju, H., Bach, S. H., & Leskovec, J. (2016, August). Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1675-1684).

49. Williams, J. J., Kim, J., Rafferty, A., Maldonado, S., Gajos, K. Z., Lasecki, W. S., & Heffernan, N. (2016, April). Axis: Generating explanations at scale with learnersourcing and machine learning. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale* (pp. 379-388).

50. Wei, D., Dash, S., Gao, T., & Gunluk, O. (2019, May). Generalized linear rule models. In *International conference on machine learning* (pp. 6687-6696). PMLR.

51. Wang, T., Rudin, C., Doshi-Velez, F., Liu, Y., Klampfl, E., & MacNeille, P. (2017). A bayesian framework for learning rulesets for interpretable classification. *The Journal of Machine Learning Research*, *18*(1), 2357-2393.

52. Ishibuchi, H., & Nojima, Y. (2007). Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning. *International Journal of Approximate Reasoning*, *44*(1), 4-31.

53. Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)*, *9*(5), 272.

54. Williams, G. J. (1991). Inducing and combining decision structures for expert systems. Australian National University.

55. Hall, L. O., Chawla, N., & Bowyer, K. W. (1998). Combining decision trees learned in parallel. In *Working Notes of the KDD-97 Workshop on Distributed Data Mining* (pp. 10-15).

56. Hall, L. O., Chawla, N., & Bowyer, K. W. (1998, October). Decision tree learning on very large data sets. In SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218) (Vol. 3, pp. 2579-2584). IEEE.

57. Saidani, N., Adi, K., & Allili, M. S. (2020). A semantic-based classification approach for an enhanced spam detection. *Computers & Security*, *94*, 101716.

58. Sirikulviriya, N., & Sinthupinyo, S. (2011, May). Integration of rules from a random forest. In *International Conference on Information and Electronics Engineering* (Vol. 6, pp. 194-198).

59. Ligeza, A. (2006). Logical foundations for rule-based systems (Vol. 11). Heidelberg: Springer.

60. Strecht, P., Mendes-Moreira, J., & Soares, C. (2021). Inmplode: A framework to interpret multiple related rule-based models. *Expert Systems*, *38*(6), e12702.

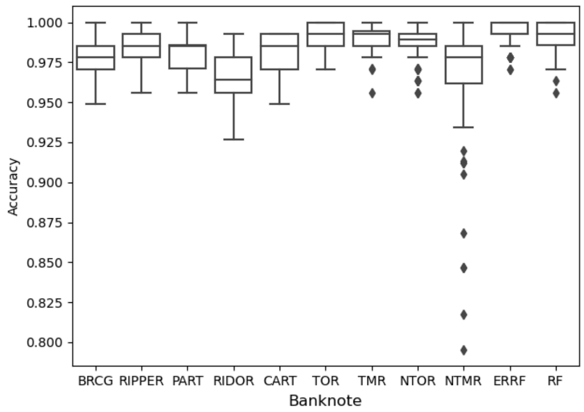
61. Zhang, Y., & Deng, A. (2015, August). Redundancy rules reduction in rule-based knowledge bases. In *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)* (pp. 639-643). IEEE.

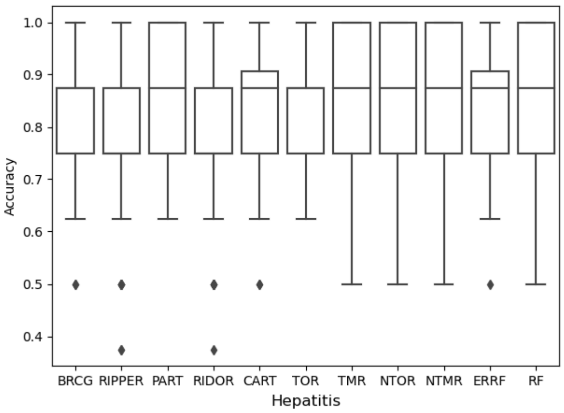
62. Nalepa, G. J. (2018). Modeling with rules using semantic knowledge engineering. Berlin, Germany: Springer.

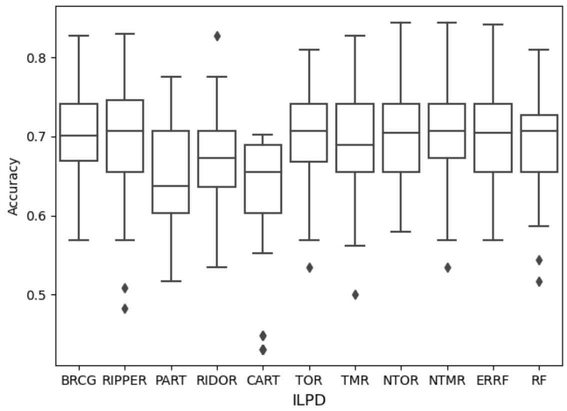
63. Andrzejak, A., Langner, F., & Zabala, S. (2013, April). Interpretable models from distributed data via merging of decision trees. In *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)* (pp. 1-9). IEEE.

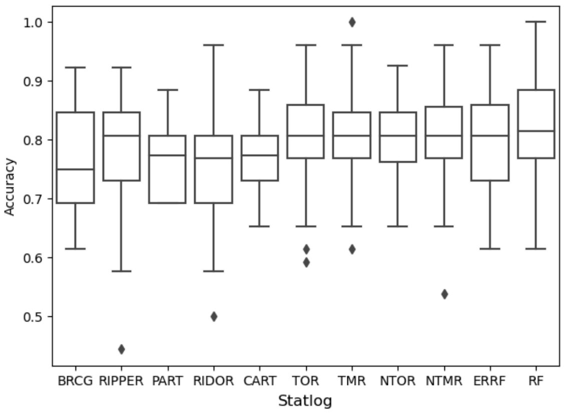
64. Arya, V., Bellamy, R. K., Chen, P. Y., Dhurandhar, A., Hind, M., Hoffman, S. C., ... & Zhang, Y. (2021, January). Ai explainability 360 toolkit. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)* (pp. 376-379).

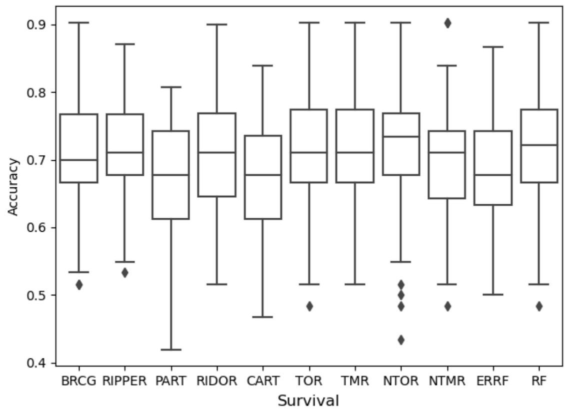
APPENDIX A: ACCURACY BOX-AND-WHISKER PLOTS

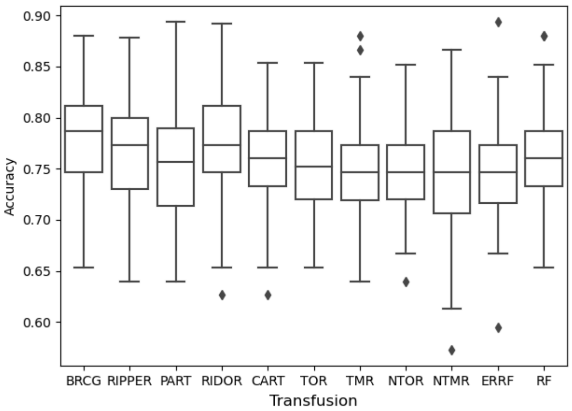


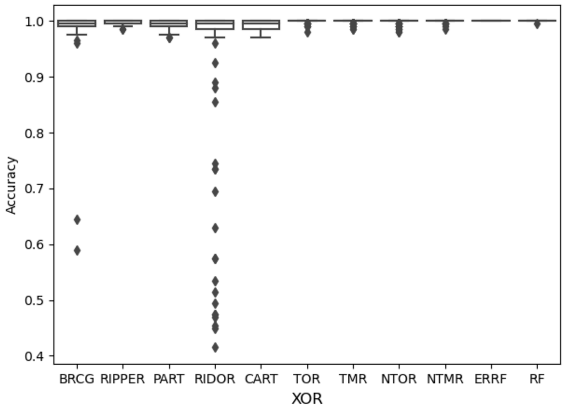




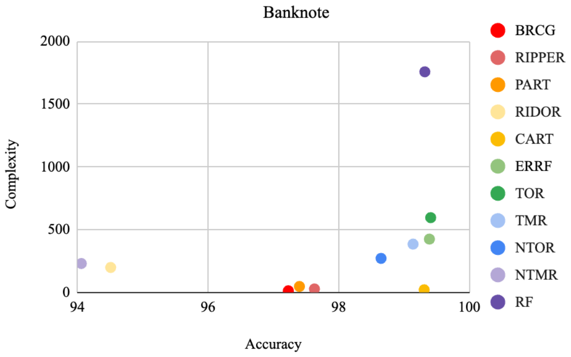


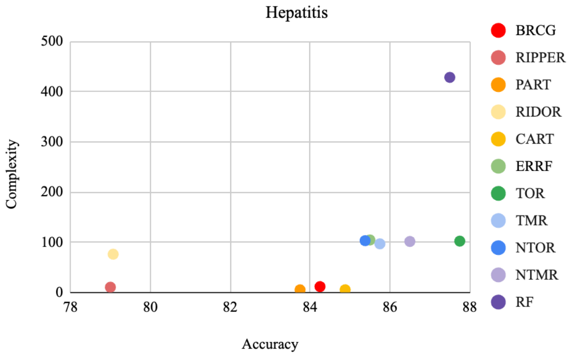


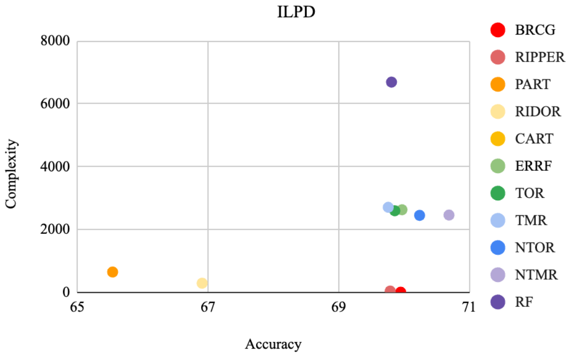


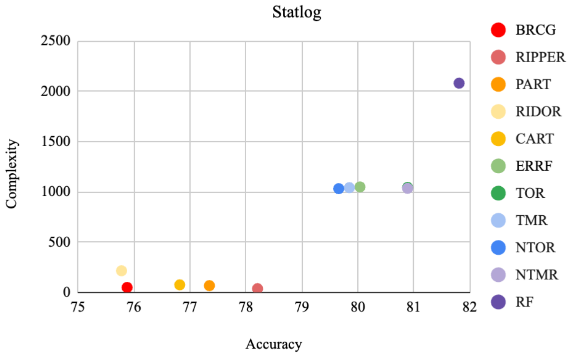

Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
BRCG
RIPPER
PART
RIDOR
CART
TOR
TMR
NTOR
NTMR
ERRF
RF
XOR

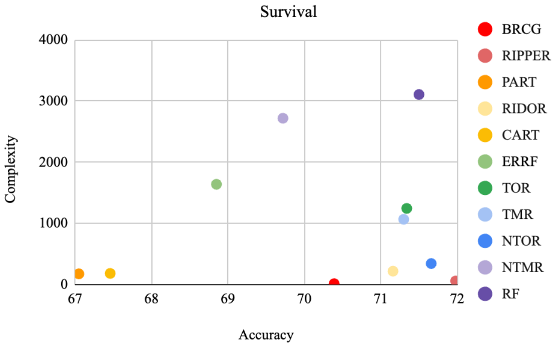
APPENDIX B: ACCURACY-COMPLEXITY TRADEOFFS

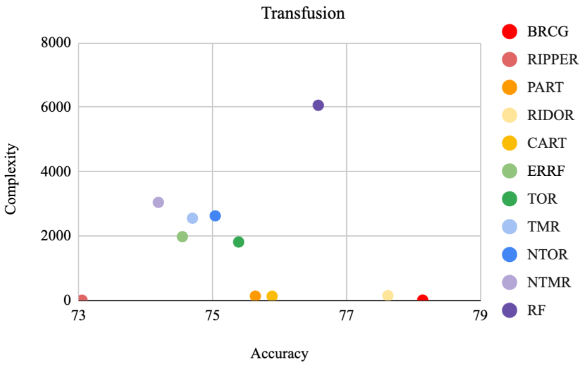


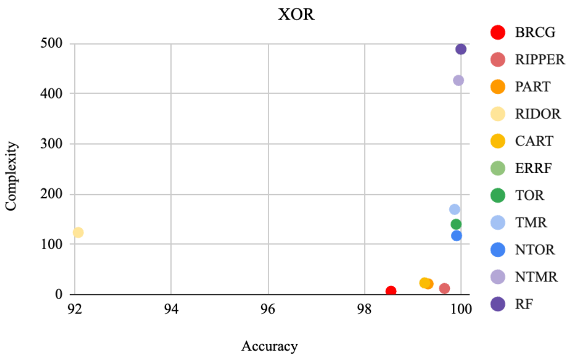




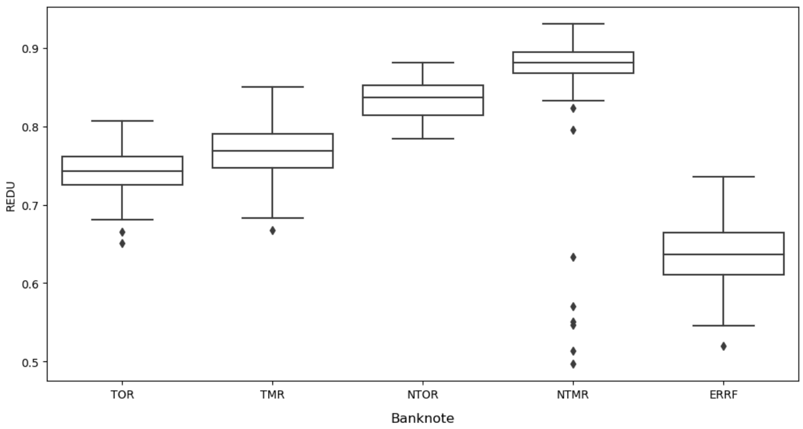


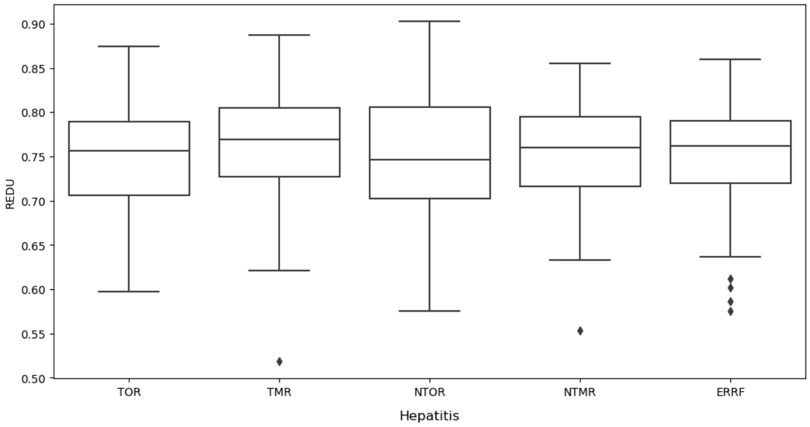


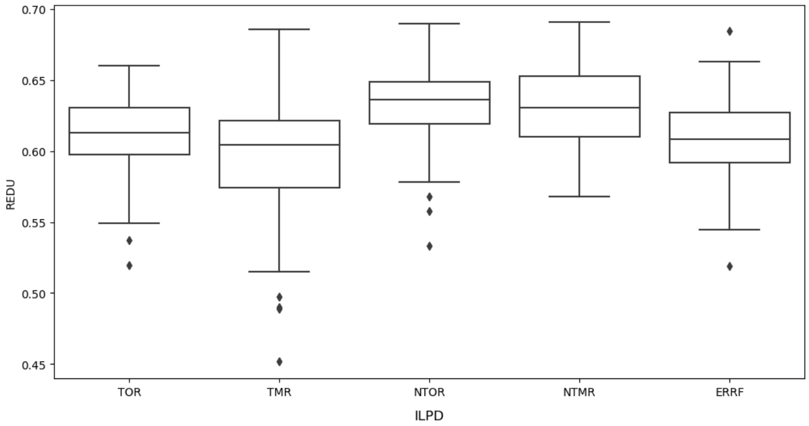


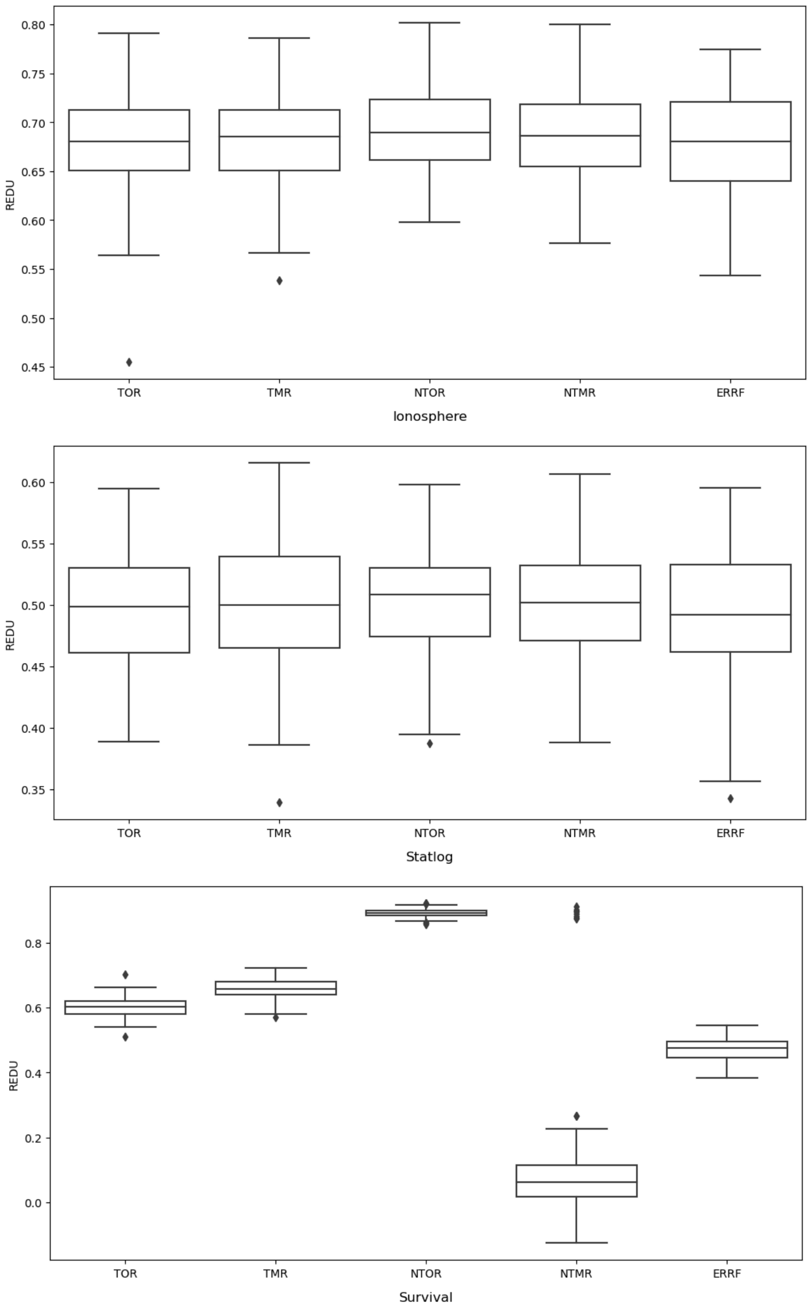

XOR
Complexity
Accuracy
500
400
300
200
100
0
92
94
96
98
100
BRCG
RIPPER
PART
RIDOR
CART
ERRF
TOR
TMR
NTOR
NTMR
RF

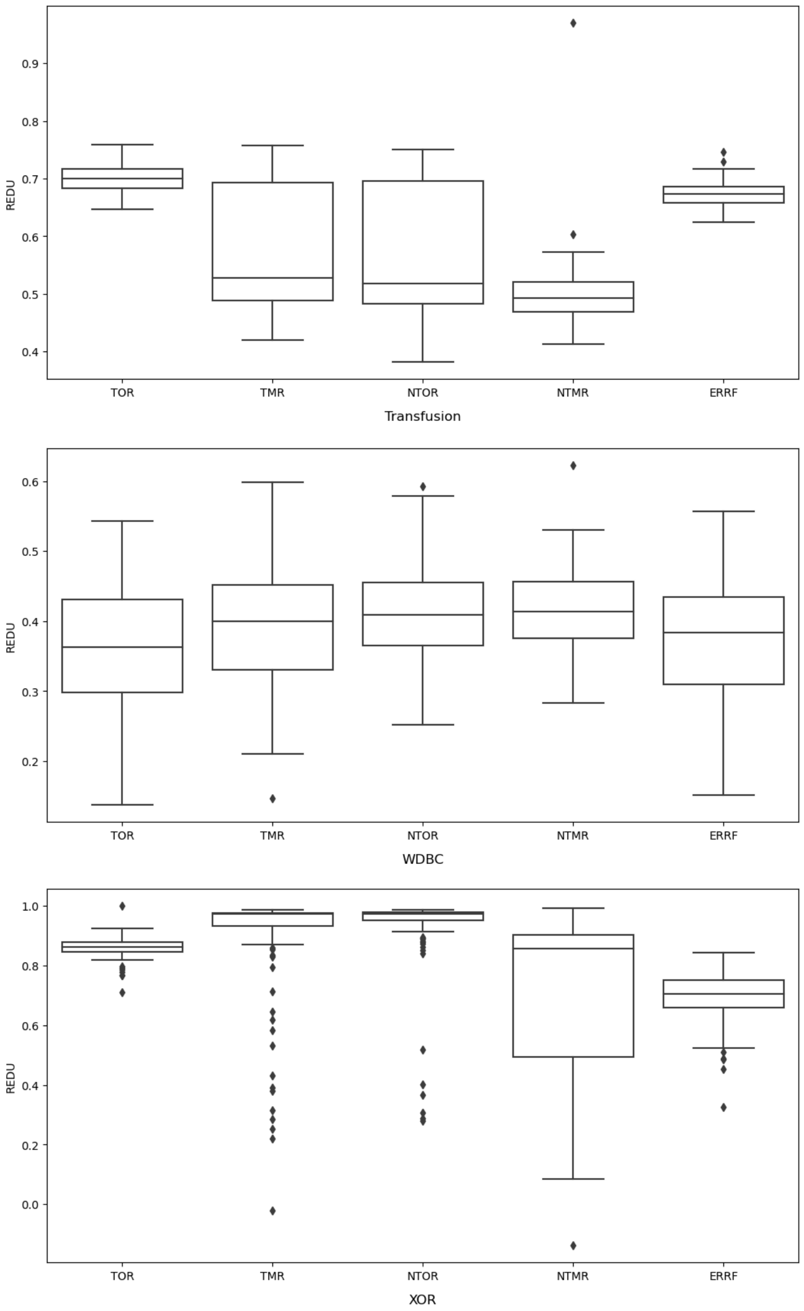
APPENDIX C: COMPLEXITY REDUCTION RATE BOX-AND-WHISKER PLOTS








REDU
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
TOR
TMR
NTOR
NTMR
ERRF
Ionosphere
REDU
0.60
0.55
0.50
0.45
0.40
0.35
TOR
TMR
NTOR
NTMR
ERRF
Statlog
REDU
0.8
0.6
0.4
0.2
0.0
TOR
TMR
NTOR
NTMR
ERRF
Survival


REDU
0.4
0.5
0.6
0.7
0.8
0.9
TOR
TMR
NTOR
NTMR
ERRF
Transfusion
REDU
0.2
0.3
0.4
0.5
0.6
TOR
TMR
NTOR
NTMR
ERRF
WDBC
REDU
0.0
0.2
0.4
0.6
0.8
1.0
TOR
TMR
NTOR
NTMR
ERRF
XOR