Universidade do Minho
Escola de Engenharia

João Pedro Torres Pimentel

Machine Learning Interpretability in a Context of Black Box Regression Models

junho de 2021

Machine Learning Interpretability
in a Context of Black Box Regression Models

João Pimentel

UMinho | 2021

Universidade do Minho
Escola de Engenharia

João Pedro Torres Pimentel

Machine Learning Interpretability in a Context of Black Box Regression Models

Dissertação de Mestrado
Mestrado Integrado em Engenharia Informática

Trabalho efetuado sob a orientação do
Professor Doutor Paulo Jorge Sousa Azevedo

junho de 2021

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros

Acknowledgments

I would like to express my deep gratitude to my supervisor, Professor Paulo Azevedo, for the dedication, motivation and guidance throughout the entire project. Most of all, I am thankful for the opportunity I was given, to extend a promising tool and develop a novel approach to a current problem.

To both my father and grandfather, who always believed in me, even more than I do. I would also like to thank them for the unconditional love, care, support and for helping me whenever I needed.

I cannot help but to thank my dog, Black, for always keeping me company, especially by napping in my lap while I coded and wrote the dissertation. Moreover, for forcing me to go outside to play with her, helping me taking breaks from work. Certainly, the best friend I could ever ask for.

To Joana, for always being there for me, as well as listening to me talking about the problems I encountered and helping me the best way she could. For all the hours hearing me reading the dissertation and suggesting grammatical changes to improve the experience. I cannot express how thankful I am, but know that I am eternally grateful.

Lastly, I am grateful for the great friends I made during the 5 years of the course, with special emphasis on Bruno, Carlos, Carolina, Jaime, Pedro and Rodolfo. I am grateful for you guys putting up with me, for the fun times we shared, the jokes, the games, the meaningful and deep conversations and all the support. Without any doubt, friends to keep around forever.

João Pimentel

Statement of Integrity

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Resumo

Interpretabilidade em Aprendizagem Máquina num Contexto de Modelos de Regressão Caixa Negra

As máquinas têm demonstrado várias vantagens em comparação com os humanos, nomeadamente a reproduzir e escalar tarefas, apresentando velocidade e precisão elevadas. Todavia, nem sempre é possível compreender o funcionamento dos seus algoritmos. Assim, a necessidade de explicar os resultados destes tem vindo a crescer, levando ao aumento da relevância de ferramentas de explicabilidade, já que estas possibilitam a redução das divergências entre a interpretação do modelo e o nível de raciocínio humano.

O principal objetivo desta dissertação passou pelo desenvolvimento de uma técnica *drill-down* para avaliar modelos de regressão caixa negra, considerando interações multivariável no âmbito dos preditores. Assim, propomos EDRs, uma combinação entre DRs e EDPs. De modo a facilitar a sua análise, foram implementadas múltiplas formas de visualização: *boxplots*, histogramas e gráficos de densidade, exibindo distribuições completas, uma visualização em grafo para explorar interações entre preditores e tabelas de desempenho, comparando os quartis de cada distribuição com uma referência. Com base em pontos de corte e uma distribuição de referência, foi ainda efetuada uma extrapolação de contra-factos para regressão.

Aplicaram-se quatro algoritmos distintos a uma gama heterogénia de conjuntos de dados com o intuito de eliminar qualquer potencial enviesamento de modelo. Estas experiências mostraram que as EDRs apresentam vantagens em comparação com os EDPs. O número de gráficos a analisar foi reduzido, já que apenas os subgrupos interessantes são apresentados. Além disso, podem ser detetadas interações compostas por mais de três condições. Foi, também, considerado um caso de estudo, retratando um problema de seleção de modelo. As EDRs mostraram-se cruciais para compreender como os modelos se comportam em relação a combinações específicas de dados e provar que o melhor modelo geral nem sempre é o melhor para certos subgrupos. Deste modo, as EDRs podem ser usadas para escolher um modelo ou para gerar *ensembles*, usando os modelos com melhor desempenho para cada subgrupo.

Apesar das vantagens comparativamente às ferramentas existentes, o uso das regras não esgota o domínio das variáveis, pois não se exibem todas as combinações possíveis, com até três condições. No futuro, pode ser proveitoso estudar uma discretização dos preditores numéricos guiada pelas regras, já que esta etapa depende de técnicas externas. Meta-modelos também devem ser definidos para produzir *ensembles* baseados no desempenho de cada subgrupo.

Palavras-Chave: Aprendizagem Máquina, Desempenho, Interpretabilidade, Regressão

Abstract

Machine Learning Interpretability in a Context of Black Box Regression Models

Machines have shown several advantages compared to humans, namely to reproduce and scale tasks, presenting high speed and precision. However, it is not always possible to understand how the algorithms used work. Consequently, the need to explain the results of these models has been increasing, leading to a boost in the relevance of explainability tools, as these enable the reduction of divergences between the interpretation of the model and the human level of reasoning.

The main goal of this dissertation consisted of developing a drill-down technique to evaluate black box regression models, that considered multivariate interactions within the scope of the predictors. Thus, we propose EDRs, a combination between DRs and EDPs. In order to ease the examination of these, multiple visualization forms were implemented. Namely, boxplots, histograms and density plots to display complete distributions of values, a network visualization to rapidly check interactions of every feature condition and performance tables, comparing the quartiles of every distribution with a reference. Based on the cutting point values and a reference distribution, an extrapolation of counter-factual examples to regression was also implemented.

Four distinct algorithms were applied to a heterogeneous range of datasets in order to eliminate any potential model bias. These experiments showed that EDRs present some advantages in comparison to EDPs. First, the number of plots to analyze is reduced, as only subgroups that differ significantly from the reference and similar subgroups are presented. Also, interactions composed by more than three conditions of feature values can be detected. A case study was considered, applying the developed tools to a model selection problem. EDRs showed to be crucial in helping users to understand how the models behave regarding specific combinations of data. Moreover, it was shown that the best model overall is not always the best for every subgroup. Hence, EDRs can be used to select a model or to generate ensembles, using the best performing models for each subgroup.

Despite the advantages compared to the existing tools, the usage of rules does not exhaust the domain of variables, as not every possible combination of values, with up to three conditions, is displayed. In the future, a rule based discretization of numerical features might be proven fruitful, as this step relies on external techniques. Meta-models are also to be defined to produce ensembles based on performance for each subgroup.

Keywords: Interpretability, Machine Learning, Performance, Regression

Contents

List of Acronyms

2D ALE Second Order Accumulated Local Effects. 20, 21

2D PDP Two Dimensional Partial Dependence Plot. 20, 21

AI Artificial Intelligence. 5, 6

ALE Accumulated Local Effects. 19–21, 31

ANN Artificial Neural Network. xiii–xv, 33, 46, 47, 55, 58, 59, 61–63, 66–70, 72, 74, 76–78, 82, 83, 85

AR Association Rules. 27, 29, 31

c-ICE Centered Individual Conditional Expectation. xii, 17, 18

CAR Classical Association Rules. 27–30

CAREN Class project Association Rule ENgine. 35, 63, 65

CGPA Cumulative Grade Point Average. 65, 70, 72, 84, 97

CV Cross Validation. 35, 63, 65

d-ICE Derivative Individual Conditional Expectation. 18

DARPA Defense Advanced Research Projects Agency. 1

DM Data Mining. 7, 8

DR Distribution Rules. iv, v, xii, xiii, 29–31, 34–37, 39–43, 46, 49, 63, 86

EDP Error Dependence Plot. iv, v, xii–xiv, 12–14, 31, 34–36, 44–53, 55–60, 62, 63, 86, 87

EDR Error Distribution Rules. iv, v, xii–xv, xvii, 35–39, 41, 44–64, 66, 69, 71, 72, 76–87

EU European Union. 1

GBM Gradient Boosting Machine. xii–xv, 34–39, 41–46, 56, 57, 60–63, 66–73, 76–79, 81–85

GDPR General Data Protection Regulation. 1

GPA Grade Point Average. 65

GRE Graduate Record Examination. 65, 84

ICE Individual Conditional Expectation. xii, 16–19, 26, 27, 31

ICI Individual Conditional Importance. 16

IQR Interquartile Range. 46, 61, 67, 68, 77

KDD Knowledge Discovery in Databases. xii, 7, 8

KS Kolmogorov-Smirnov. 30

LHS Left Hand Side. 28, 30

LIME Local Interpretable Model-Agnostic Explanations. 22–24, 31

LIVE Local Interpretable Visual Explanations. 23, 31

LOCO Leave-One-Covariate-Out. 23, 31

LOR Letter of Recommendation. 65, 70, 74, 77, 84, 97

LORE Local Rule-Based Explanations. 23, 31

MAE Mean Absolute Error. xvii, 10, 31, 66, 84

MAPE Mean Absolute Percentage Error. 11

MCR Model Class Reliance. 15, 16, 31

MEDP Multiple model Error Dependence Plots. 14

ML Machine Learning. 1, 3–6, 8, 10, 64

MSE Mean Square Error. 10, 11, 31

OPUS Optimized Pruning for Unordered Search. 29

PDP Partial Dependence Plot. xii, 16–21, 25–27, 31

PEP Parallel Error Plot. xii, 13, 14, 86

PFI Permutation Feature Importance. 15, 16, 31

PI Partial Importance. 16

Q1 First Quartile. 36, 40–42, 44, 46, 48, 51, 52, 54, 56, 61, 65, 67, 70–72, 75–78, 80

Q3 Third Quartile. 36, 40–42, 44, 46, 47, 51, 52, 54–56, 59, 61, 65, 67, 68, 70, 71, 75, 77–80, 82

QAR Quantitative Association Rules. 28–31

REC Regression Error Characteristic. xii, 11, 12, 31

RF Random Forest. xiii–xv, 34, 46, 48, 49, 51, 53, 54, 60–63, 66–70, 72, 73, 75, 77, 78, 80–85

RHS Right Hand Side. 28–31, 36, 37

RMSE Root Mean Square Error. xvii, 10, 11, 66, 84

RROC Regression Receiver Operating Characteristic. 11

SHAP Shapley Additive Explanation. 24, 25, 31

SOP Statement of Purpose. 65, 75, 97

SVM Support Vector Machine. xiii–xv, 34, 40, 49–53, 55, 56, 60–63, 66–70, 72, 74, 76–78, 80–85

TOEFL Test of English as a Foreign Language. 65, 84

VIN Variable Interaction Network. 21

VINE Visual Interaction Effects. 26, 27, 31

XAI Explainable Artificial Intelligence. 1, 5–7

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, the motivation and the main goal behind the development of this dissertation are presented. In addition, the structure of the following chapters is also described.

1.1 Motivation

Compared to humans, machines show many advantages that can be key to solve specific problems. These advantages can be seen in terms of reproducibility, scaling, speed and, in recent times, accuracy [1]. However, recent automatic processes are, in most cases, extremely complex, making it difficult to understand their behavior. These models are denominated black boxes or opaque models, since one can neither understand their internals, nor the reasons behind certain results. Consequently, end users have been pressuring for explainability and transparency, as these models are used to make important and costly decisions, such as in the areas of health and finances [2]. In fact, decision makers will always feel necessity for an explanation in order to fully trust the model, regardless of the accuracy of the model.

This need for explainability results from the fact that advanced Machine Learning (ML) algorithms tend to be non-interpretable, resulting in divergences between the interpretation of the model and the human level of reasoning. This leads to a need of choosing between a more complex model with better results or a simpler model that is not capable of generating results as good as the first. Therefore, the interest in Explainable Artificial Intelligence (XAI) was renewed. Specifically, according to the Defense Advanced Research Projects Agency (DARPA), XAI aims to ”produce more explainable models, while maintaining a high level of learning performance (prediction accuracy); and enable human users to understand, appropriately, trust, and effectively manage the emerging generation of artificially intelligent partners” [3].

It is also important to notice that, due to the European Union (EU) General Data Protection Regulation (GDPR), an individual has the right to explanation when decisions are made in an automated way [4]. As a consequence, the need for transparent and fair algorithms is urgent, being one of the greatest challenges in Machine Learning and Data Science.

1.2 Goal

The main goal of this dissertation consists of studying state of the art methods that help in explaining black box regression models, specifically in numeric prediction. In addition, a drill-down method to study this type of models is to be developed, describing multivariate interactions within the scope of the predictors.

By approaching the problem using a drill down method to analyze areas of the error, anomalous regions can be discovered, i.e., areas where the performance of the model differs from the global performance. Therefore, the algorithms in study can be evaluated in the context of explainability, namely to analyze the fairness of the model in terms of critical subgroups of data.

1.3 Structure of the Dissertation

This dissertation is divided in sex main chapters. The present one introduces the motivation, context, the main goal and contributions of this work. On Chapter 2, the core concepts necessary to understand the subject are presented and explained in detail. Chapter 3 consists on presenting and discussing the state of the art, namely various methods to evaluate models in terms of explainability and its paradigms. Afterwards, in Chapter 4, the details of the implementation and results are characterized and discussed. Extending the latter, Chapter 5 characterizes the process of application of the developed tools to a case study problem. Lastly, Chapter 6 describes the main conclusions and the proposed future work.

Chapter 2

Theoretical Foundations

In this chapter, relevant theoretical topics are presented and explained in order to aid understanding the subject in study. The first subject to be explained is Machine Learning, including a brief introduction to the topic, followed by some concepts relevant to the problem, such as different types of learning, regression and capability to generalize. Then, the importance of Explainable Artificial Intelligence in order to guarantee secure and trustworthy algorithms is addressed. Lastly, the concept of Data Mining in the process of Knowledge Discovery in Databases is described.

2.1 Machine Learning

Machine Learning (ML) is an area of computer science with the purpose of automating tasks. To that end, the algorithms learn from data, based on a mapping function. Therefore, the datasets used in this process are composed by examples, which are sets of values commonly referred as features. These can be defined as quantitative measures with respect to the data and expressed as a vector $x \in \mathbb{R}^n$, with x_i being a feature and n the total number of features [5, 6].

ML algorithms are divided in two main phases: learning and evaluation. The first stage is designated as training, being either supervised or unsupervised, exposing the algorithm to data and originating the model. The second step is known as test and represents the moment when the model is exposed to new data [6].

Moreover, depending on what task the automatic process is designed to perform, the ML task may be labeled as classification, regression, clustering, among other categories [6]. In addition, in order to evaluate the model, i.e., to discern if the model is learning the information correctly, a performance measure, such as accuracy or error rate, is often used. This measure allows to assess the generalization capacity of the model [6].

2.1.1 Types of Learning

Depending on how ML algorithms learn the task at state, they can be broadly classified as supervised or unsupervised learning algorithms. The first utilizes data that contain targets, also knows as labels, decision variables or ground truth, alongside its features. These targets define the real category or value of the input data. Hence, based on the value of the label y and its respective vector x, the algorithm estimates $p(y|x)$, learning to predict the value from the input vector. Examples of supervised algorithms are Support Vector Machines, Decision Trees, Linear Regression, among others [6]. The second kind of algorithms are able to learn relevant information with respect to the structure of the dataset, i.e., by learning useful properties through the features present in the data, not taking into account targets. For this reason, the models learn implicitly (e.g., synthesis or denoising), or explicitly (e.g., density estimation) the probability distribution that characterizes the dataset based on a random vector x [6]. Examples of this type of algorithms are clustering (dividing the data into clusters of similar examples) and dimensional reduction algorithms, such as Principal Component Analysis and Autoencoders [6].

2.1.2 Regression

In machine learning, regression is a type of task where the machine has to predict a numeric value. To do so, the model generates an approximation of an unknown function f, denoted by $\hat{f} : \mathbb{R}^n \longrightarrow \mathbb{R}$, to map the observed data. This task is similar to classification, differing in the format of the output, since in classification the output is a categorical value. To that end, an input, characterized by the vector x is assigned a certain estimated value, denoted by $\hat{y}$, given $\hat{y} = \hat{f}(x)$ [5, 6].

An example of a regression task can be found in the prediction of the exempted claim amount that an insured person will make, or to predict future prices of securities, houses, among others [6].

2.1.3 Generalization Capacity

The central challenge in ML is to make sure that the models perform well when facing new inputs and, not only, when facing the examples they were trained on. For this reason, in order to learn a task correctly, a model should be able to generalize. Consequently, during the training phase, an error measure, known as training error, is calculated. This measure is useful to understand how the training is performing and it should be reduced to its minimum value. However, this is not a simple optimization problem, since the generalization error or test error, i.e., the expected error when facing unseen inputs, is desired to be as low as possible as well. In addition, it is important to make sure that the difference between these two measures is small [6].

When the learning process presents low performance, the model might be suffering from overfitting or underfitting. While overfitting happens when the gap between the errors is large, underfitting occurs when the model is not able to obtain a sufficiently low error value on the training set [6]. Moreover, overfitting is usually due to the attempt to memorize every variation of the training set, striping the model of its capability to generalize [5]. That said, the generalization capacity of a model, i.e., the capability of the model to fit a large variety of functions, is extremely important to determine if a model is capable of predicting correctly or not. Although a high capacity allows the model to solve complex tasks, if the capacity is higher than needed, the model may overfit. Contrarily, if the capacity is low, the model will struggle to solve complex tasks [6]. Figure 1 shows the boundary between overfitting and underfitting zones and how training and test errors evolve with increasing capacity, illustrating the explained behavior.

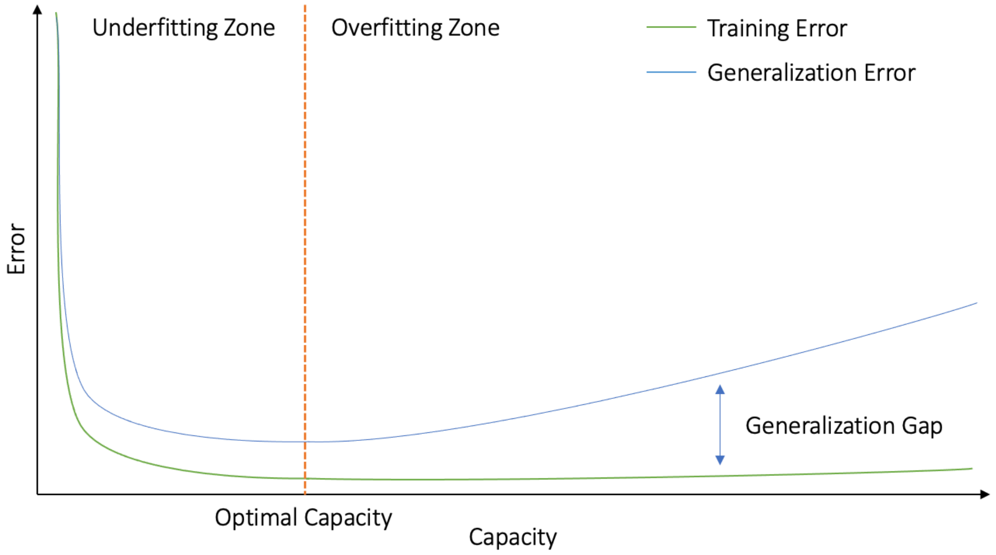


Figure 1: Relation between capacity and error. Adapted from Goodfellow et al. [6].

2.2 Explainable Artificial Intelligence

Explainable Artificial Intelligence (XAI) is a research field that aims to make Artificial Intelligence (AI) systems more understandable to humans. The term was introduced in 2004 by Van Lent et al. [7], in order to describe the ability of their system to explain the behavior of AI controlled entities in applications of simulation games. Despite the term being relatively recent, the problem of explainability has existed since the mid 1970s, with studies to explain expert systems [8]. However, with the enormous advances in ML, the pace of progress towards explainability has slowed down. This was due to the fact that the focus shifted from explaining the models to implement models and algorithms with greater predictive power [9].

Nevertheless, in recent years, this topic has grown in interest, being a direct result of the incorporation of AI and ML across industries and in critical decision making processes. Consequently, there were social, ethical and legal pressure calls for these new AI techniques to be explainable and understandable. This was a result of the lack of detailed information about the chain of reasoning that leads to certain decisions [1, 9, 10].

Because of this, XAI is essential for users to understand, trust and manage AI results. According to Adadi and Berrada [9], the need for explainable AI systems may branch from four reasons, each one capturing different motivations for explainability. The first reason is explain to justify. This is a direct consequence of AI enabled systems yielding biased or discriminatory results [11, 12]. As a result, in order to explain a particular outcome, there is a need for reasons or justifications, rather than a description of the logic of reasoning behind the process in general. The use of XAI systems grants the necessary information to justify results, in particular when unexpected decisions are made. As a consequence, there is a provable form to guarantee that algorithmic decisions are fair and ethical, resulting in the increase of trust in the respective system [9]. This becomes extremely important now that people have the right to explanation in automated decisions [4]. The second reason is explain to control, helping to prevent things from going wrong. By understanding more about the system, there is an increase of visibility over unknown vulnerabilities and flaws, helping to identify and correct errors in non critical situations [9]. Thirdly, explain to improve. Although similar to the latter reason, a model that can be explained and understood is easily improved. As a result, by knowing the reasons behind the system producing certain results, the users will know how to improve it [9]. Lastly, XAI allows for discovery, as asking for explanations is helpful to learn new information and knowledge. For instance, ML algorithms may uncover hidden patterns, allowing people to learn from them [9].

Another important topic is security. As stated by Hall [1], if a ML learning system has been compromised, either by its training data, its outputs altered, or inputs altered to created unpredictable decisions, debugging, explanation and fairness techniques are needed. This is a direct result of the difficulty in determining if a system was compromised, since proving that a model is accurate and fair has very little importance if the data or model can be altered without the knowledge of the user.

Regardless of the fact that explainability is a powerful tool to justify AI based decisions, verify predictions, improve models and gain new insights into the problem in question, not every AI system has a pressing need for it. In reality, if AI systems had to explain every decision they make, it could result in less efficient systems, forced design choices and a bias towards less capable and versatile outcomes. Furthermore, the process to make a system explainable is expensive, requiring considerable resources in every stage of the process [9].

In short, the need for explainable systems depends on the degree of functional opacity caused by the complexity of the AI algorithms and the degree of resistance of the application domain to errors. In regard to the first, if the degree is low, a high level of interpretability is not required. Concerning the second, if the application domain has high resistance to errors, a low level of interpretability is acceptable [9]. For instance, an algorithm used for a promotional campaign has lesser need for explainability than one used to diagnose patients. Thus, domains where the cost of making a wrong prediction is high present a potential need for XAI approaches.

2.2.1 Explainability

According to Gilpin et al. [13], good explanations are the ones that ”you can no longer keep asking why”. When it comes to explainability tools, these can be divided into two bins: model agnostic and model specific. The first consists of tools that can be used regardless of the algorithm in question. Usually, these tools analyze relations between the input and the output, without being necessary to observe the logic of the model. The latter refers to methods that are specific to certain algorithms, only being applicable to a single type or class of algorithm [1]. It is important to highlight that, as model specific techniques tend to use the model to be interpreted directly, they lead to potential more accurate measurements [1].

2.2.2 Interpretability

As reported by Doshi-Velez and Kim [14], interpretability can be defined as ”the ability to explain or to present in understandable terms to a human”. Interpretability can be achieved through intrinsic or *post-hoc* methods. Intrinsic methods, also known as transparent boxes or *ante-hoc* methods, refer to models that were created to be interpretable on their own [2]. Oppositely, *post-hoc* methods refer to the application of tools to analyze pre-trained models, independently of the complexity of the model [15].

Furthermore, among interpretability techniques, two paradigms stand out, distinguishing the methods between local and global explanations. On the one hand, local methods focus on explaining the prediction of a specific instance or group of similar instances. On the other hand, global methods intend to describe the behavior of the model in broad terms [1]. Recently, Britton [16] introduced the concept of regional explanations, defining methods that are intended to interpret a set of instances, with special focus on instances whose behavior differs from the global.

2.2.3 Fairness

Broadly, the study of fairness can be described as disparate impact analysis, i.e., when the predictions of the model are different across demographic groups, beyond some reasonable threshold [1]. With the increase in usage of automated systems in day to day actions, it is important to guarantee that the algorithms produce non discriminatory results, regardless of ethnicity, gender, and other sensitive demographic segments [1].

2.2.4 Accountability

Considering the potential impact on society as a consequence of automated decision making algorithms, it is crucial that these are designed and implemented with some level of accountability. In this context, accountability refers to an obligation to report, explain and justify the algorithm, reducing negative social impact or potential harm [17]. Moreover, as both models and the data used to train the first are created by people, the decisions made by an algorithm, including mistakes with undesired consequences, are, ultimately, a responsibility of a person [17]. Therefore, producing accountable models is to produce models that are prepared for the risks, taking into account multiple principles of XAI, such as explanability, accuracy, fairness, among others [17].

2.3 Data Mining

Data Mining (DM) can be defined as the core of the process of Knowledge Discovery in Databases (KDD). While the latter consists of an automatic process of exploratory analysis and modeling large data repositories, the first utilizes algorithms that explore the data in order to uncover unknown patterns [18]. Thanks to the enormous amount of data produced everyday and its accessibility, DM is a topic of great importance and necessity [18].

Broadly, as characterized by Maimon and Rokach [18], the process of KDD can be described as in Figure 2. Additionally to a previous understanding of the domain of application, steps such as data selection, cleaning and transformation are necessary to increase the reliability of the data and achieve desirable results. These may include handling missing values, noise removal, or handling outliers, i.e., observations that deviate from other observations [19], among other procedures [18]. After these data processing steps, ML techniques can be applied, with the purpose of uncovering hidden patterns or relations. Lastly, as a consequence of the unavoidable alterations that data suffers regularly, either in structure, mining goals, or other causes, this process is dynamic, as it can be repeated as many times as necessary [18].

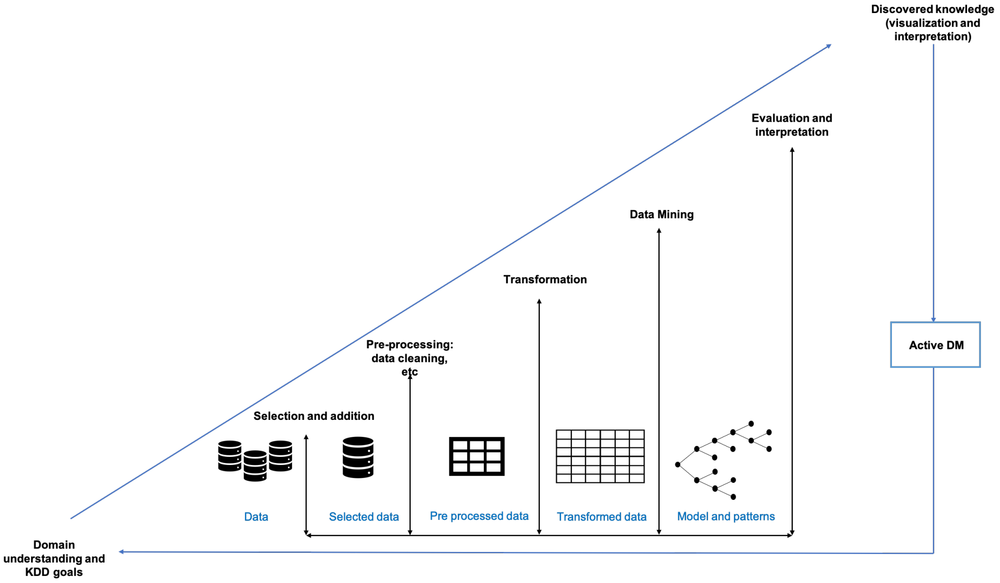

Discovered knowledge (visualization and interpretation)
Evaluation and interpretation
Data Mining
Transformation
Pre-processing: data cleaning, etc
Selection and addition
Active DM
Data
Selected data
Pre processed data
Transformed data
Model and patterns
Domain understanding and KDD goals

Figure 2: General view of the process of KDD. Adapted from Maimon and Rokach [18].

2.4 Summary

Machine Learning is an area of computer science, with the purpose of automating tasks, based on previous data regarding a certain task. It can be divided into two phases: learning and evaluation. The first refers to generating an appropriate function to approximate the data and the second is, generally, used to determine how well the algorithm is capable of fulfilling its purpose. Regarding the tasks, the most common are classification, regression, clustering assignments. Nevertheless, there are others. Moreover, an algorithm can be supervised, i.e., the model learns by observing the desired results, or unsupervised, where the model tries to uncover hidden patterns in the features of the data. It is important to notice that every model needs a good generalization capacity, preventing underfiting and overfiting problems.

Explainable Artificial Intelligence is a research field that has been growing in interest, due to the demand to understand, manage and trust Artificial Intelligence systems. Additionally, to help end users understand some decisions made by the algorithms, this field helps to improve and to make sure the models are safe to use, uncovering possible flaws and alterations made without knowledge. Some of the main concepts associated with this field are explainability, interpretability and fairness. Together, these concepts help to uncover possible flaws and patterns, impossible to discover by looking at the models themselves. Nonetheless, not every system has a pressing need for the inclusion of explainable techniques, as it might be expensive and some areas are not heavily affected by errors, when in comparison to others.

Lastly, similarly to Machine Learning, Data Mining uses available data in order to perform certain tasks. However, in this case, the objective here is to discover knowledge, i.e., uncover hidden relations or patterns that might be helpful to a certain business, for instance. It is crucial that a previous understanding of the domain exists, allowing the user to specify the main goals of the mining process. Then, data has to be gathered and transformed, in order to be processed in the way that suits the goals best, after the mining process. As a consequence of the inevitable amount of alterations that data suffers on daily basis, this process can be repeated as many times, due to alterations in structure, goals, among other causes.

Chapter 3

State of the Art

The present section provides insight on the developed work and achieved progress throughout the years in the interpretability field of ML, as well as important tools to study subgroups of data such as distribution rules. For each method, their advantages and shortcomings are addressed and discussed.

3.1 Performance Analysis

Estimating the performance of a regression model can be summarized as calculating the differences between the true and predicted values, as presented in Equation 1. Here, y_i represents the real value for a certain example i and $\hat{y}_i$ the predicted value regarding the same example, by the algorithm [20].

$$e_i = y_i - \hat{y}_i \tag{1}$$

In terms of representation, this can be achieved using scalar or graphical metrics. On the one hand, scalar methods quantify an estimate of the expected error, usually for the complete model. On the other hand, graphical metrics allow the study of changes in the performance for different conditions. A common example of a scalar metric is the Mean Absolute Error (MAE), presented in Equation 2, where n is the number of samples. Similarly, the Mean Square Error (MSE) differs from the first by utilizing square values of the error, instead of their absolute, as seen in Equation 3. One variation of the MSE is the Root Mean Square Error (RMSE), where the square root operation is applied to the value obtained by the MSE, as in Equation 4 [20].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i| = mean_{i=1,n}(|e_i|) \tag{2}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} e_i^2 = mean_{i=1,n}(e_i^2) \tag{3}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} = \sqrt{mean_{i=1,n}(e_i^2)} \tag{4}$$

Although easy to calculate, these metrics have some problems. For instance, they are scale dependent, i.e., if the task concerns distinct scales or magnitudes, these should not be applied [20, 21]. In order to solve this issue, one can use measures based on percentage errors. Therefore, a division of the difference between the real and predicted values by the real value has to be calculated, as seen in Equation 5 [20].

$$p_i = \frac{|e_i|}{y_i} \tag{5}$$

One example of these metrics can be seen in Equation 6, the Mean Absolute Percentage Error (MAPE). Multiple variations of these error metrics are available to use, relying on different functions, such as the median [20]. Nonetheless, these methods provide a single metric for the entire model, ignoring the possibility of some interactions between predictors being more prone to errors than others and, consequently, not explaining important aspects about the model.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n} 100 \times |p_i| = mean_{i=1,n}(100 \times |p_i|) \tag{6}$$

There are multiple methods to compare the quality of multiple models, by estimating the relative loss of information in order to balance overfitting and underfitting. Examples of these are the Akaike Information Criterion, using the number of parameters as a measure of complexity [22], and Bayesian Information Criterion, utilizing the number of parameters and the number of observations as measures of complexity [23]. However, these methods only evaluate the overall model, similarly to the previous ones.

Regarding graphical methods to evaluate regression models, by providing information that concerns the changes in the performance for different conditions, we can look at examples such as lift charts [24], Regression Error Characteristic (REC) curves [25], REC surfaces [26], Regression Receiver Operating Characteristic (RROC) space [27], among others. For example, REC curves plot the error tolerance in comparison to the percentage of instances predicted under the same tolerance, describing an estimation function of the error cumulative distribution [25]. Following the same idea, REC surfaces add the target value to the graphic, as seen in Figure 3, allowing us to study which types of target values are more prone to certain errors (e.g., small errors) and what errors the model produce for a particular range of target values that are crucial in a specific application [26].

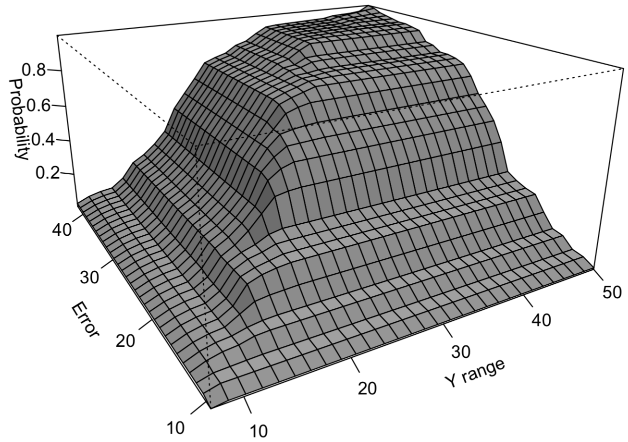


Figure 3: REC surface from dataset Boston (cf. Table 1), trained with a Random Forest model (cf. Table 2).

Even though these approaches produce important information to understand the problem, these only consider the target values to calculate the error and error tolerance. Additionally, these examine the model as a whole, ignoring possible interactions of predictors that may lead to better or worse performance, when in comparison to the overall model. Consequently, Areosa and Torgo [28] proposed Error Dependence Plots (EDPs), establishing a relation between the expected error and the values of a certain predictor variable. These consist of a graphical representation of the expected error and the domain of a predefined feature, using box plots. The error is calculated using cross validation, i.e., randomly permute the data, create k equal sized partitions and, for each one, train the model using all instances but the ones in the partition, calculate the error of the model for the prediction of these instances and add the values to the error estimate [28]. Consequently, EDPs tend to have higher reliability on larger datasets and, generally, on data bins containing more instances [28]. Since presenting the distribution of the error for every possible value of a numeric variable is complex, due to possible lack of repetitions in the data, the values are discretized into meaningful bins. By doing so, it is possible to collect several error values per bin and approximate the distribution of the error. Moreover, the ideal scenario would be to select the bins based on some precious knowledge of the domain. However, as this is not always possible, these can be selected using quartiles of the distribution of the variables. For example, $[0\% - 10\%]$ for extremely low values, $[10\% - 35\%]$ representing low values, $[35\% - 65\%]$ concerning central values, $[65\% - 90\%]$ regarding high values and $[90\% - 100\%]$ in respect to extremely high values [28]. Moreover, for categorical predictors, this process is not needed, as the variables are already discrete. Nevertheless, if too many categories are present, prioritizing some and merging the remaining may be helpful, allowing better visualization [28]. Figure 4 depicts an example of an EDP, for dataset Boston (cf. Table 1), regarding the values of the feature *crim*. For instance, the model tends to produce better results for very low and low values and slightly worse for middle and very high values of *crim*, when compared to the general model.

Considering that EDPs ignore interactions among predictors and, consequently, possible situations that may have an impact on the performance of the model, Areosa and Torgo [28] developed a variant of EDPs to evaluate up to 3 predictors at the time. Conceptually, these are similar to regular EDPs, differing

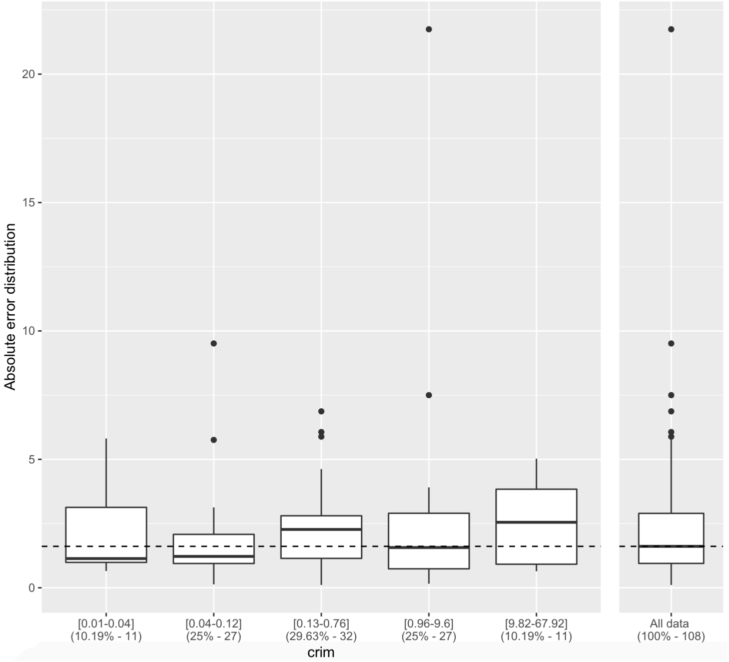


Figure 4: EDP from dataset Boston (cf. Table 1) to analyze the feature *crim*, trained with a Random Forest model (cf. Table 2).

in the fact that these present the error distribution across all possible combinations of bins between the predictors, instead of a single variable [28]. Moreover, assuming that a combination of values between the features is not present in the data, the respective box plot is not displayed [28].

Although important tools, the limitation when plotting multiple variables simultaneously removes some usage from EDPs, since most real world problems tens to have more than 3 predictors. Thus, the method ignores potential interactions between these when analyzing the errors and, possibly, causing misleading conclusions [29]. Because of this, Areosa and Torgo [29] also proposed Parallel Error Plots (PEPs), using a method similar to Parallel Coordinate Plots [30], in which each variable is shown on the *X-axis* and is represented by a vertical bar. This results from a process of uniformization of the scale of the values of every variable, i.e., mapping the original range of values for each variable to $[0-1]$, with 0 corresponding to the minimum and 1 to the maximum values among the instances. Doing this, allows the representation of all values in the same *Y-axis*, as depicted in Figure 5. This process allows the representation of each instance as a line that crosses each bar according to the respective transformed value of the variables [29]. This novel approach informs the user about the dependency between the expected error and the various predictor variables, by trying to show the error profile across the latter, simultaneously. In order to do so, PEPs divide the errors into very high errors, e.g., the top 10% largest errors, and the remaining ones, coloring each line according to the respective error. This approach allows the study of the main objective of PEPs, explaining the conditions that lead to worse performance [29]. However, PEPs have some limitations, namely in the presence of outliers, leading to a suppression of the remaining values. Although, the use of methods that are robust against outliers to scale the values may solve this issue. Moreover, in large datasets or datasets containing an extensive amount of predictors, the visualization might be complex. Therefore, using a smaller subset of the data or the predictors, based on feature relevance, may produce more pleasant results [29].

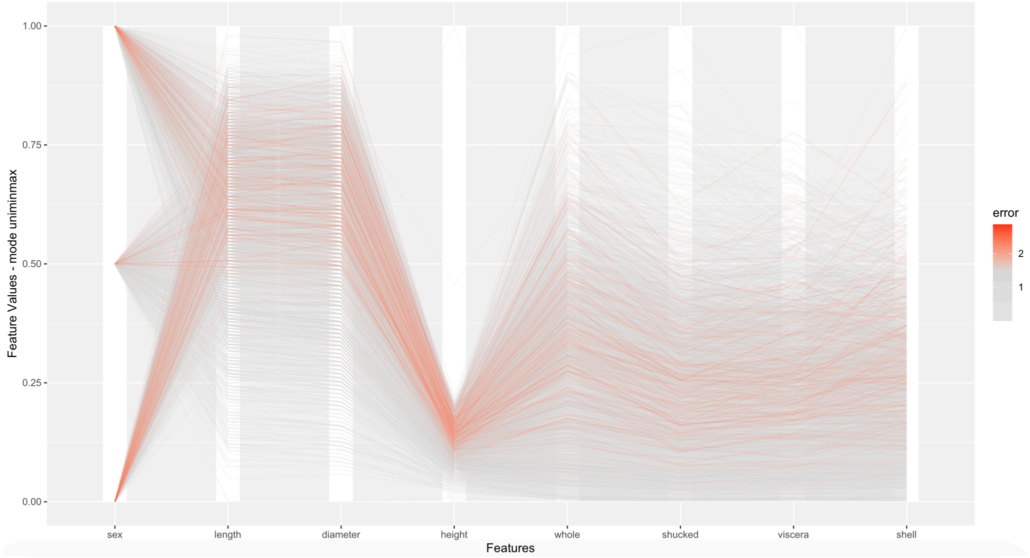


Figure 5: PEP from dataset Abalone (cf. Table 1) to analyze the feature *rings*, trained with a Gradient Boost Machine model (cf. Table 2). The top 10% estimated logarithmic errors are colored in shades of red, with higher errors represented by a greater degree of saturation. Moreover, high errors are prevalent for center to high values of length and diameter, and close to 0.25 for uniformed values of height. Furthermore, the feature sex does not appear to have a strong impact on the performance of the model. Lastly, the remaining features do not display high errors for extremely low values within their range.

Lastly, Areosa and Torgo [29] extended EDPs in order to allow the comparison of multiple models, with Multiple model Error Dependence Plots (MEDPs). Additionally, these are particularly useful to identify whether the best performing model is outperformed in a certain range or category of values and, consequently, to compare models presenting similar overall performance [29]. Similarly to EDPs, MEDPs allow the visualization of interactions between features, specifically between 2 variables with the use of Bivariate MEDPs. This method sets the bins of one of the variables on the *X-axis* and plots the MEDPs across the values of the second feature, representing the conditioned estimated error distribution [29]. By doing so, these present the user with the ability of selecting which model to trust based on desired characteristics of the expected risk [29].

3.2 Interpreting Black Box Models

The following section focuses on existing methods that try to explain the predicted value of a black box model, instead of its prediction error. Respectively, well established *post-hoc* methods, as defined in section 2.2.2, that are used to interpret black box regression models, prioritizing model agnostic techniques. Hence, the tools described evaluate any type of regression algorithm after its training phase. As a consequence, the methods presented receive the trained model as an input alongside other information, if necessary, like training data, containing or not the real target values. Moreover, models that are inherently interpretable, i.e., white-box models, are not approached, as these are not *post-hoc* methods. Although, this type of models can be used as an example of global explanations, being interpreted by analyzing the model itself. Examples of these can be found in decision trees, traditional linear models, business rule systems, among others [1].

It is important to state that, generally, the choice of an adequate explanation method is dependent of the context of the problem, time limitations, personal preferences and the users view in regards to the results obtained using the model.

3.2.1 Global Methods

Global methods are used to describe how the model behaves from an overall perspective [1]. Some evaluate the importance of each feature with regard to the predicted values [31–35]. Other methods try to calculate the effect one or more features have on the predicted value [36–38], while identifying possible interactions between these predictors [36–42]. Moreover, some methods involve emulating the original model, generating a relation between the values of the features and its outputs [1, 43–46]. Hence, the following sections contain several techniques regarding global methods, from feature feature importance, to surrogate models.

3.2.1.1 Feature Importance

Global feature importance quantifies the global contribution of each predictor to the outputs of a model, based on the entire dataset [1]. Typically, these methods are model specific to tree based algorithms. For instance, a simple heuristic rule for these can be calculated from the depth and frequency at which a variable is split in a tree, where higher and more frequent predictors in the tree are more important [1]. Regarding artificial neural networks, usually, variable importance is associated with the aggregated absolute magnitude of the parameters of the model for a specific property of interest [1]. Moreover, for some models, global feature importance can only be obtained by analyzing the relation between the input variables and the target variable. Therefore, these methods represent the magnitude of the relation between the response and a specific input feature of the model, compared to other input variables. However, this may lead to some bias, as some less robust measures can be biased towards large-scale variables [1].

Regarding concrete cases, multiple methods have been proposed for regression models, such as Permutation Feature Importance (PFI) [31], Gini Importance [32], Regressional RelieffF [33], Model Class Reliance (MCR) [34], among others. Gini Importance is a model specific technique, applied to tree based models, measuring the number of nodes split by the predictor, averaged by all trees in the model. By doing so, Gini Importance measures the homogeneity of the feature in question [32]. PFI is also a model specific method, used in tree based models and ensembles, that estimates the importance of a feature based on the prediction error when the values of the feature are permuted, i.e., the order of the values is altered. Thus, in the case of a feature not being important for the prediction, permuting its values does not affect the performance and accuracy of the model drastically [31, 47, 48]. This method does not require the retraining of the model, as the metric is calculated using a perturbed test set or out-of-bag samples as input of the original model [31, 47, 48]. Additionally, the existence of correlated features may lead to over estimation in the importance of those predictors, calculating the metric with possible unrealistic data combinations [49]. Recently, Fisher et al. [34] proposed MCR, a model agnostic version of PFI, based on two possible methods of permutation to calculate model reliance, the importance metric. The first consists of splitting the data in half and exchanging the predictor values between the two groups. The second permutes over all $n!$ possible combinations in the data, where n represents the number of rows in the dataset. Moreover, MCR indicates the upper and lower limits to which a class of models depends on the predictors to predict accurately, using importance estimates based on permutation, providing a robust measure of importance [34]. Proposed by Casalicchio et al. [35], Individual Conditional Importance (ICI) and Partial Importance (PI) are visual tools to help in the visualization of how alterations in the values of a feature affect the performance of the model. These can be used to evaluate the model globally or for individual observations, being variants of PDPs and ICE plots, differing in the fact that these display the feature importance instead of the expected prediction. As a consequence, ICI and PI curves can be used to analyze and compare the feature importance across different subgroups of instances present in the data [35]. For instance, by generating sets of data according to other predictors and calculating a conditional feature importance on this set, may reveal interactions [35, 49]. Lastly, a possibly more accurate alternative would be the use of drop-column importance, calculating the decrease in the performance when a feature is removed from the model. However, this technique is more demanding in terms of computational cost and time, as it requires the model to be trained on the whole, excluding the predictor [48].

3.2.1.2 Feature Effect Plots

Originally proposed by Friedman [36], Partial Dependence Plots (PDPs) are a model agnostic tool that present a visualization tool to study how the response function changes, based on the range of values of a predictor of interest, while averaging out the effects of the remaining features [1, 36, 50]. The core idea behind PDP lies in how these calculate the average predicted value $\hat{f}$. For each value of a specific feature of interest, x_s, the estimation of the prediction is calculated by averaging the predicted value when x_s is fixed and the complementary features, x_c, change over their marginal distribution [36]. Hence, Equation 7 is calculated using the training data of dimension N rows. For each grid value of $x_s = v$, N instances are forged as $< v, x_c >$, where $x_c = \{x_{1c}, ..., x_{Nc}\}$ is a vector containing the values of the predictors that are not the feature of interest, i.e., each instance in the dataset is merged with the feature of interest with value v. Additionally, $P(x_c)$ represents the probability of occurrence of x_c in the training data. Then, the model is queried with the forged instances, generating n instances that are averaged, representing $\hat{f}(v)$ [36].

$$\bar{f}_s(x_s) = \mathbb{E}_{x_c}[\hat{f}(x_s, x_c)] = \int \hat{f}(x_s, x_c)P(x_c)d_{x_c} \tag{7}$$

In order to obtain the full plot, this process is repeated across the desired feature values, as in Equation 8 [36].

$$\bar{f}_s(x_s) = \frac{1}{N}\sum_{i=1}^{N}\hat{f}(x_s, x_{ci}) \tag{8}$$

Moreover, PDPs can be used as a base for other feature importance methods, as proposed by Greenwell et al. [39], using the flatness of the curve as a metric to evaluate the importance of the feature. Besides this, PDPs are a global method in terms of instances, but local regarding input features [1].

The partial functional relationship often varies depending on the values of the remaining features, leading to some loss of information. Because of this, Individual Conditional Expectation (ICE) plots extend PDPs, unveiling individual conditional relationships, previously masked by the averaging factor of PDPs. Additionally, these allow the visualization of individual conditional relationships, plotting an entire distribution of individual conditional expectation functions for a feature x_s, i.e., displaying N estimated curves, one for each set of values in x_c present in the training data, instead of the average partial effect [37]. Hence, each curve defines the conditional relation between x_s and $\hat{f}$ for the values of x_{ci} [37]. For instance, when the predictors in x_c have no influence in the association between x_s and $\hat{f}$, all ICE curves lie on top of each other. If $\hat{f}$ is additive in function of x_c and x_s, the curves are parallel, and when the partial effect of x_s on $\hat{f}$ is influenced by the remaining features, the curves differ from each other in terms of shape [37]. Moreover, plotting an ICE curve for each predictor of a single test observation may allow the study of the sensitivity of the fitted value to alterations in each predictor for the example being studied, similarly to Contribution Plots proposed by Štrumbelj and Kononenko [51].

Sometimes, ICE curves generate over-plotting situations, due to a wide range of interceptions, making it difficult to discern the heterogeneity in the model. In these scenarios, the use of Centered ICE (c-ICE) plots is useful, as these remove level effects and, consequently, unclutter the data displayed [37]. The process to produce these starts by choosing a location x^* in the range of x_s to join every prediction line at that point. For instance, choosing x^* as the minimum or maximum values of the predictor generates the most interpretable plots. As seen in Equation 9, for each curve $\hat{f}^i$ in ICE curves, the corresponding c-ICE curve is given by a subtraction between the original and the fitted model for the selected x^*, where 1 represents a vector of ones of the appropriate dimension. By doing so, the point $(x^*, \hat{f}(x^*, x_{ci}))$ acts as a base case for each curve, e.g., if x^* corresponds to the minimum value of x_s, all curves originate at 0, removing the differences in level, generated from different values in x_{ci}. In the case of x^* being the maximum value, the result is a plot that isolates the combined effect of x_s on $\hat{f}$, maintaining x_c fixed, i.e., the level of each centered curve reflects the cumulative effect of x_s on $\hat{f}$ in relation to the base case [37].

$$\hat{f}^i_{centered} = \hat{f}^i - 1\hat{f}(x^*, x_{ci}) \tag{9}$$

In order to better understand, Figure 6 contains an ICE plot on the left and a c-ICE on the right, similarly to the example provided by Goldstein et al. [37]. These plots examine the relation between the age of houses in a census tract, i.e., $s = age$, and the corresponding median value of the house. On the one hand, the PDP curve (thick curve with yellow outline), in the ICE plot, is mostly flat, displaying a slight decrease in the median prices as age increases. However, the ICE curves contain some observations that present an increase in age and in the median values, describing individual behavior that departs from the average one. On the other hand, the c-ICE plot shows the cumulative effect of age on the output, increasing for some instances and decreasing for others. These differences suggest the existence of interactions between x_s and x_c in the model [37].

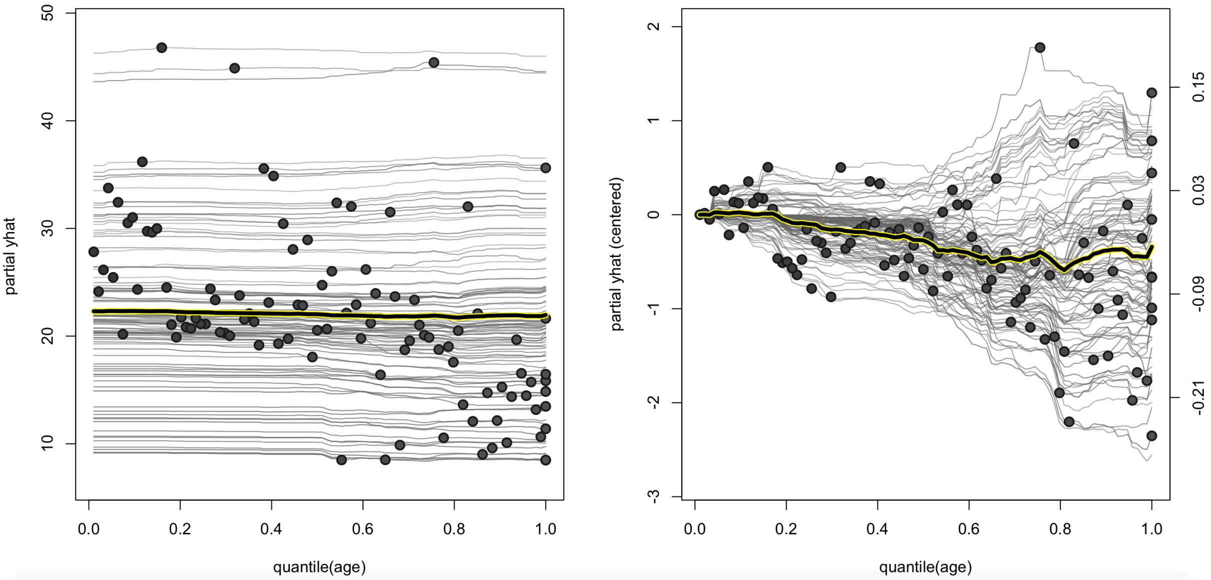


Figure 6: ICE and c-ICE plots from dataset Boston (cf. Table 1) to analyze feature age, trained with a Random Forest model (cf. Table 2). Adapted from Goldstein et al. [37]. The left part of the image contains the ICE plot, where the thick highlighted line represents the PDP. Additionally, for some ICE curves, higher age indicates higher median value. The right part presents a c-ICE with x^* equal to the minimum value of feature age. For some instances, an increase of age leads to increase in the target and decrease to others, suggesting the existence of interactions between age and the remaining features.

Derivative ICE (d-ICE) plots analyze and explore interactions in the partial derivative of $\hat{f}$ with respect to the feature of interest, x_s [37]. Considering that x_s does not interact with the remaining predictors in the model, $\hat{f}$ can be calculated as in Equation 10, meaning that the relation between x_s and $\hat{f}$ does not depend on x_c. Therefore, the N curves in the ICE plot would share a common shape, differing in level shifts depending on the values of x_c [37].

$$\hat{f}(x) = \hat{f}(x_s, x_c) = g(x_s) + h(x_c),\ so\,that\ \frac{\partial \hat{f}(x)}{\partial x_s} = g'(x_s) \tag{10}$$

Hence, there are two possible scenarios. Either the model presents interactions, or these do not exist. The first scenario leads to the derivative curves being heterogeneous, while in the second the curves are equivalent, i.e., the plot displays a single line [37].

Additionally, ICE, c-ICE and d-ICE plots allow for the visualization of interactions between two variables, taking advantage of the use of colors regarding a second feature of interest. Thus, it is possible to analyze the influence of this new predictor in the relationship between x_s and $\hat{f}$. For instance, assuming the second variable being studied is categorical, one can assign one color per class, plotting each prediction line $\hat{f}^i$ using the color of the categorical predictor. If the variable is continuous, the curves may be plotted taking advantage of color shades from light (low value) to dark (high value) [37].

However, PDPs and ICE plots can produce erroneous results when the predictors are strongly correlated, as these require extrapolation of responses across the values of the predictors, producing unrealistic combinations [38]. To solve this problem, Apley and Zhu [38] proposed Accumulated Local Effects (ALE). These compute the alterations in the output for data instances inside small windows of values of the predictor of interest, not requiring the generation of new data and being less computationally expensive [38]. ALE plots are similar to Marginal plots, both avoiding extrapolation by using the conditional density instead of the marginal density. Hence, to better understand ALE plots, it is necessary to understand Marginal plots. As seen in Equation 11, a Marginal plot of the effect of X_s can be defined by a function $f_{s,M}(x_s)$ versus x_s [38].

$$f_{s,M}(x_s) = \mathbb{E}[\hat{f}(X_s, X_c)|X_s = x_s] = \int \hat{f}(x_s, x_c)P_{c|s}(x_c|x_s)d_{x_c} \tag{11}$$

A simple estimate of $f_{s,M}(x_s)$ can be given by Equation 12, where $N(x_s) \subset \{1, ..., n\}$ is the subset of instances for which x_{si} is part of some small and appropriately selected neighborhood of x_s. Additionally, $n(x_s)$ is the number of items in the neighborhood. Although there are more sophisticated kernel smoothing methods to estimate $f_{s,M}(x_s)$, the main problem associated with using $f_{s,M}(x_s)$ to visualize the main effect of X_s occurs when the predictors are correlated. Ergo, using $f_{s,M}(x_s)$ is similar to regressing the target variable onto X_s, while marginalizing over, i.e., ignoring, the second variable [38].

$$\hat{f}_{s,M}(x_s) = \frac{1}{n(x_s)} \sum_{i \in N(x_s)} \hat{f}(x_s, x_{ci}) \tag{12}$$

As a consequence, in the case of the target variable being dependent of x_s and x_c, the function reflects both of their effects, due to the omitted variable bias phenomenon in regression. Contrarily, ALE plots present a method of assessing the main and interaction effects of predictors, avoiding the foregoing problems [38].

Equation 13 represents the local effect of x_s on $\hat{f}$ at (x_s, x_c) [38].

$$\hat{f}_s(x_s, x_c) = \frac{\partial \hat{f}(x_s, x_c)}{\partial x_s} \tag{13}$$

The ALE main effect of x_s is given by Equations 14 and 15, where z_{s0} represents a value near the lower bound of the effective support of P_s, for instance, below the smallest observation of x_s. The choice of this value is not important, as it only affects the vertical translation of the plot of $\hat{f}$ versus x_s, and the constant is calculated in order to center the plot vertically [38].

$$\hat{f}(x_s) = \int_{z_{s0}}^{x_s} \mathbb{E}[\hat{f}_s(X_s, X_c)]|X_s = z_s)dz_s - constant \tag{14}$$

$$\hat{f}(x_s) = \int_{z_{s0}}^{x_s} \int_{x_c} \hat{f}_s(z_s, x_c)P(x_c|z_s)dx_c dz_s - constant \tag{15}$$

Therefore, $\hat{f}(x_s)$ can be interpreted as the accumulated local effects of x_s. Firstly, these calculate the local effect for $\hat{f}(x_s, x_c)$ of x_s at (z_s, x_c). Then, this effect is averaged across all values of x_c with weight $P_{c|s}(x_c|z_s)$, avoiding the use of marginal density $(P(x_c))$, similar to Marginal plots that use $P_{c|s}(x_c|x_s)$, and, consequently, extrapolation. Note that, opposed to directly averaging $\hat{f}$ as Marginal plots, by averaging across x_c and accumulating up to x_s the local effects, ALE avoid the omitted nuisance variable bias, the main problem of Marginal plots to assess the main effects of predictors. Lastly, the averaged local effect is accumulated/integrated over all values of z_s up to x_s [38].

Other examples of techniques used to analyze the effects a feature has on the target variable can be found in Residual plots [52–55], Trellis plots [56], Conditioning plots [38, 57, 58] and Conditional Response plots [38, 59]. The first are scatter plots of residuals, i.e., the difference between the observed value and the regression line, in comparison to the fitted value or the values of a predictor. By doing so, it is possible to analyze the relation between both, given that the plot does not take in consideration feature interactions, even though these may occur due to the existence of more variables in the model [52–55]. Trellis plots are useful to analyze specific scenarios, as these produce a sequence of plots, each showing the dependence of $\hat{f}$ on two variables. The values of $\hat{f}$ are plotted based on the values of the second variable, while conditioned by the first variable, i.e., using a subset of instances that are eligible based on the condition applied to this feature [56]. Conditioning plots are a variant of Trellis plots, plotting the relation between x_s and $\hat{f}(x_s, x_c)$ using dots, for a collection of discrete values of x_c [38, 57, 58]. Lastly, Conditional Response plots display the relation between x_s and $\mathbb{E}[\hat{f}(x_s, X_c)|X_c \in S_k]$, for each set S_k, with $k = \{1, ..., N\}$ and N being the number of instances, in some partition of the space of X_c [38, 59].

3.2.1.3 Feature Interaction

Additionally to exploring the effects that each predictor has on target variable, uncovering prominent interactions between features is a very important aspect of interpreting a model. Both PDPs and ALE plots present variants that allow the visualization of the relation between 2 predictors and the prediction, namely Second Order ALE plots (2D ALE) and Two Dimensional PDPs (2D PDPs) [36–40]. The latter account for the total effect of the two variables, as in Equation 16. Additionally, the interactions can be calculated by replacing x_s with a vector of two features, in Equations 7 and 8 [40].

$$PDP(x_i, x_j) = \hat{f}(x_i) + \hat{f}(x_j) + \hat{f}(x_i, x_j) \tag{16}$$

Contrarily, 2D ALE plots exclusively provide the visualization of the additional interaction between the features of interest, i.e., the second order effect, as seen in Equation 17. Moreover, this variant uses similar calculation method to evaluating only one predictor, differing by using k^2 rectangular cells, contrarily to k intervals in a grid. By doing so, the local effects are accumulated in two dimensions [38].

$$ALE(x_i, x_j) = \hat{f}(x_i, x_j) \tag{17}$$

Furthermore, both PDPs and ALE plots allow the visualization of higher order interactions. Nevertheless, the visualization of these may become confusing for more than 3 variables [38, 40].

Moreover, some methods based on some principles of 2D PDP were developed by Greenwell et al. [39] and Friedman et al. [41], in order to measure the feature interactions and obtain a score. The latter is denominated H-Statistic and is quite computationally expensive. However, it is capable to detect all types of interactions, generating a value between 0 and 1 for each pair of features, in which 0 indicates absence of any interactions and 1 informs that the effect on the output is entirely produced from the interaction [41]. Moreover, this metric is based on the assumption that, in the absence of interactions between the variables, the partial dependence can be decomposed as the sum of each partial dependence function of the variables, as in Equation 10 [41]. Then, H-Statistic measures the fraction of the variance of the two dimensional partial dependence function, not captured by the previous sum, as seen in Equation 18. Although efficient in identifying the interaction between variables, H-Statistic does not distinguish the regions in the domains that contain the strongest interactions [41].

$$H_{ij} = \frac{\sum_{k=1}^{n} [\hat{f}(x_{ik}, x_{jk}) - \hat{f}(x_{ik}) - \hat{f}(x_{jk})]^2}{\sum_{k=1}^{n} \hat{f}_{2D}(x_{ik}, x_{jk})} \tag{18}$$

Another method used to graphically identify interactions between all features, including interactions of 3 or more predictors is Variable Interaction Network (VIN), proposed by Hooker [42]. Nonetheless, this tool does not point out the specific scenarios or values for which the interactions occur, neither the magnitude of the interaction [42].

3.2.1.4 Global Surrogate

Surrogate models are important tools to explain and debug models. These are interpretable models and can give global or local information about predictions and errors. In order to do so, these emulate the original model, by constructing a relation between its inputs and predicted values. However, there are few guarantees the surrogate model accurately represents the more complex original model from which it was generated. As a consequence, it is necessary to evaluate the accuracy of the models, taking into consideration error metrics between the predictions and the response function being studied. Guaranteeing low and stable errors for the data to be explained may allow for better results. Additionally, these models can be combined with direct explanations, fairness and debugging techniques to increase the interpretability of the models [1]. For instance, some global surrogate models utilize decision trees to emulate the original models [43–45], others general additive models or decision rules [46], among others.

3.2.2 Local Methods

In order to study and evaluate a specific instance or a group of similar instances, the use of local methods is crucial [1]. Some of these are based on conditional situations, explaining how changes in the predictors alter the predicted value [60, 61]. Others study the importance of feature values in relation to the output, possibly decomposing the prediction [1, 46, 62–67]. Moreover, the study of interactions among features is an important technique, explaining hidden patterns and confirming or disproving known ones [67, 68]. Thus, in the following sections, multiple techniques among various types of local methods are addressed, from conditional situations, to counter factual explanations and local feature importance.

3.2.2.1 Ceteris Paribus Plots

Ceteris Paribus plots, or "what if" plots, are a useful model agnostic tool to evaluate the responses produced by a model around distinct values of a single feature. To do so, none of the remaining values of the feature is altered. Therefore, *Ceteris Paribus* plots present the response of a model as a function of a single variable [60, 61]. For instance, this tool can be used to explain possible ways to increase the credit score of a specific client, boosting the explainability of the model by generating multiple scenarios using the available features [60]. Moreover, this tool can be used to compare multiple models and to understand if a model is locally stable for a certain prediction [60].

3.2.2.2 Local Surrogate

Introduced by Ribeiro et al. [62], Local Interpretable Model-Agnostic Explanations (LIME) use surrogate models to explain regions around an observation of interest. Firstly, the original observation of interest is transformed into a simplified input space of binary vectors. Then, a new dataset of similar observations is created by sampling features that are present in the transformed instance and, for each observation, a measure of similarity is applied in order to associate weights to each example based on the proximity to the instance in study. Consequently, this dataset is used to train an interpretable model, locally accurate to the values predicted by the model [62]. The parameters of this model can be used to describe the average behavior of the response function around the observation of interest and to generate reason codes, i.e., plain text explanations of a prediction based on the values of the features [1]. Additionally, LIME has the advantage of generating simplified explanations using the most important local variables [62]. However, this method is unstable and sometimes inconsistent, as altering the size of the neighborhood or the sample can alter the explanations [69]. In order to solve the latter problem, Ribeiro et al. [63] proposed Anchors. These are a model agnostic tool that generate high precision sets of plain text rules, describing the prediction of a model based on the input values. Moreover, Anchors highlight the part of the input that is sufficient for the model to generate a prediction. Consequently, these are intuitive and easy to understand by users [63], enhancing trust in the model when the importance of the variables are in conformity with the domain knowledge of the users [1]. Additionally, Anchors are predominantly applied to classification problems and can produce results more precise than LIME [63].

Other important local tools based on surrogate models are Local Interpretable Visual Explanations (LIVE) and Local Rule-Based Explanations (LORE). LIVE consists of a modified implementation of LIME focused on regression tasks. Differing from the latter, the dataset for local exploration is generated by perturbing the explained instance, one feature at the time. Additionally, all generated observations are treated as similar to the observation of interest, i.e., all the neighbor points have the same weight and the original variables are used as interpretable inputs [64]. LORE starts by learning a local interpretable predictor, a decision tree, on a synthetic neighborhood, generated by a genetic algorithm, to produce a local explanation. Then, a meaningful explanation is derived from the predictor, consisting of a decision rule and a set of counterfactual rules. The first allows the explanation of the reasons that produced the decision and the latter suggests changes in the values of the features of the instance that may lead to different outcomes [46].

3.2.2.3 Local Feature Importance

Local feature importance refers to values that explain how much a certain feature contributed to the prediction [1]. Methods like LIME, Anchors and variants produce interesting results to analyze the feature importance of examples. Although these can be used on nearly every model, these are approximate methods [1, 62, 63]. Thus, the need for Shapley local variable importance, an exact method with theoretical guarantees from economics and game theory [70]. Nonetheless, this is a very time consuming technique [65]. As a consequence, this method is not always used, being replaced by techniques as LIME, Anchors, Leave-One-Covariate-Out (LOCO), among others [1]. Regarding LIME, this tool produces sparse, i.e., simplified, explanations, using only the most important features of the data [62]. Similarly, Anchors produce sparse explanations, but can be more specific. Moreover, these generate rules about the most important features, instead of numeric values [63]. That said, although LIME variants and LOCO generate explanations in real time, these will not be as accurate as Shapley explanations [1].

In regard to concrete methods, LOCO variable importance is a model agnostic technique that allows for local interpretations. It is based on leaving one feature out of the prediction, i.e., by setting the value of the feature in question to missing, zero, its average or other similar measure [66]. Hence, for each row of the dataset, the model predicts the output using the full row and then again for each predictor left out. Thus, the feature that produces the largest absolute impact on the prediction is labeled as the most important feature for that specific example. However, this method may produce worse results when complex nonlinear dependencies exist in the model. Additionally, LOCO can rank the features by their impact on the output as a per-row basis, creating global explanations [66].

When it comes to Shapley explanations, these have credible theoretical support and derive consistent local variable contributions to predictions of the model. In terms of the process, firstly, observations are transformed into a simplified form of binary values. It is important to know that the explanation models are restricted to additive feature attributions methods, i.e., the predicted values are linear combinations of binary input vectors. As seen in Equation 19, g represents the explanation model, x a binary vector of dimension N and ϕ_i consists of the weight of the i^{th} feature, where $i = \{1, ..., N\}$, measuring how that feature contributes to the prediction. Therefore, by finding the optimal weights, it is assured that the model has desirable properties of local accuracy and consistency and can be rank ordered to generate reason codes [65].

$$g(x) = \phi_0 + \sum_{i=1}^{N} \phi_i x_i \tag{19}$$

Moreover, Shapley explanations can be model agnostic or model specific, as these use a variant of LIME for model agnostic explanations and take advantage of tree structures for tree-based models [65]. Besides this, although a local method, Shapley explanations can be aggregated to create global explanations, enhancing understanding by explaining each observation of the dataset [1, 65].

3.2.2.4 Prediction Decomposition

Additionally to giving information about feature importance, Shapley explanations can be used to decompose the prediction, i.e., to separate the explanation of the prediction into smaller explanations based on the most important features. For each prediction, Shapley Additive Explanation (SHAP) values explain the difference between the average output of the model and the obtained value, assigning each feature a certain weight based on this difference [51, 65, 67]. This prediction is calculated for all possible combinations of features, considering and excluding the feature of interest, in order to determine its weight [65]. However, unrealistic combinations of features might appear, due to some level of correlation [67]. Moreover, as stated before, the calculation of SHAP values is time demanding. Nonetheless, there are some approximations. One example of this is *BreakDown*, presenting a fast approximation of SHAP values, based on model relaxations [64]. Additionally, proposed by Lundberg et al. [67], an optimized version for tree ensembles is available.

3.2.2.5 Feature Interaction

In order to take into consideration the possibility of correlated features in the data, Lundberg et al. [67] proposed SHAP interaction values, an extension of SHAP values based on Shapley interaction index from game theory [71]. These capture pairwise interaction effects, differing from previous methods that could not directly represent these, but divided the impact of an interaction among each feature. Therefore, SHAP interaction values guarantee consistency while explaining interaction effects for individual predictions and present a way to measure potentially hidden pairwise combinations in tree based models [67].

Concretely, the interaction values represent pairwise combinations, forming a matrix of values that represent the impact of all pairs of features on the prediction of the model. Each element of this matrix is calculated following Equations 20 and 21, where S represents the subset of input features, M is the number of input features and N is the set of all input features. It is important to notice that this equation is only applied when $i \neq j$ [67]. Moreover, the interaction value between features i and j is split equally, i.e., $\Phi_{i,j} = \Phi_{j,i}$ and the total interaction effect is given by $\Phi_{i,j} + \Phi_{j,i}$ [67].

$$\Phi_{i,j} = \sum_{S \subseteq N \setminus \{i,j\}} \frac{|S|!(M - |S| - 2)!}{2(M-1)!} \nabla_{i,j}(S) \tag{20}$$

$$\nabla_{i,j}(S) = f_x(S \cup \{i,j\}) - f_x(S \cup \{i\}) - f_x(S \cup \{j\}) + f_x(S) \tag{21}$$

The main effects for a prediction can be defined as the difference between the SHAP value (weight) and the SHAP interaction values for that feature, as in Equations 22 and 23. Therefore, SHAP interaction values allow the consideration of main and interaction effects for individual predictions, following similar axioms as SHAP values [67].

$$\Phi_{i,i} = \phi_i - \sum_{j \neq i} \Phi_{i,j} \tag{22}$$

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(M - |S| - 1)!}{M!} (f_x(S \cup \{i\}) - f_x(S)) \tag{23}$$

Typically, the impact of predictors in tree based models is accomplished by using a bar chart, representing the global feature importance, or using a PDP, describing the effect of altering a single feature [50]. However, as SHAP values produce results unique to every prediction, more informative visualization techniques can be used. Proposed by Lundberg et al. [67], SHAP summary plots replace standard feature importance bar charts, as these do not represent the range and distribution of impacts that the feature has on the prediction and how its values relate to its impact. Therefore, summary plots sort the feature by their global impact and display dots representing SHAP values in a violin-like plot for each feature [67]. Additionally, Lundberg et al. [67] introduced SHAP dependence plots, based on the ability of PDPs to represent the expected output of a model when the value of one or more predictors are fixed, as plotting how the output reacts to alterations in a feature helps in explaining how the model depends on that feature. While PDPs only produce lines, SHAP dependece plots display vertical dispersion due to interaction effects in the model, usually visible by each dot being colored with the value of the interacting predictor. Moreover, combining dependence plots with interaction values may reveal global interaction patterns [67]. Furthermore, Datta et al. [68] presented Quantitative Input Influence, a tool that measures the degree of influence of the predictors for solo or multiple predictions. However, this tool can only be used for classification tasks [68].

3.2.2.6 Counter Factual Explanations

There is a slight difference between counter factual explanations and adversarial examples. The latter can be defined as a sample of input data that was previously modified in order to alter the predicted result [72]. The first exemplifies small alterations of feature values in order to alter the output, while explaining it [73]. For instance, one counter factual explanation should indicate, alongside the predicted value, a subset of different feature values that would lead to a different prediction and its value [74]. Consequently, counter factual explanations can be useful to explain the reasons behind a certain outcome [75]. Moreover, one can use these to analyze the fairness of a system [76] and to identify bugs and errors in the models, since these are concise and easy to understand by humans [77]. However, as a consequence of their nature, by generating explanations, security and privacy leaks can occur, exposing instances of the training data to the users [74]. In terms of concrete tools, Krause et al. [78] proposed a visual tool that utilizes PDPs in order to produce data with different outputs, by altering the values of the predictors.

3.2.3 Regional Explanations

Proposed by Britton [16], regional explanations describe behaviors that affect significant regions of the data. By doing so, these are neither global, nor local methods, since they produce results more general than local methods and more specific than global techniques. Regional explanations can be obtained in one of two ways. Either an algorithm identifies a region of the data where many instances share a common behavior, being provided a succinct description of the data cluster, or the behavior itself is described [16].

Alongside the paradigm, Britton [16] also proposed a model agnostic tool to evaluate algorithms, designated Visual Interaction Effects (VINE). This tool utilizes modified ICE curves to produce detailed information about how feature interactions affect the predictions of the model. Additionally, in order to identify the interactions, one must read the produced chart, not being required any detailed statistical analysis. It is important to mention that the prediction function of the model has to be passed as an input of VINE [16].

In order to address the issue of overplotting associated with ICE curves, VINE clusters similar curves utilizing a measure of slope similarity and represents a centroid curve for each cluster instead of multiple ICE curves. Moreover, to explain what clustered curves have in common, i.e., what differentiates them from the remaining curves, a decision tree with depth of 1 is used to predict membership in the cluster against all other points is used [16]. This method allows the identification of the feature and the split value that reduces the entropy between the curves inside and outside the cluster the most. Hence, the split can be used as an explanation on what characteristics make the clusters unique [16]. Also, to avoid duplication or very similar cluster explanations, an algorithm to merge clusters is used, given that there is not an *a priori* indicator of the ideal number of clusters. Furthermore, in VINE charts, the horizontal represents the strength of feature interactions, calculated as the sum of Dynamic Time Warp distances between each VINE curve and the PDP curve, normalized by the maximum value of the latter curve, and the vertical axis indicates the overall feature importance [16].

VINE explanations detect real subsets of data, based on shape similarity and not simple fitting noise, and successfully measure feature interactions. Additionally, VINE curves present higher fidelity than PDPs and allow users to generate human readable characteristics for subsets. However, being based on ICE curves, these have fundamental limitations, consequence of extrapolation issues. Moreover, the tool produces better results when most features in the dataset are numerical. Nonetheless, it is not suitable to process large datasets, as these may use a large amount of memory and time to be processed[16].

3.3 Association Rule Learning

Association Rule (AR) Learning, introduced in 1993 by Agrawal et al. [79], represents a classical method to detect data patterns in the form of rules. In broad terms, given a set of transactions, where each transaction is a set of items, i.e., atomic elements of data, an AR is an expression $X \Rightarrow Y$, in which X and Y are sets of items. The idea is that transactions in the database that contain the items in X, also contain the items in Y [80]. For instance, a rule can be defined stating that 95% of customers that buy cookies and butter also buy milk at a store, where 95% is the confidence of the rule and the support would be the percentage of transactions in the database that contain both X and Y. Moreover, the confidence metric can be seen as the conditional probability, i.e., its predictive capability, and a measure of the strength of the rule. It is also important to distinguish the concepts of antecedent and consequent. In the example above, the antecedent consists of cookies and butter and the consequent of milk alone. Thus, the problem associated with AR is to find all rules that satisfy user-specified minimum values of support and confidence. Some examples of common applications of AR can be found in customer segmentation based on buying patterns, store layout, among others [80].

3.3.1 Classical Association Rules

Although AR can be divided into multiple variants, it is important to understand the basics. That said, as stated before, Classical Association Rules (CAR) were introduced by Agrawal et al. [79].

Assuming the example of a database related to a shop, let $Z = \{I_1, I_2, ..., I_N\}$ be a set of binary attributes. In this case, these sets are called items. Moreover, assume that T represents a database of transactions. Each transaction t can be represented as a binary vector, where $t[i] = 1$ means that the item was bought, and $t[i] = 0$ otherwise. It is important to note that, for each transaction, there is a tuple in the database. Let X be a set of items in Z. If, for all items I_i, $t[i] = 1$, the transaction t satisfies X [79].

In the original work, Agrawal et al. [79] define an AR as an implication of the form $X \Rightarrow I_j$, where X is a set of items and I_j is a single item of Z that is not present in X. The rule $X \Rightarrow I_j$ is satisfied in the set of transactions T if a factor C is a value between 0 and 1, included. This factor C can be defined as the percentage of transactions in T that satisfy X and I_j simultaneously.

Depending on the subject and purpose of application of CAR, some constraints can be utilized. These constraints can be divided into two categories: Syntactic and Support Constraints. Regarding the first, they are related to restrictions on items that can appear in a rule. For instance, only rules that have a specific item appearing in the consequent or antecedent may be considered important. It is also possible to use combinations of the above scenarios, where all rules must have items of a predefined itemset A in the consequent and items from another set B in the antecedent. When it comes to the latter, these are related to the number of transactions that support a specific rule, i.e., the fraction of transactions in T that satisfy the union of items in the consequent and antecedent of the rule. Contrarily to confidence, support corresponds to statistical confidence of the rule. Additionally, Support Constraints can also be important in situations where only rules above a specific threshold are to be accounted for (value usually known as *minsupport*), e.g., due to business reasons [79].

3.3.2 Quantitative Association Rules

Due to the fact that CAR are not sensitive to data types and, as a consequence, do not consider numerical attributes, the need for mining information in databases with quantitative attributes appeared. Initially proposed by Srikant and Agrawal [81], Quantitative Association Rules (QAR) can be generated by creating categorical events from quantitative data, i.e., each event can either be a categorical item or a numeric interval. Hence, one example of a possible rule is $\{sex = male, age \in [5, 10]\} \Rightarrow \{height \in [100cm, 150cm]\ (confidence\ 80\%)\}$. In order to produce these, an algorithm is used to approximately find all rules, based on a discretization technique of numeric attributes, alongside an interest filter to reduce redundant and similar rules [81]. This has proven to be a strong tool for mining quantitative data. However, the use of intervals generated from numeric values is limited, can be misleading and lead to loss of information [82]. Therefore, Aumann and Lindell [82] proposed a new approach to mine QAR, with the purpose of revealing interesting behavior of subsets of the general population, while not relying on discretization of numeric values. Consequently, comprehensive measures, such as mean and variance, are used to understand the distributions of the various numerical attributes. Even though mean and variance are the most common, different measures, like median, are allowed [82]. That said, for example, by observing the rule $\{sex = female\} \Rightarrow \{wage:\ mean\ 8\$\ per\ hour\ (overall\ mean\ wage = 9\$)\}$, one can immediately notice that a group of people earns less than the average wage in the database [82].

In terms of discovery of interesting rules, the scenarios to look for are the ones where the distribution of a specific subset differs from the population. In order to identify these situations, standard statistical methods to measure the significance of disparity between distributions are used [82]. According to Aumann and Lindell [82], one rule is defined as $subset \Rightarrow mean\ or\ variance\ for\ subset$, i.e., the Left Hand Side (LHS) of the rule describes the characteristics of the subset of the population and the Right Hand Side (RHS) the deviated behavior. It is also important to notice that, differing from the implementation suggested by Srikant and Agrawal [81], there is no exponential blowup in the number of rules and items [82]. Lastly, in formal terms, let $E = \{e_1, ..., e_n\}$ be a set of attributes of the database D. The expressions $E_q \subseteq E$ and $E_c \subseteq E$ represent the quantitative and categorical attributes, respectively. Moreover, C is the set of all possible categorical values. Each transaction in D can be defined as a tuple $t = (< e_1, v_1 >, ..., < e_n, v_n >)$ of attributes and values, where $e_i \in C$ if categorical and $e_i \in \mathbb{R}$ if numeric [82].

Furthermore, Webb [83] proposed an extension of QAR, Impact Rules, where the RHS of each rule contains statistics about the target for a specific subgroup, defined by the antecedent, similarly to Aumann and Lindell [82]. Contrarily to the latter, Impact Rules are based on the Optimized Pruning for Unordered Search (OPUS) [84] algorithm, which is not an itemset-based algorithm, and do not rely only on distribution based measures of interest, such as variance [83]. These have less computational costs, avoiding, in many cases, the requirement for constraints imposed on rules to be discovered, such as minimum cover. Moreover, the measures of interest used in this context have a wide range of practical applications, allowing the identification of subsets that contribute the most/least to the output, in opposition to determining groups that are different [83]. For instance, the sum metric can be useful in scenarios where the target measures the end objective, e.g., the total profit generated. In addition, the impact measure is useful when the target is an intermediate variable, e.g., income from a transaction. The latter reveals the total contribution of a group and, consequently, its influence on the general distribution, favoring large groups with individuals that contribute more than the average to the result. Therefore, a subgroup that has higher mean in comparison to the overall average might not be the subgroup that has the most impact, as it can be small [83]. Moreover, some alterations and variants have been proposed regarding these. Namely to discard insignificant rules [85], prune derivative rules considering ancestors and children rules [86] and to efficiently dispose of uninteresting rules in large and dense datasets, since applying statistical tests to identify significant rules requires considerable computation costs and access to data in order produce the necessary statistics [87].

3.3.3 Distribution Rules

Introduced in 2006 by Jorge et al. [88], Distribution Rules (DR) are a generalization of AR and are particularly appealing when the property of interest is numerical. Contrarily to CAR, DR do not need to pre-discretize the numerical attribute of interest. Additionally, when in comparison with QAR, these do not require a reduction of the set of values in the RHS to a summary given by mean, variance or other measure, keeping the whole set of values of the attribute. Consequently, there is no loss of information [88].

Formally, a DR follows the form $A \Rightarrow y = D_{y|A}$, where A is a set of items, y represents the target attribute, originally quantitative, but easily extended to categorical data, and $D_{y|A}$ is an empirical distribution of the values of y for the examples where A is observed, i.e., $D_{y|A}$ consists of a set of tuples, each composed by a particular value of y and the frequency of that value for the sample where A is present [88]. Nevertheless, it is important to notice that the attributes on the antecedent are either categorical or discretized into numerical intervals [88].

One simple example of a DR is as follows: let the target attribute be the salary of each individual in dollars, $\{man, young\} \Rightarrow \{500/2, 504/4, 670/3, 811/1\}$ states that, among young men, 2 receive a value of 500, 4 of 504, 3 of 670 and 1 of 811. That said, the process of discovery of distribution rules consists of finding all rules where the LHS has a support above the minimum defined and the RHS is statistically different from the default distribution $D_{y|\emptyset}$, based on a pre-defined threshold. This default distribution is obtained using the complete dataset, i.e., it is an empirical distribution of the attribute in study, using every record in the dataset, or a predefined reference distribution [88]. In order to measure the interest of the discovered rules, DR calculate the difference between the distribution in the RHS of the rule and a reference distribution, generally, $D_{y|\emptyset}$. This difference is calculated through a statistical goodness of fit test, such as the Kolmogorov-Smirnov (KS) test, even though other statistical tests can be used, like Cramér-von Mises tests. In regard to the KS test, the interest of a specific rule is calculated by $1 - p$, where p is the *p-value* obtained with the test [88]. In addition, a threshold is used in order to determine which rules are to be accounted for, based on the results of their goodness-of-fit tests [88].

In order to better understand the concept, Figure 7 contains a visual representation of one DR for dataset Auto (cf. Table 1). Here, one can see that many cars with 6 cylinders and originated from the United States present lower mileage per gallon when in comparison to the general population, specifically cars around 20 miles per gallon. Moreover, the remaining instances present lower millage per gallon than the general population, as seen in the right tail of the distribution.

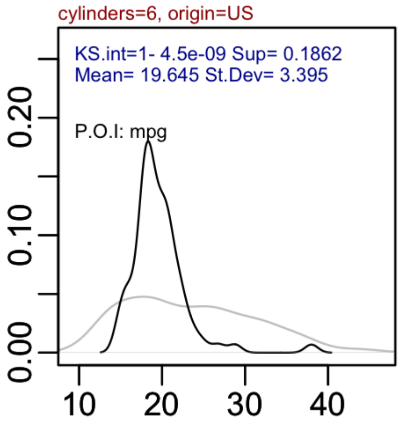


Figure 7: Graphical representation of one DR from dataset Auto (cf. Table 1), where the gray line represents the distribution of millage per gallon of the whole population of the dataset and the black line concerns the cars with 6 cylinders and originated from the United States.

Similarly to CAR and QAR, DR can be easily used in the context of subgroup discovery, regardless of the type of the property of interest, prediction, classification, clustering, among other scenarios [88].

3.4 Summary

The evaluation of black box regression models can be achieved through multiple approaches, each producing specific information. On the one hand, an algorithm can be analyzed regarding its performance. To that end, it is necessary to analyze the errors of the model, i.e., the differences between the real target values and the predicted ones. In order to do so, one can rely on scalar measures, such as the MSE or the MAE, or graphical metrics, e.g., EDPs, REC curves and surfaces, among others. Moreover, this approach is widely used to compare models based on certain requirements and useful to assess possible risks associated with the use of some models. On the other hand, interpreting a model, namely, studying the relation between the predictors and the ground truth, can unveil important scenarios, while explaining the predicted value. This can be divided into two major groups, global methods, evaluating the model as a whole, and local methods, that study one instance or a small group of similar instances. Moreover, a new approach has been proposed to define the study of similar instances as regional explanations with VINE. Some approaches generally used among global methods weigh the importance of the features regarding the output, like PFI and MCR, others analyze the effects the values of the predictors have on the target, such as PDPs, ICE plots and ALE plots. These effects may include possible interactions of values, studied with multivariate variants of PDPs and ALE plots, H-Statistic, among other tools. Moreover, some tools emulate the original model using an explainable one, based on its inputs and outputs. Regarding local methods, similarly to global methods, the analysis of feature importance can be approached, using, for instance, LIME, LOCO or Shapley explanations. Other possible technique is *Ceteris Paribus* plots, i.e., the alteration of the value of one feature, allowing the study of alterations on the predicted value. Additionally, the use of surrogate models to explain regions around an instance of interest and highlight the values that are sufficient to make a prediction using tools like LIME, Anchors, LORE or LIVE. Lastly, decomposing the prediction, specifically separating the explanation of the prediction into smaller explanations, based on the most important features, e.g., SHAP or *BreakDown*. Moreover, one can assess the interactions of features for a certain instance, using SHAP interaction values and SHAP dependence plots, or produce counter factual explanations, by studying the consequences of making modifications in the feature values in order to generate different outputs.

Furthermore, the use of AR can be an important tool to uncover subgroups of data that share some similarities, combined with some other methods to interpret the models. Hence, knowing that the classic definition of AR can only be used for categorical data, an alternative to study datasets with a numerical property of interest was developed, QAR. The first approach regarding the latter created categorical instances from quantitative records, possibly losing some information. Consequently, a new approach was defined, not needing discretization of the feature of interest and using interest measures such as mean and variance. However, sometimes the impact a certain subgroup has on the result is more important than uncovering all groups that are different, leading to another variant of QAR, Impact Rules. These rely on extra measures of importance, quantifying the weight each group has on the outcome. Lastly, as a reduction of the set of values on the RHS to a summary of measures might lead to some loss of information, DRs extend QAR by containing on the consequent an empirical distribution based on the discrete characteristics of the subset that is statistical different from a reference distribution, generally the whole population.

Chapter 4

Implementation Details and Results

In the current chapter, the datasets used for benchmarking are presented and described, as well as the different machine learning models applied to the data. Also, the discretization process and the manner to calculate error values are characterized. Afterwards, the developed tools are described in detail, namely boxplots, histograms and density plots of uncovered subgroups. Moreover, a network visualization is also presented, as well as performance tables and an extrapolation of counter factual explanations to regression. The difference between certain error types is described, as well. Finally, some interesting examples of certain subgroups are displayed and analyzed for multiple datasets.

4.1 Datasets

In order to enable the test of the developed tools, a significant number of datasets was used. These are described in Table 1. Some datasets are only composed by a relatively small number of instances, e.g., *A1* or *A7*, while others are significantly larger, as *CpuSm*. Moreover, one can see that the number of predictors varies from 4 to 37, allowing the study of a multitude of possibilities, simulating realistic problems. Representing multiple combinations between the number of numerical and categorical features is also important, as a dataset can be composed solely by one type of attributes, or by a combination of both. To allow full reproducibility of the results, the original and transformed datasets are publicly available at `https://github.com/citoplasme/MScDissertation/tree/1.1/code/validation/data`. The link contains three versions of each dataset. In the first one, the datasets are as is, with no major alterations. The second version is composed by the original datasets and four new columns per dataset, representing the predicted values of every model seen in Table 2. Lastly, the third version is fully prepared for subgroup discovery, with discretization of numerical attributes, as described in detail in Section 4.3.

4.2 Models

Each dataset mentioned in Table 1 was used as part of a regression task using the predictive learning algorithms in Table 2. Four different algorithms were used, including an Artificial Neural Network (ANN), a

Table 1: Datasets used for benchmarking.

Dataset	Number of instances	Number of predictors	Numerical predictors	Categorical predictors
A1	198	11	8	3
A2	198	11	8	3
A3	198	11	8	3
A4	198	11	8	3
A6	198	11	8	3
A7	198	11	8	3
Abalone	4177	9	7	2
Acceleration	1732	14	11	3
Airfoild	1503	5	5	0
Auto	392	8	7	1
AvailPwr	1802	15	8	7
Bank8FM	4499	8	8	0
Boston	506	13	13	0
ConcreteStrength	1030	8	8	0
CpuSm	8192	12	12	0
FuelCons	1764	37	25	12
MachineCpu	209	6	6	0
MaxTorque	1802	32	19	13
Servo	167	4	2	2

Gradient Boosting Machine (GBM), a Random Forest (RF) and a Support Vector Machine (SVM), in order to avoid any model dependency bias within the experiments. Moreover, similarly to the datasets, in order to allow full reproducibility of the experiments, we used the R programming language [89] and open source implementations of the black-box models.

Table 2: Regression algorithms and respective parameters used for benchmarking.

Model	Parameters	Package
Artificial Neural Network	size = 10, maxit = 1000, decay = 0.1, na.action = na.omit, linout = TRUE	nnet [90]
Gradient Boosting Machine	distribution = ”gaussian”, n.trees = 10000, interaction.depth = 1, shrinkage = 0.001, cv.folds = 5, n.cores = NULL, verbose = FALSE	gbm [91]
Random Forest	ntree = 500	randomForest [92]
Support Vector Machine	kernel = ”radial”, cost = 1, epsilon = 0.1, gamma = 1/(data dimension)	e1071 [93]

In addition, no hyper-parameter tuning was performed, as the goal is solely to avoid any model bias in the experiments.

4.3 Data Discretization

In order to find interesting subgroups using DRs, the numerical features have to be discretized previous to the discovery process. Ideally, this action should be performed with user or domain specific requirements, i.e., some level of knowledge on the ideal division of the values. However, as a large number of different datasets is being studied, we opted for the use of quartiles of the data, similar to the default approach proposed by Areosa and Torgo [28] regarding EDPs and explained in Section 3.1. Additionally, by utilizing the same approach, we are able to directly compare the results produced by our proposed tools and EDPs.

Furthermore, as our approach uses CAREN [88, 94–96] to generate DRs, one may consider using the different methods provided by this tool to deal with numeric attribute, such as Fayyad-Irani algorithm [97], or Srikant discretization [81]. Nevertheless, CAREN does not output the transformed dataset, only the rules and their measures. Consequently, it would not be possible to directly compare the results with other tools that need the discretization of numeric attributes, such as EDPs.

4.4 Error Calculation

Another crucial step during performance analysis is to guarantee that the expected prediction error values provide an estimate of the risk associated with using a certain model. Therefore, based on these values, and their correspondent feature values, the end user can assess the risk of usage. That is, whether the error values are within some tolerance limits and, consequently, the suitability of the model for the task. As the estimation of the error directly affects the quality of the results, the usage of a trustworthy method is imperative. Similarly to Areosa and Torgo [28], we followed the same Cross Validation (CV) mechanism to obtain error estimates for every available instance. Specifically, a 10 fold CV was used to calculate the prediction of the model for each example, i.e., the full dataset is divided into 10 different groups that serve as test sets one at a time. For each hold out set, the model is trained using the remaining data and evaluated using the first as a test set. Ergo, every data instance is part of only one test set. Finally, by comparing these predictions with the real values, reliable error estimates are attained.

4.5 Error Distribution Rules

In order to address some problems associated with EDPs, as the production of uninteresting plots for the end user to analyze, we propose Error Distribution Rules (EDR). These can be defined as an extension of EDPs, that use DRs to select interesting subgroups. This combination allows the production of lesser plots, as only those containing the distributions of combinations of characteristics that differ significantly from a reference distribution are generated. In fact, if we wanted to analyze every interaction that contains up to three variables, we would have to produce $N + \frac{N!}{2! \times (N-2)!} + \frac{N!}{3! \times (N-3)!}$ EDPs, where N represents the number of features in the dataset. Moreover, as DRs do not have any limitation regarding the number of features in a certain combination, this approach can unveil interactions previously unattainable by EDPs, since these are limited to a maximum of three variables. Lastly, in terms of software used to produce the plots, EDRs rely on the R package *ggplot2* [98].

Sections 4.5.1, 4.5.2 and 4.5.3 are composed of examples of the developed graphical methods. Thus, in order to allow some level of comparison between the techniques, the examples used are shared among them. The rules used in these examples were obtained from dataset *A6* (cf. Table 1), trained with a GBM model (cf. Table 2) and using logarithmic errors for better visualization. The discovery process had a minimum support of 5%. Regarding the single rule examples, these portray the comparison between DR $NO3 = [0.7 - 1.59], mxPH = [7.75 - 8.22], oPO4 = [7.4 - 31.33]$ with the global distribution, as this is our reference distribution. Similarly, the multi-rule examples compare subgroups partially or fully characterized by $NH4 = [82.1 - 176.67]$ between themselves and the reference distribution.

4.5.1 Boxplot

The first visualization method implemented uses the exact same graphical tool as EDPs to represent distributions, boxplots. These compactly depict the data by displaying summary statistics as the median, the first (Q1) and third (Q3) quartiles, located at 25% and 75%, respectively, the whiskers and some outliers. Concerning the whiskers, the lower one is calculated as $Q1 - 1.5 \times (Q3–Q1)$ and the upper as $Q3 + 1.5 \times (Q3–Q1)$ [99]. For convenience, minimum will be representative of the first whisker and maximum of the second throughout the remaining of the document.

To produce them from an EDR, it is only necessary to convert its distribution to a vector, as the RHS of each DR contains an empirical distribution. To do so, every value is replicated by the frequency it occurs in the distribution. Moreover, the *X* axis is composed by the description of the subgroups and the *Y* axis by the error values. A simple example can be seen in Figure 8, where, by analyzing the plot, one can perceive that the subgroup presents smaller error values in comparison to the global values, i.e, presents a better performance.

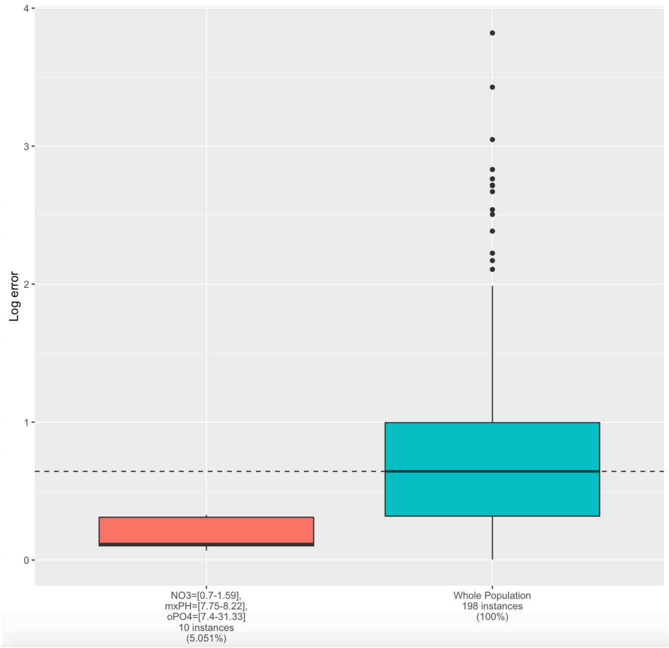


Figure 8: Example of a boxplot visualization of a EDR from dataset A6 (cf. Table 1), trained with a GBM model (cf. Table 2).

Another possibility is to compare multiple subgroups simultaneously, as depicted in Figure 9. Here, it is clear that every subgroup that is in part characterized by $NH4 = [82.1 - 176.67]$ has a tendency to produce smaller errors than globally expected.

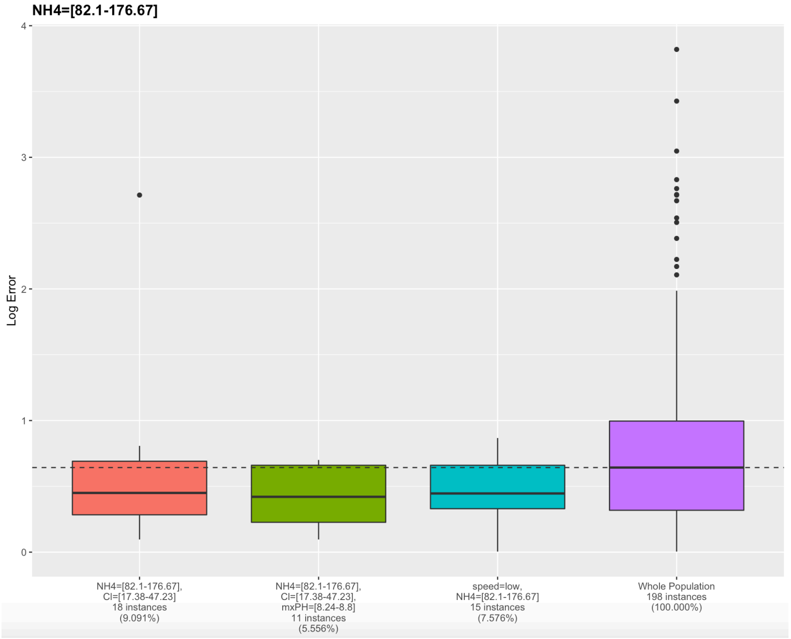


Figure 9: Example of a boxplot visualization of multiple EDRs from dataset A6 (cf. Table 1), trained with a GBM model (cf. Table 2).

4.5.2 Histogram

Even though boxplots produce extremely pleasant results, sometimes an extra level of detail about the distributions is necessary. That said, the second graphical option is to plot histograms of the DRs. In order to do so, the error values are divided into bins and the number of data instances in each bin is counted, allowing for the calculation of a relative frequency, displayed in the *Y* axis. This frequency is helpful for the visualization, as the use of an absolute count would lead to the subgroups being covered by the global distribution. Similarly to boxplots, histograms are generated using the RHS values contained in each rule. However, the error values are displayed on the *X* axis, instead of the *Y* axis.

The same example as in Figure 8 is now presented using an histogram in Figure 10. It is clear that the subgroup contains a higher density of smaller values of error, namely between 0 and 0.5, differing from the whole data, which covers a larger amount of possible values. By comparing the histogram with the corresponding boxplot, one can not only confirm the better performance, but also have a better understanding on how these values are spread.

Once more, it is possible to produce a histogram containing multiple distributions at once, as seen in Figure 11. Nonetheless, with the increase in the number of factors to plot, the visualization becomes more difficult, as we can see for error values between 0 and 1.

4.5.3 Density Plot

In order to improve over the histogram visualization of the rules, we decided to use density plots, a smoothed and continuous version of the first. For instance, by looking at Figure 12 we can see the two major error values where the instances of the subgroup are condensed.

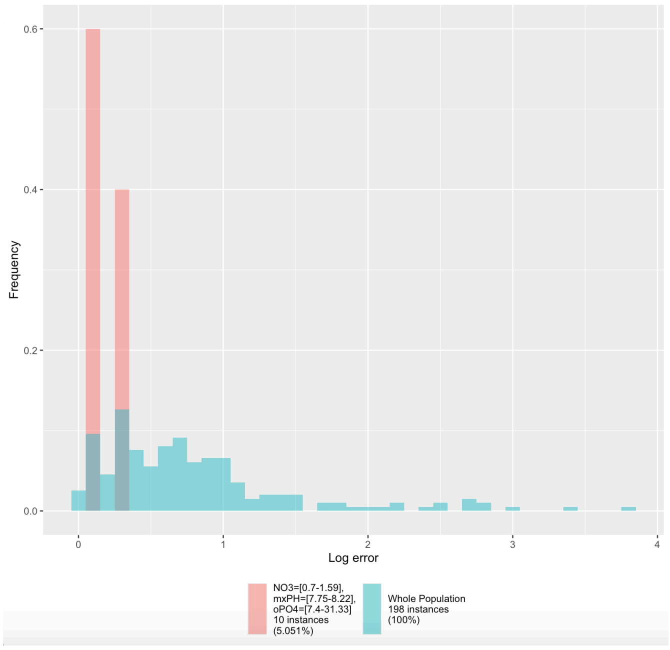


Figure 10: Example of an histogram visualization of a EDR from dataset A6 (cf. Table 1), trained with a GBM model (cf. Table 2).

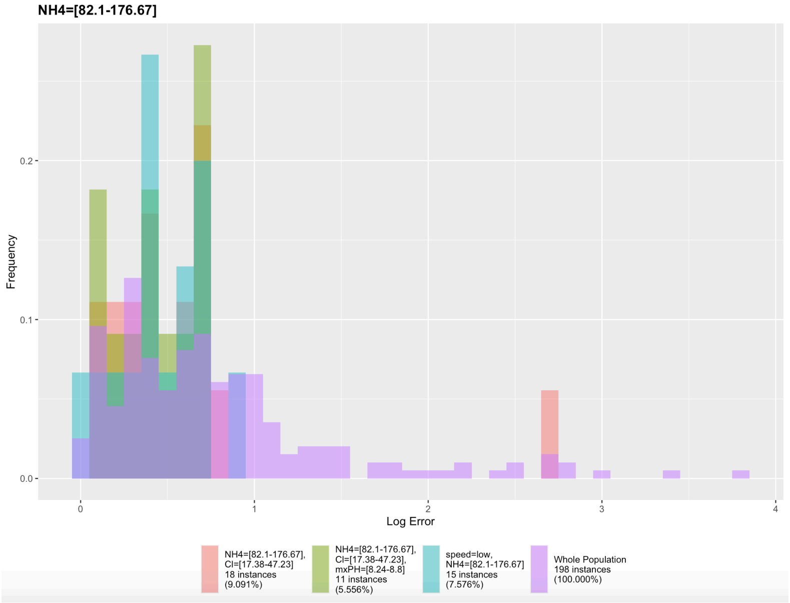


Figure 11: Example of an histogram visualization of multiple EDRs from dataset A6 (cf. Table 1), trained with a GBM model (cf. Table 2).

Regarding Figure 13 and comparing the plot with Figure 11, it is clear that this approach produces simpler plots to analyze. Although the error values are still grouped between 0 and 1, each distribution is now visible and easily compared. For example, in the histogram, the subgroup defined by $NH4 = [82.1 - 176.67], Cl = [17.38 - 47.23]$ is not very visible, except for outlier around 2.7. However, in the density plot it is easily perceived, being relatively similar to subgroup $NH4 = [82.1 - 176.67], Cl = [17.38 - 47.23], mxPH = [8.24 - 8.8]$.

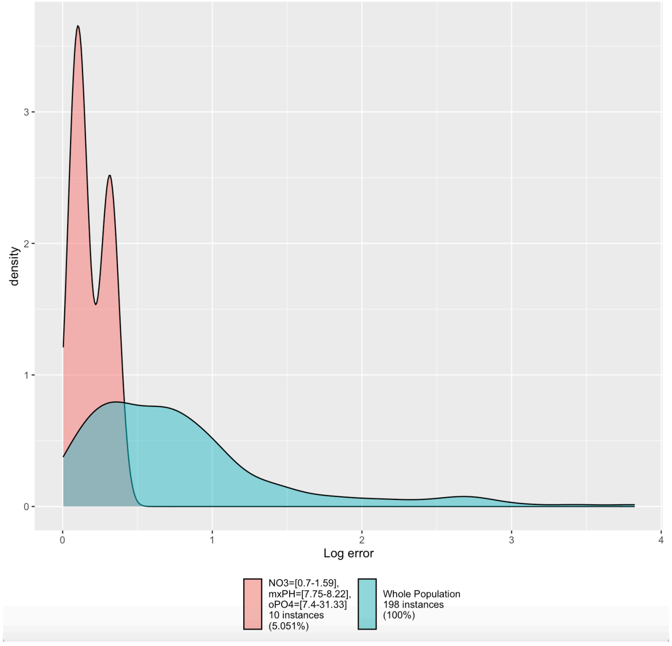


Figure 12: Example of a density plot visualization of a EDR from dataset A6 (cf. Table 1), trained with a GBM model (cf. Table 2).

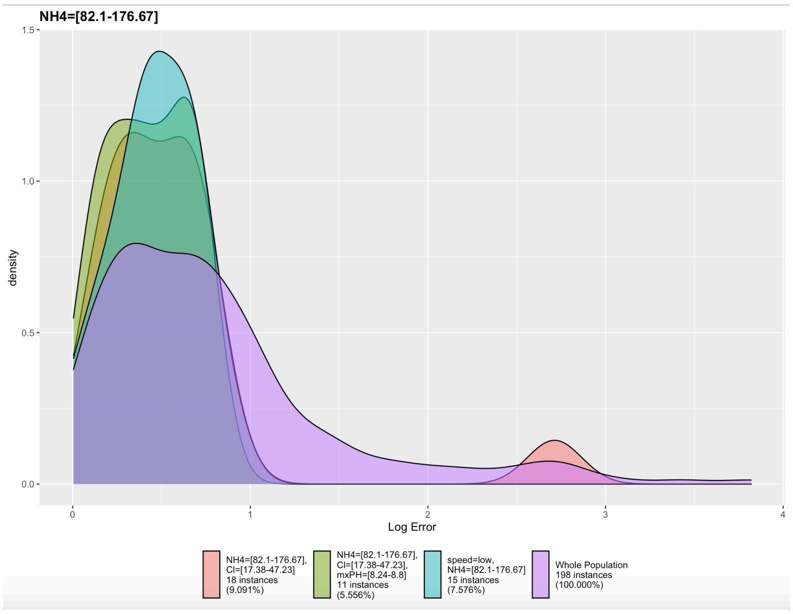
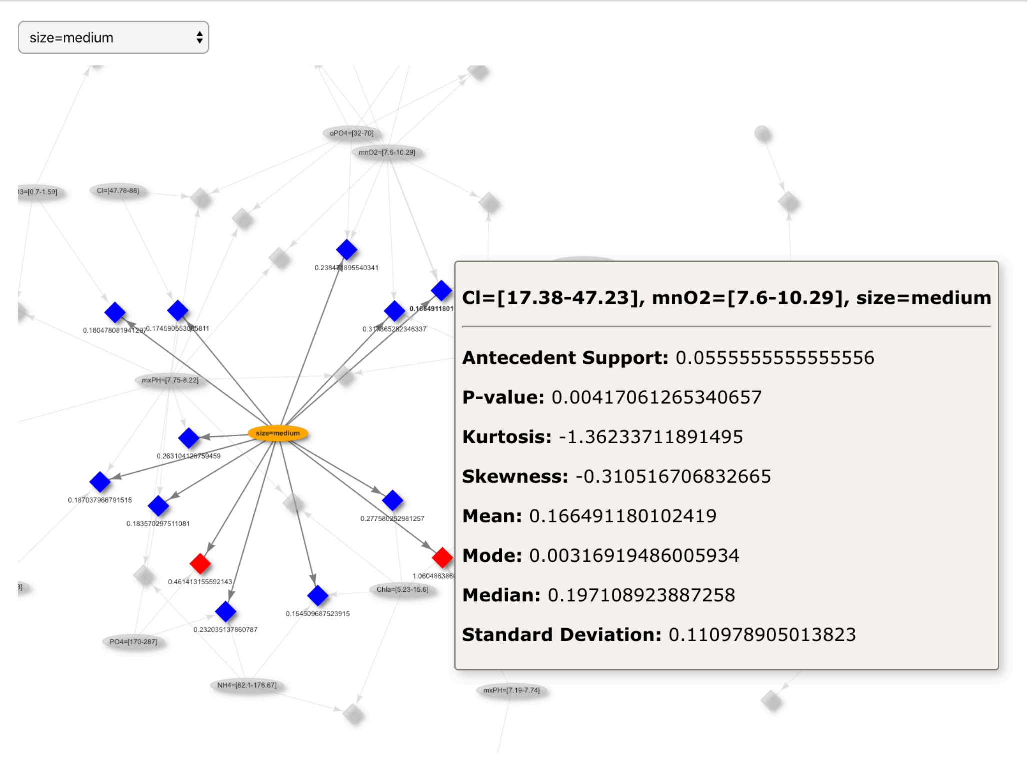


Figure 13: Example of a density plot visualization of multiple EDRs from dataset A6 (cf. Table 1), trained with a GBM model (cf. Table 2).

4.5.4 Network Visualization

As the number of subgroups discovered in a dataset can be extremely high, a simple and fast method to analyze the derived results is convenient. Consequently, we developed a method that produces an interactive network, using the *R* package *visNetwork* [100]. The network produced contains elliptic nodes, corresponding to bins of features, and diamond shaped nodes, that represent the DRs. Note that these can either be colored blue, if the median of the subgroup distribution is equal or smaller than the global, or red if higher, providing a simplistic, yet rapid way to assess the performance of a subgroup. A simple example of this can be seen in Figure 14, produced from dataset *A7*, using a SVM model. Moreover, the discovery parameters used were 5% for minimal support and 1×10^{-14} for minimal improvement filter. Another noteworthy aspect of this visualization method is the possibility to highlight a certain node and its direct connections. By doing so, or by hovering a node, its characteristics are displayed, i.e., its designation and, in the case of diamond nodes, values such as mean, model, or standard deviation.


size=medium

Cl=[17.38-47.23], mnO2=[7.6-10.29], size=medium

Antecedent Support: 0.0555555555555556

P-value: 0.00417061265340657

Kurtosis: -1.36233711891495

Skewness: -0.310516706832665

Mean: 0.166491180102419

Mode: 0.00316919486005934

Median: 0.197108923887258

Standard Deviation: 0.110978905013823

Figure 14: Network visualization of DRs from dataset A7 (cf. Table 1), trained with a SVM model (cf. Table 2).

4.5.5 Performance Tables

As seen before, the visualization methods have some limitations in terms of the number of distributions to compare at a given time. Consequently, another method was implemented, allowing the comparison of some key points of multiple subgroups, in the form of performance tables. These compare multiple subgroups with a reference distribution, selected as an input parameter by the end user.

Concerning the algorithm to create the tables, the first step is to generate the key points, i.e., minimum, Q1, median, Q3 and maximum, for the reference distribution and for each subgroup. Note that, contrarily to the boxplots, minimum and maximum values represent the actual lowest and highest values of the distribution, and not the lower and upper whiskers. Then, the resulting dataset of subgroups is melted, originating a dataset with three columns: subgroup, variable and value. The variable column is composed by the key point indication, which allows the correct comparisons. After this step, we loop over every row and generate categorical values for them, indicating how it behaves regarding the reference distribution. The possible values are Higher, Equal or Lower, considering the difference between the value of the subgroup and the reference. Lastly, using the transformed dataset, a plot is produced, being similar in some ways to a heatmap, as seen in Figure 15. This example compares every EDR discovered with the complete error distribution, allowing the analysis of how each subgroups behaves in comparison to the whole data. For instance, one can see that $season = spring$ contains both the minimum and maximum error values of the whole data. Moreover, $PO4 = [292.62 - 771.6]$ and $oPO4 = [205.64 - 564.6]$ are also defined by the same maximum value as the whole data.

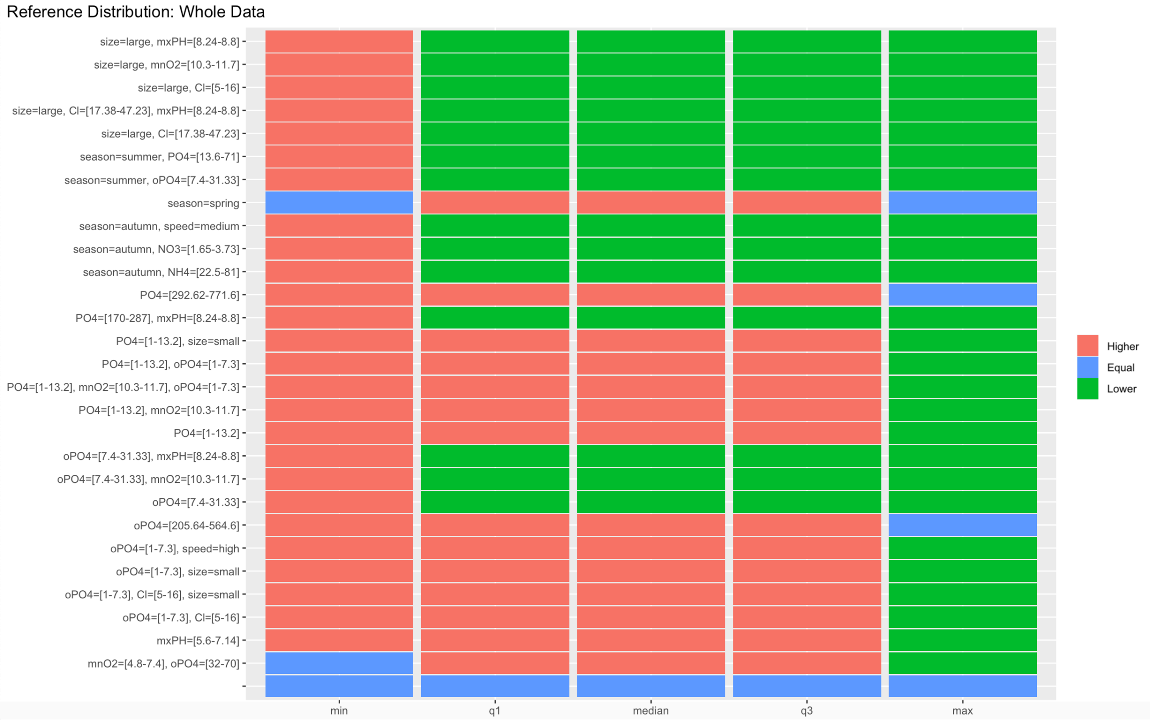


Figure 15: Performance table of DRs from dataset A4 (cf. Table 1), trained with a GBM model (cf. Table 2).

However, sometimes it might be helpful to compare only a certain parcel of subgroups with a very specific distribution. That said, Figure 16 depicts the comparison of every subgroup that extends $PO4 = [1 - 13.2]$ with $PO4 = [1 - 13.2]$ itself, for dataset *A4* (cf. Table 1). This visualization allows users to see how the interaction of $PO4 = [1 - 13.2]$ with other feature values affects the performance of the model. For instance, by grouping $PO4 = [1 - 13.2]$ with $size = small$ or $oPO4 = [1 - 7.3]$, the overall performance decreases, as seen by the Q1, median and Q3 values. Mind that, in the worst case scenario, the minimum and maximum values of a variations of $PO4 = [1 - 13.2]$ are the ones of the subgroup by itself. Moreover, it is also interesting to see that $PO4 = [1-13.2], mnO2 = [10.3-11.7]$ results in higher values for minimum, Q1 and Q3, but lower for both maximum and median, indicating an intriguing performance.

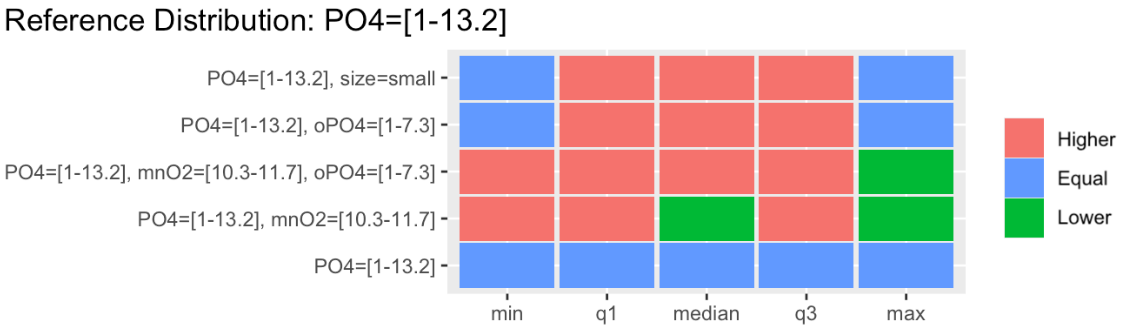


Figure 16: Performance table of a filtered group of DRs based on feature *PO4* from dataset A4 (cf. Table 1), trained with a GBM model (cf. Table 2).

4.5.6 Counter-Factual Analysis

Another approach that extends the idea behind performance tables and helps users, by focusing on subgroups with the most interesting performance values, is the study of counter-factual situations. These reduce the number of cases for the users to analyze, focusing only on the ones that differ in some key aspect of the distribution. The process starts by generating a dataset for each subgroup discovered, composed by other subgroups that share at least one data bin, i.e., a condition, with the first subgroup. In a way, the data structure can be seen as a hash table, i.e., a list of pairs, each composed by a key and a value. Each key represents a subgroup and each value a dataset of subgroups. These datasets are composed by the various values of the quartiles (minimum, Q1, median, Q3, maximum) and are ordered by the number of conditions shared with the subgroup (key) associated to them, in descending order. By doing so, the subgroups that have the most similarities are closer to the top of the dataset. Then, every instance is compared to a reference distribution, passed as input. Usually, this reference is represented by the whole data. Moreover, not only the values of each dataset are compared, but the subgroups that can be seen as their keys as well. Similarly to performance tables, three possible values are calculated: Higher, Equal or Lower. The major difference to the foretold occurs here, as the values are filtered if the quartile values associated with these are equal to the quartiles of their key. For example, in Listing 1, we see that rules 2 and 3 are associated with rule 1. Due to the fact that the performance of rule 2 is exactly the same as its key (rule 1), it is dropped, as it is only interesting to analyze subgroup 3 in the context of possible counter-factual scenarios. Moreover, as the analysis depends on the interests of the users, by providing these with the various cutting points, these are able to analyze the scenarios that are more interesting based on their preferences, without losing the notion of how the distribution behaves.

Listing 1: Example of a subgroup and its dataset of similar subgroups, before filtering.

```
(1) A = [1, 2], B = [3, 4]: Higher Higher Higher Higher Equal

     (2) A = [1, 2], B = [3, 4], C = class1
        Higher Higher Higher Higher Equal

     (3) A = [1, 2], D = [5, 7]
```

```
        Lower Higher Lower Lower Equal
```

Listing 2 depicts an example of a final counter-factual for dataset *A4* (cf. Table 1), concerning the subgroup $size = large, Cl = [5 - 16]$ in relation to the whole data. As seen, this combination is characterized for having higher error values for the minimum point and lower for the remaining points. Moreover, after filtering non-interesting subgroups that share one or more feature conditions with the first, only $oPO4 = [1 - 7.3], Cl = [5 - 16], size = small$ and $oPO4 = [1 - 7.3], Cl = [5 - 16]$ are selected. Both groups are defined by the same behavior in comparison to the whole data, i.e., lower values for the maximum errors and higher for every other point. Consequently, their behavior differs from $size = large, Cl = [5-16]$, making them counter-factual scenarios that lead to a differing performance regarding the expected error magnitudes.

Listing 2: Example of a subgroup and its dataset of similar subgroups, after filtering, from dataset A4 (cf. Table 1), trained with a GBM model (cf. Table 2)

```
>> size=large , Cl=[5−16]
>> Higher Lower Lower Lower Lower

        > oPO4=[1 −7.3], Cl=[5−16], size=small
        Higher Higher Higher Higher Lower

        > oPO4=[1 −7.3], Cl=[5−16]
        Higher Higher Higher Higher Lower
```

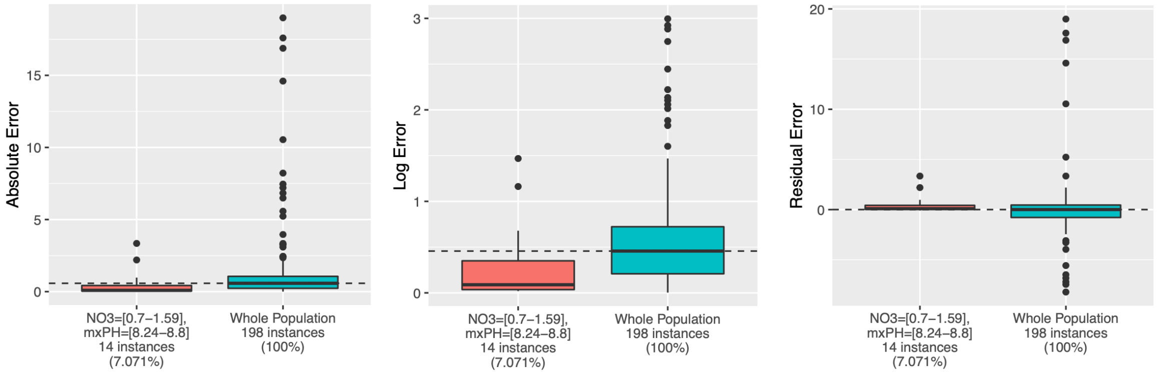
Obviously, the discovery of counter-factual examples can be achieved by analyzing the performance tables. Nonetheless, presenting the examples in a textual format yields an easier and faster way to highlight the relevant characteristics of the predictive model.

4.5.7 Effect of Error Measure

Another important aspect to mention is the impact of the measures of error in the results. Based on the performed experiments, absolute and logarithmic errors always produce the same DRs, differing only in metrics such as mean, median, mode, among others, as expected. Despite this, the behaviors of the subgroups compared to the overall data remain unchanged. Contrarily, the use of residual errors leads to the discovery of different DRs, as the distributions follow different forms, due to the existence of negative and positive values. This leads to situations where a subgroup that presents better performance than global, using absolute or logarithmic errors, is identified as having a distinct behavior or is not identified at all, due to not differing significantly from the whole data.

In order to exemplify the results described above, Figure 17 presents the same subgroup, $NO3 = [0.7 - 1.59], mxPH = [8.24 - 8.8]$, from dataset *A7* (cf. Table 1), trained with a GBM model (cf. Table 2), using absolute (Figure 17a), logarithmic (Figure 17b) and residual (Figure 17c) errors. Additionally, these rules were uncovered using a minimum support of 5% and 1×10^{-14} for minimal improvement filter.

By analyzing both the absolute and logarithmic version of the subgroup, it is clear that for both cases, Q3 is smaller than the global median and that the their median is close to Q1. In short, these plots show a subgroup with better performance than expected, based on the complete data. However, by observing the residual version, even though the median of the subgroup is still close to its Q1, these are slightly higher than the global median. Hence, by using the same comparison method as before, the subgroup would show worse performance than expected. Nonetheless, residual errors cannot be compared in the same way, due to existence of negative values. Thus, these can be used to study the tendency of a model to over or under predict.



(a) Absolute errors.

(b) Logarithmic errors.

(c) Residual errors.

Figure 17: Multiple versions of a subgroup from dataset A7 (cf. Table 1) to analyze the impact of the type of error, trained with a GBM model (cf. Table 2).

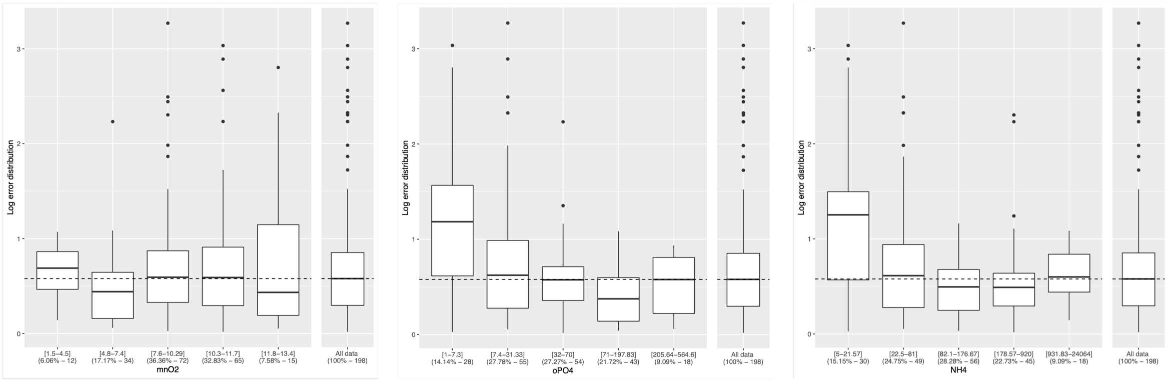
Another important aspect from the comparison between these three variants is the higher visibility granted from the logarithmic errors. Ergo, we opted to use these to present some examples.

4.5.8 Illustrative Examples

Often, when analyzing large datasets, uncovering interactions or frequent combinations of feature values is important to understand and predict the behavior of the models. Consequently, the usage of rules allows users to uncover these interactions easily, specifically the ones that act differently from the whole data. Therefore, in order to prove that EDRs produce interesting results and extend the analysis provided by EDPs, a few examples are considered in the following sections.

4.5.8.1 A1

The following example was obtained by training dataset *A1* with a GBM model and 5% as the minimum value of support. By analyzing the EDPs present in Figure 18, one can infer some behavioral aspects about the performance of the model. For instance, in Figure 18a, the data bin characterized by $mnO2 = [7.6-10.29]$ appears to have a distribution similar to the whole data. Moreover, in Figure 18b it is notorious that $oPO4 = [71 - 197.83]$ presents a better performance than global and $oPO4 = [7.4 - 31.33]$ is identical to general. It is also important to notice that, in Figure 18c, the bin $NH4 = [22.5 - 81]$ also presents a similar error distribution when in comparison to global.



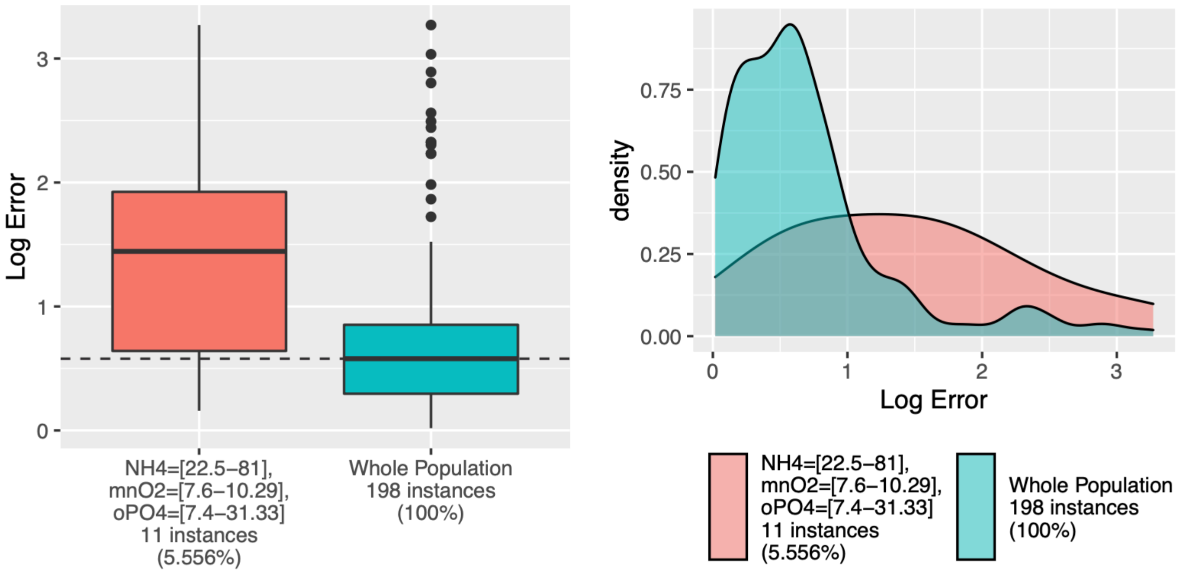
(a) Feature *mnO2*.

(b) Feature *oPO4*.

(c) Feature *NH4*.

Figure 18: EDPs from dataset A1 (cf. Table 1) to analyze features *mnO2*, *oPO4* and *NH4*, trained with a GBM model (cf. Table 2).

However, with the help of the EDR presented on Figure 19, we can see that, for example, the subgroup $NH4 = [22.5–81], mnO2 = [7.6–10.29], oPO4 = [7.4–31.33]$ is characterized for having higher errors than the complete dataset. By analyzing the boxplot in Figure 19a, it is clear that the error distribution of the subgroup has a higher value for *Q1*, median and *Q3* when comparing with the whole dataset. Besides this, the *Q1* value is even higher than the median for the whole data, displaying a significantly worse performance. Moreover, by viewing the density plot in Figure 19b, it is clear that, contrarily to the whole population of data, where the majority of error values are between 0 and 1, these are widely spread for the subgroup, culminating in worse performance.

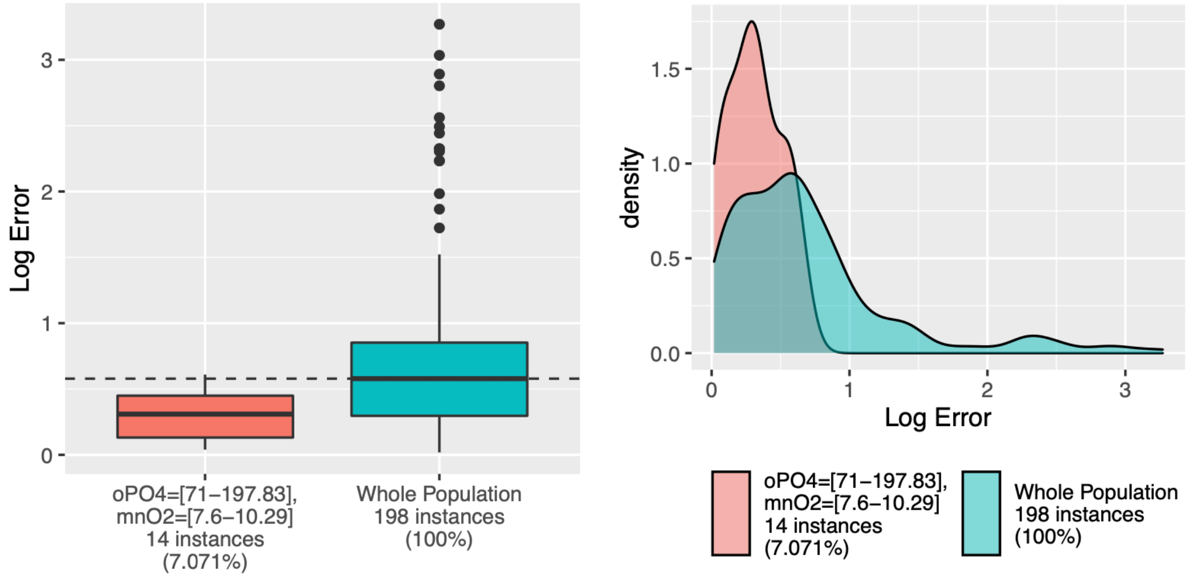


(a) Boxplot.

(b) Density plot.

Figure 19: EDR from dataset A1 (cf. Table 1) to analyze subgroup $NH4 = [22.5–81], mnO2 = [7.6–10.29], oPO4 = [7.4–31.33]$, trained with a GBM model (cf. Table 2).

Oppositely, as seen in Figure 20, the subgroup $oPO4 = [71–197.83], mnO2 = [7.6–10.29]$ presents smaller errors than expected. For instance, by looking at Figure 20a, we can see that the subgroup values are centered around a lower value and are more contained, since the interquartile range (IQR) is smaller. In fact, the global median is slightly shorter than the upper whisker of the distribution of the subgroup. Additionally, by inspecting Figure 20b, we can confirm that the distribution of values of the subgroup is centered around smaller values and rapidly decreases after reaching the maximum density.



(a) Boxplot.

(b) Density plot.

Figure 20: EDR from dataset A1 (cf. Table 1) to analyze subgroup $oPO4 = [71\text{–}197.83], mnO2 = [7.6\text{–}10.29]$, trained with a GBM model (cf. Table 2).

Thus, unless the multivariate variants of EDPs were used, it would be difficult to predict the described behaviors for these subgroups, as the univariate EDPs do not present enough information.

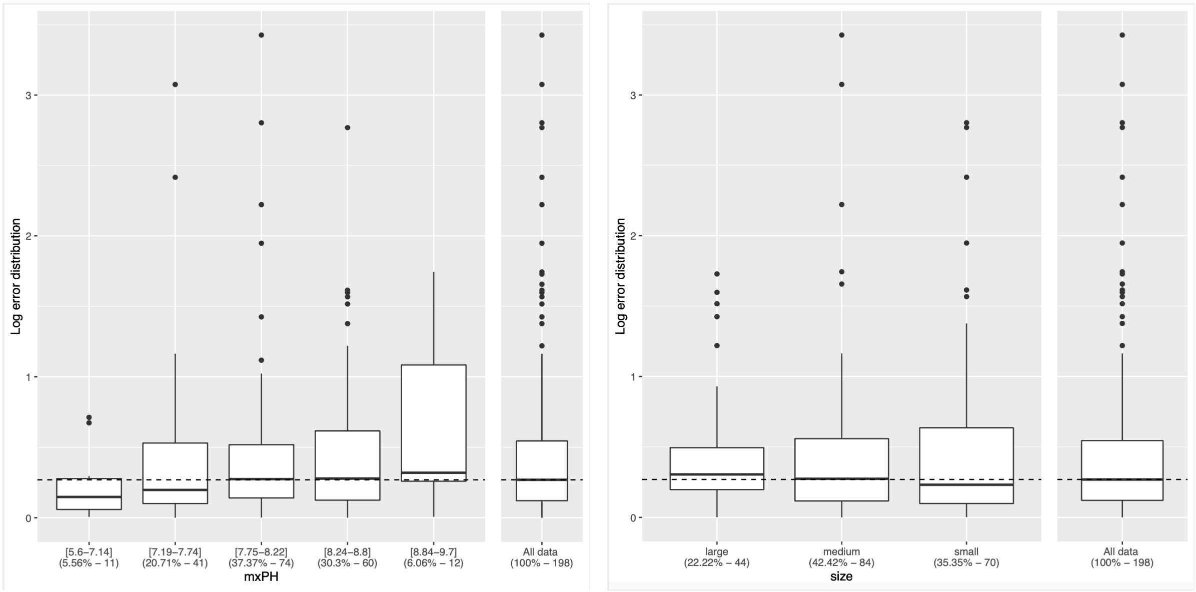
4.5.8.2 A2

In this example, we utilize dataset *A2* trained using an ANN model and set the minimum value of support as 5% per rule. By observing Figure 21, we can clearly see that the bins $mxPH = [8.24 - 8.8]$ (Figure 21a) and $size = small$ (Figure 21b) do not present a significant difference in comparison to the global performance. However, both are characterized for having slightly higher Q3 values. In addition, $size = small$ also presents lower values for the median and Q1 of the distribution.

Nonetheless, with the aid of DRs, we can identify that a combination of these characteristics corresponds to a subgroup with an error distribution tending to higher values. Although the dimensions of the dataset are small, the subgroup has considerable representation, illustrating 6% of the total data, 20% of $mxPH = [8.24 - 8.8]$ and 17% of $size = small$.

4.5.8.3 A3

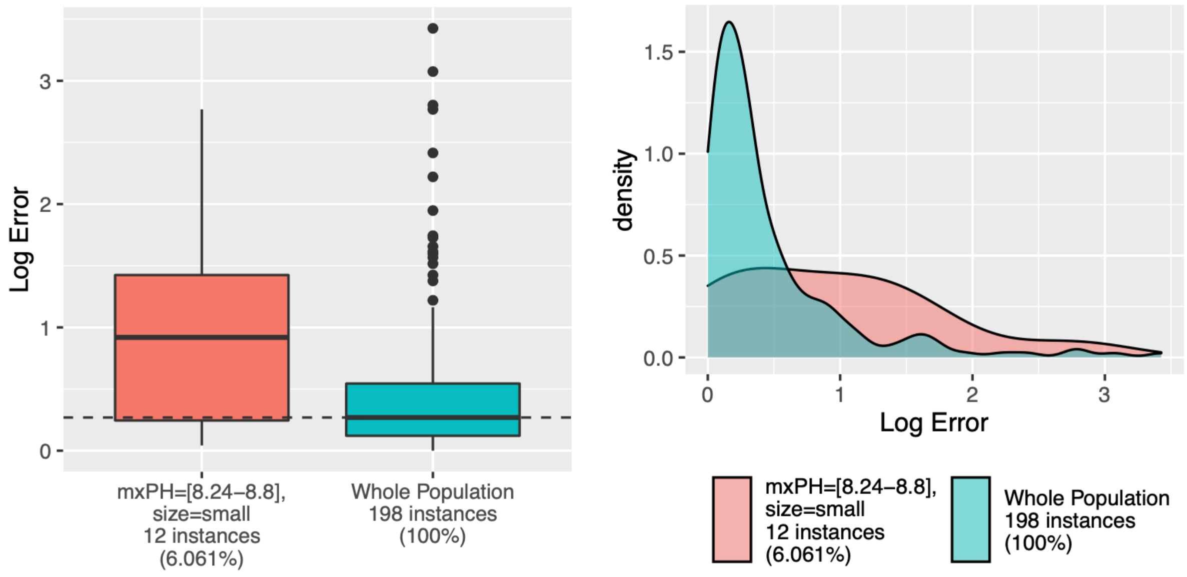
In addition to using the same discovery parameters as before, a RF model was used for this example of dataset *A3*. Observing its EDPs in Figure 23, one can notice that *medium* and *high* values of variable



(a) Feature *mxPH*.

(b) Feature *size*.

Figure 21: EDPs from dataset A2 (cf. Table 1) to analyze features *mxPH* and *size*, trained with an ANN model (cf. Table 2).

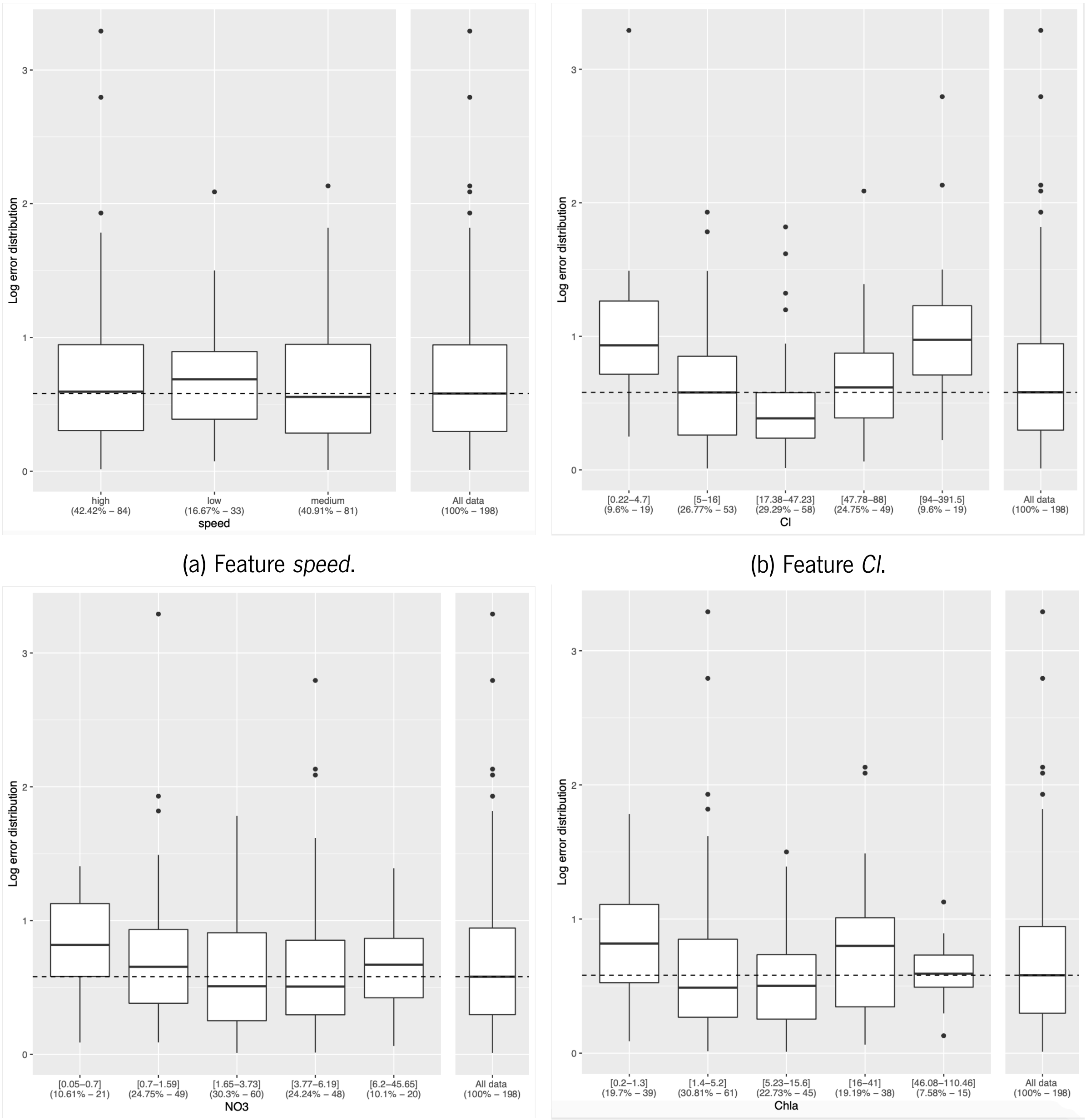


(a) Boxplot.

(b) Density plot.

Figure 22: EDR from dataset A2 (cf. Table 1) to analyze subgroup $mxPH = [8.24 - 8.8], size = small$, trained with an ANN model (cf. Table 2).

speed (Figure 23a) do not present representative differences in relation to the average behavior. Moreover, feature *Chla* (Figure 23d) presents considerable higher error values for $Chla = [0.2 - 1.3]$ and feature *NO3* (Figure 23c), specifically $NO3 = [0.7 - 1.59]$, slightly higher than the whole data. Furthermore, Figure 23b depicts the EDP of feature *Cl*, where it is clear that $Cl = [17.38 - 47.23]$ is characterized for having lower error values than average. In fact, the Q3 of this distribution is marginally smaller than the global median.



(a) Feature *speed*.

(b) Feature *Cl*.

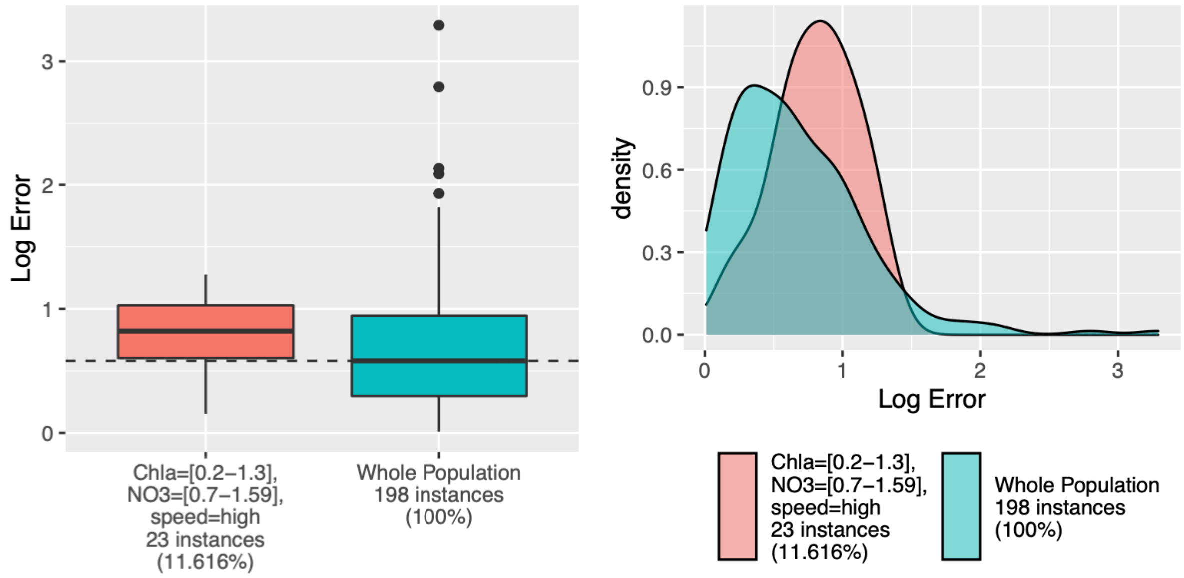
(c) Feature *NO3*.

(d) Feature *Chla*.

Figure 23: EDPs from dataset A3 (cf. Table 1) to analyze features *speed*, *Cl*, *NO3* and *Chla*, trained with a RF model (cf. Table 2).

By utilizing EDRs, in Figure 24 we can see that by combining the characteristics $speed = high$, $Chla = [0.2 - 1.3]$ and $NO3 = [0.7 - 1.59]$, the model produces higher errors than expected. As a matter of fact, the difference between the medians is relevant, as the median of the subgroup is around 0.8, while the global is close to 0.55. The latter being somewhat equal in value to the Q1 of the subgroup.

A second example can be seen in Figure 25, depicting subgroup $Cl = [17.38 - 47.23], speed = medium$. Here, a better performance than expected is seen. Besides that, we can see that this group has a considerable dimension, representing 14.6% of all data, 36% of $speed = medium$ and 50% of $Cl = [17.38 - 47.23]$. This high representation has probably an impact on the EDPs, especially on the latter example, as the subgroup presents even smaller errors on average than the ones expected by

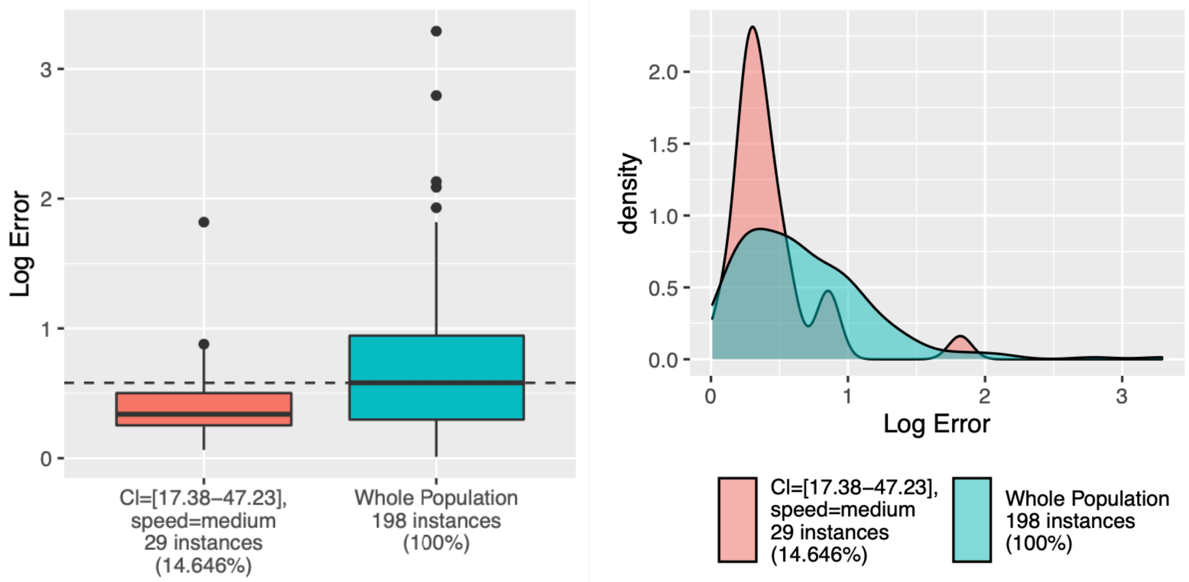


(a) Boxplot.

(b) Density plot.

Figure 24: EDR from dataset A3 (cf. Table 1) to analyze subgroup $speed = high$, $Chla = [0.2 - 1.3]$ and $NO3 = [0.7 - 1.59]$, trained with a RF model (cf. Table 2).

analyzing Figure 23b.



(a) Boxplot.

(b) Density plot.

Figure 25: EDR from dataset A3 (cf. Table 1) to analyze subgroup $Cl = [17.38 - 47.23], speed = medium$, trained with a RF model (cf. Table 2).

4.5.8.4 Abalone

Similarly to EDPs, the usage of DRs allows us to find distinct regions with only one variable. For example, take dataset Abalone, trained with a SVM and a minimum value of support of 5%. Figure 26 depicts the EDP to analyze feature *sex* and clearly demonstrates that $sex = I$ (Infant) has slightly better performance than anticipated by analyzing the model as a whole.

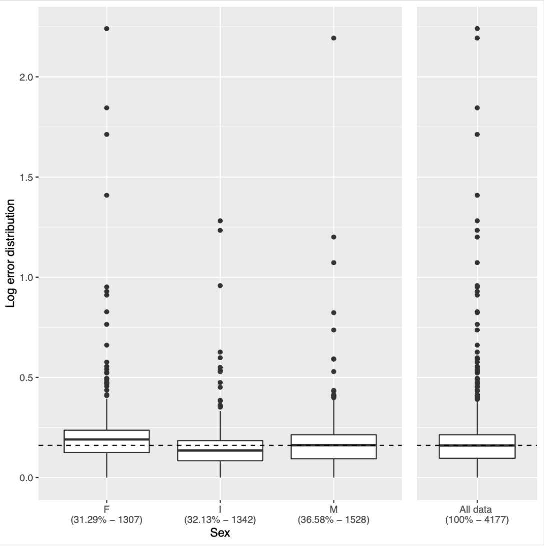
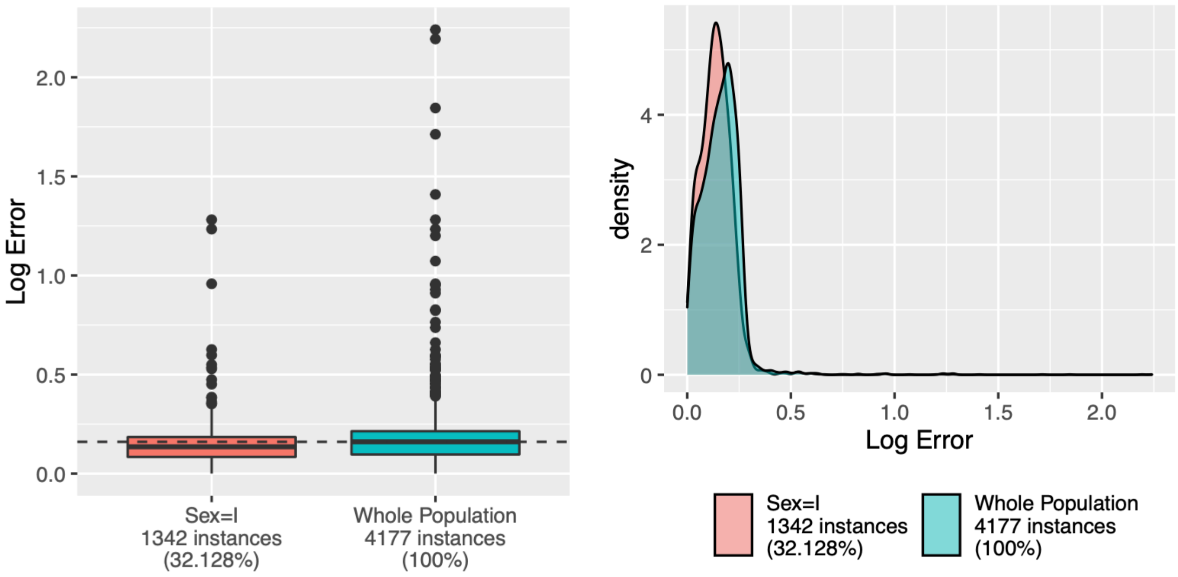


Figure 26: EDP from dataset Abalone (cf. Table 1) to analyze feature *sex*, trained with a SVM model (cf. Table 2).

With the aid of EDRs, this situation is also uncovered, as witnessed in Figure 27. It is important to note that, as every other rule, the minimum value of support is extremely important, as this acts as a filtering method. However, in this case, the subgroup has a representation of 1342 instances, i.e., 32.13% of the whole dataset, easily surpassing the 5% minimum support.



(a) Boxplot.

(b) Density plot.

Figure 27: EDR from dataset Abalone (cf. Table 1) to analyze subgroup $sex = I$, trained with a SVM model (cf. Table 2).

4.5.8.5 Acceleration

The great advantage of using EDRs lies in the fact that these do not have dimensionality restrictions like EDPs, allowing the analysis of interactions between more than just three variables. Due the a high number of instances and features in this dataset, the minimum filtering value of support was increased to 10%. Moreover, regardless of training the four models, the one that allowed us to present the most interesting details was the SVM. Observing some EDPs in Figure 28, one can recognize that, for instance, $attribute6 = [12.3 - 18.7]$ (Figure 28b), $attribute13 = [30.5 - 37.2]$ (Figure 28c) and $attribute14 = [3.26 - 3.68]$ (Figure 28d) have a slightly better performance than global, both in terms of median and Q3 values. In addition, $attribute1 = nominal1$ (Figure 28a) is similar to the supra mentioned, despite having a considerable number of outliers.

By connecting the four bins mentioned, EDRs detect a subgroup with a better performance than globally expected, both in terms of median, as in Q1 and Q3 values, as depicted in Figure 29. Additionally, in Figure 29a, it is visible that the global median is greater than the Q3 value of this group. Moreover, it is important to emphasize that this is a set with a considerable representation, corresponding to 10.6% of the total data, 15.6% of $attribute1 = nominal1$, 19.5% of $attribute6 = [12.3 - 18.7]$, 28.3% of $attribute13 = [30.5 - 37.2]$ and a hefty 41.2% of $attribute14 = [3.26 - 3.68]$.

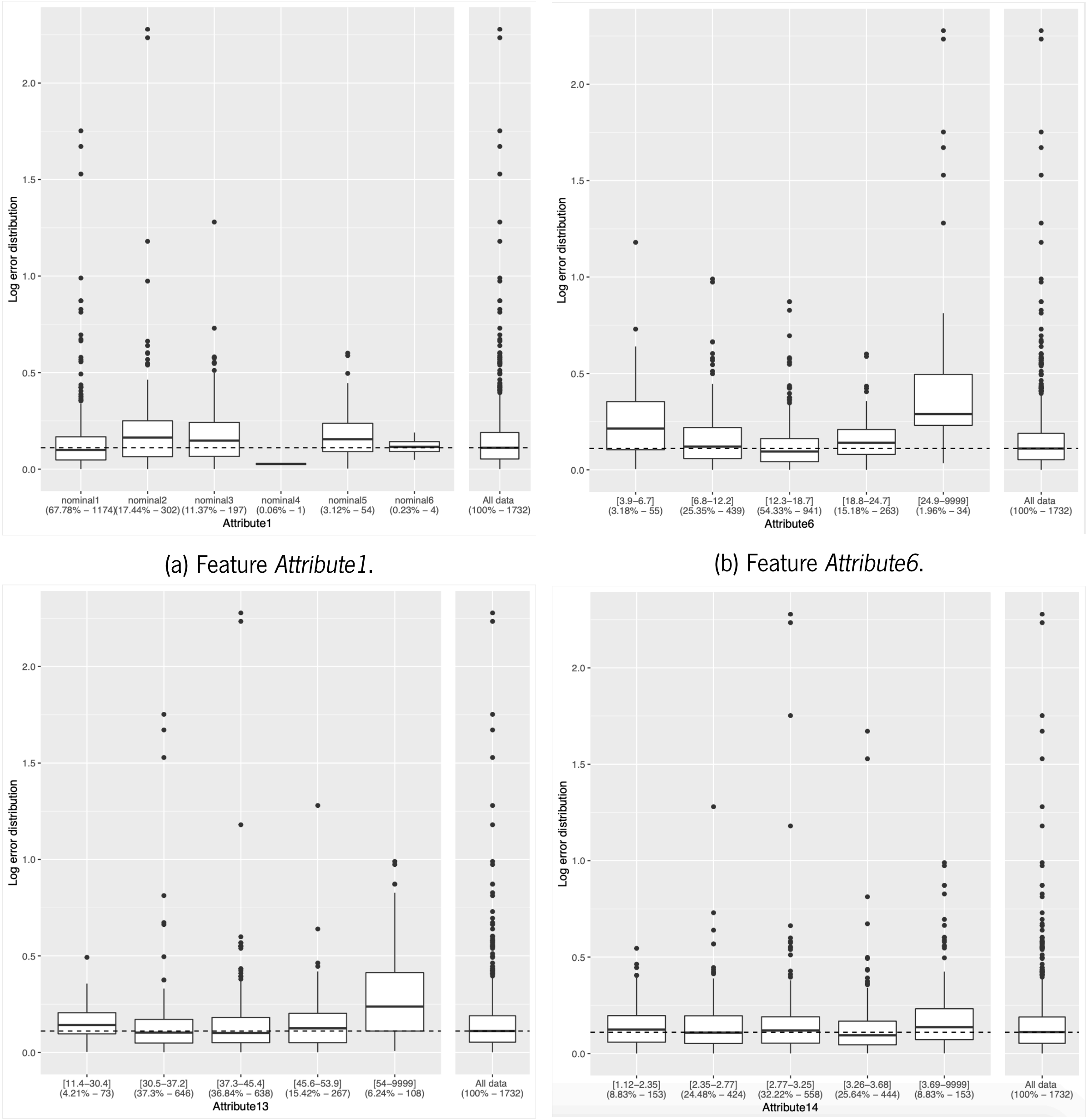
It is important to highlight that, even by using the multivariate versions of EDPs, this interaction would not be discovered, as it is composed by a combination of four features.

4.5.8.6 Airfoild

Using the same discovery parameters as in Section 4.5.8.5 applied to a RF model trained with this dataset, an interesting example can be found in the subgroup $Suction = [0.001 - 0.003]$, $AngleOfAttack = [0 - 2]$. This represents 16.1% of all data, 58% of $AngleOfAttack = [0 - 2]$ and 68% of $Suction = [0.001 - 0.003]$. By observing the EDPs for features *AngleOfAttack* and *Suction*, in Figures 30a and 30b, respectively, it is noticeable that the bins depicted in the subgroup suggest a better performance than overall.

Moreover, as these bins were select by EDRs as having interesting behaviors, we can analyze their density plots in Figure 31. This type of visualization might help users in understanding some small differences between the groups. For instance, the error values of $Suction = [0.001 - 0.003]$ have a higher density around smaller values than the values of $AngleOfAttack = [0 - 2]$, possibly explaining the lower median value of the first.

It is also important to analyze the subgroup that is characterized by both bins, as this can be seen as a derivation of the original bins. Focusing on Figure 32, specifically in Figure 32b, it is noticeable that the higher value of density is superior to 3.5, around the same error values as the ones in Figure 31. In addition, the individual higher values of density were both smaller than the peak of this combination, as well as having more instances with higher error values as seen on the right tail of their distributions. Thus, the conjunction of these two feature values yields more accurate results.



(a) Feature *Attribute1*.

(b) Feature *Attribute6*.

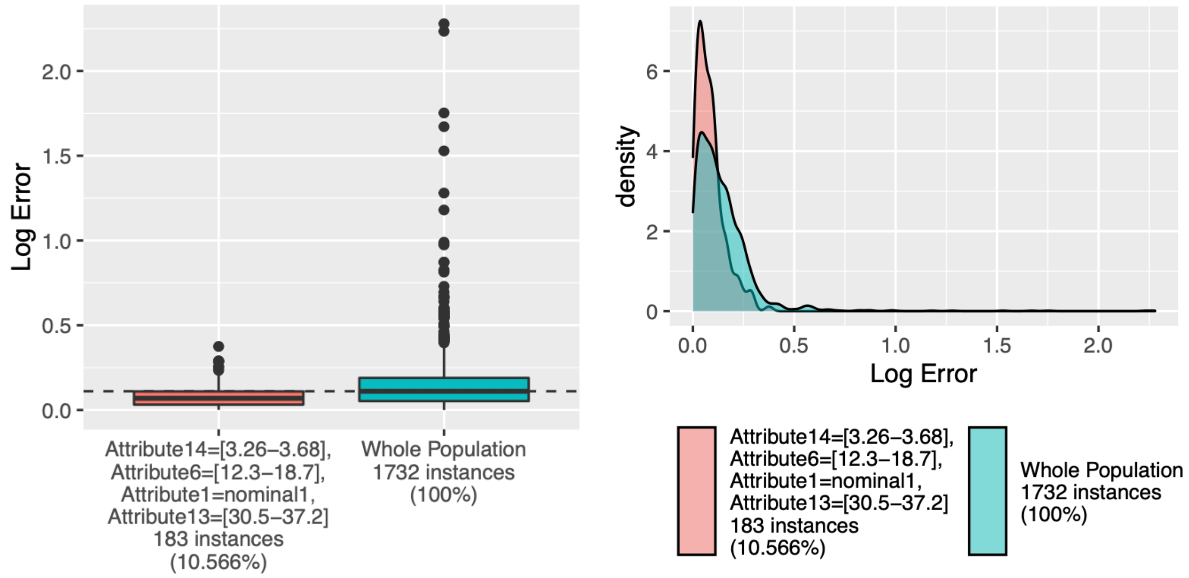
(c) Feature *Attribute13*.

(d) Feature *Attribute14*.

Figure 28: EDPs from dataset Acceleration (cf. Table 1) to analyze features *Attribute1*, *Attribute6*, *Attribute13* and *Attribute14*, trained with a SVM model (cf. Table 2).

4.5.8.7 Availpwr

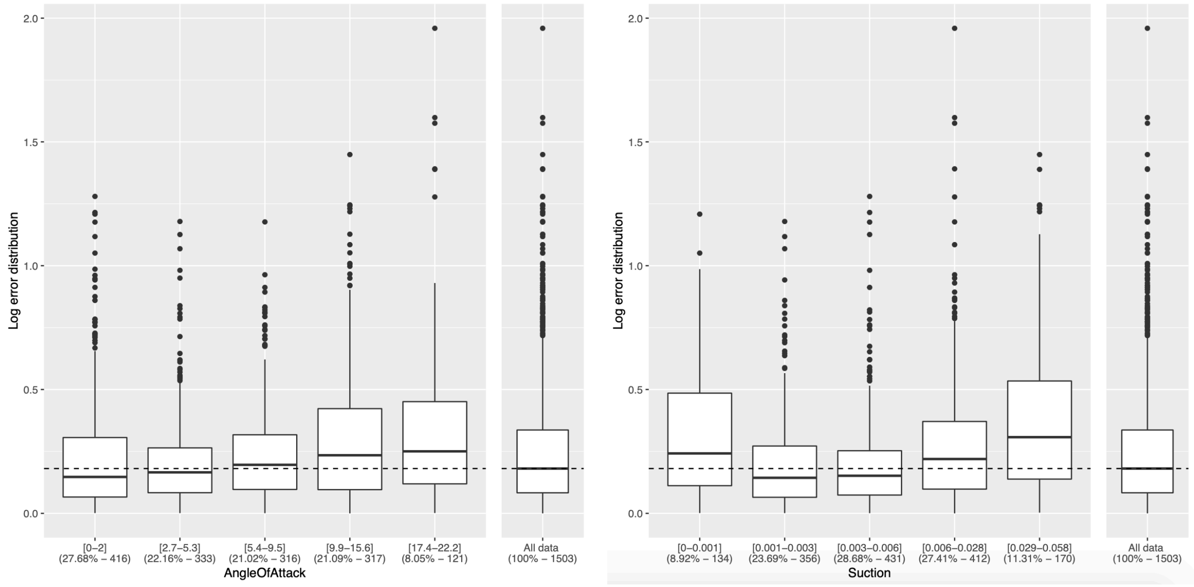
Figure 33 depicts an interesting example, as it is visible that the subgroup and general error distributions are similar to some extent, i.e., the main peaks have a similar density values (Figure 33b). However, the second peak of density of the subgroup occurs for much lower errors (close to 2.0) than for the general population (close to 4.0). This difference leads the subgroup to have a better performance than the general one, since it has a higher instance density for lower errors. Moreover, we can confirm this behavior tending for smaller errors by analyzing the boxplot representation of this EDR in Figure 33a. The latter shows not only a smaller median value, but also Q1 and Q3 values for the subgroup, corroborating the



(a) Boxplot.

(b) Density plot.

Figure 29: EDR from dataset Acceleration (cf. Table 1) to analyze subgroup $attribute14 = [3.26 - 3.68], attribute6 = [12.3 - 18.7], attribute1 = nominal1, attribute13 = [30.5 - 37.2]$, trained with a SVM model (cf. Table 2).



(a) Feature *AngleOfAttack*.

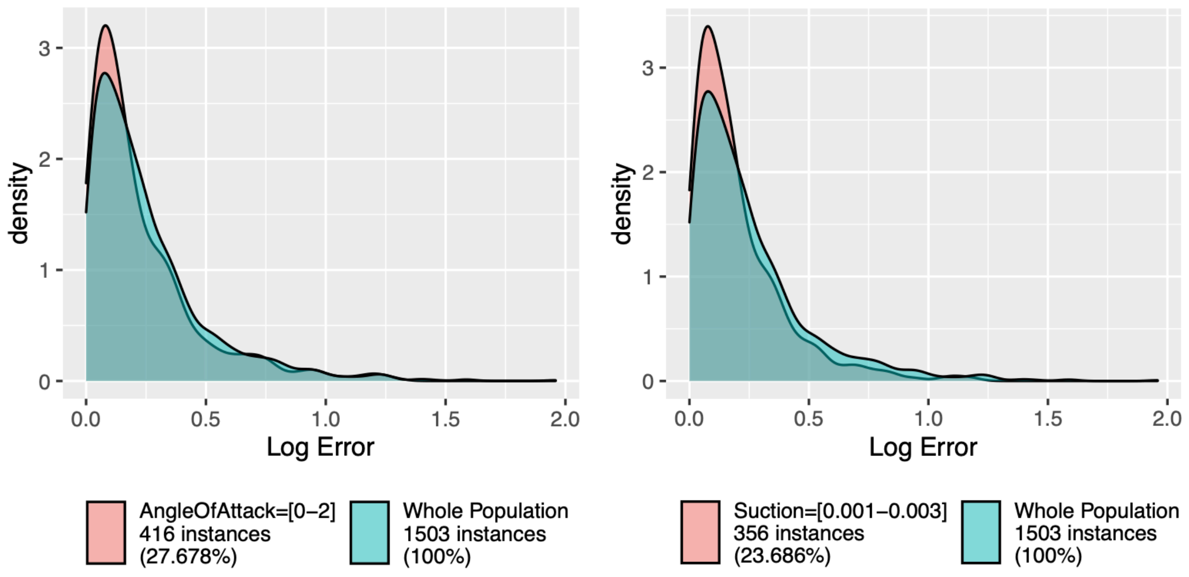
(b) Feature *Suction*.

Figure 30: EDPs from dataset Airfoild (cf. Table 1) to analyze features *AngleOfAttack* and *Suction*, trained with a RF model (cf. Table 2).

above analysis.

4.5.8.8 Boston

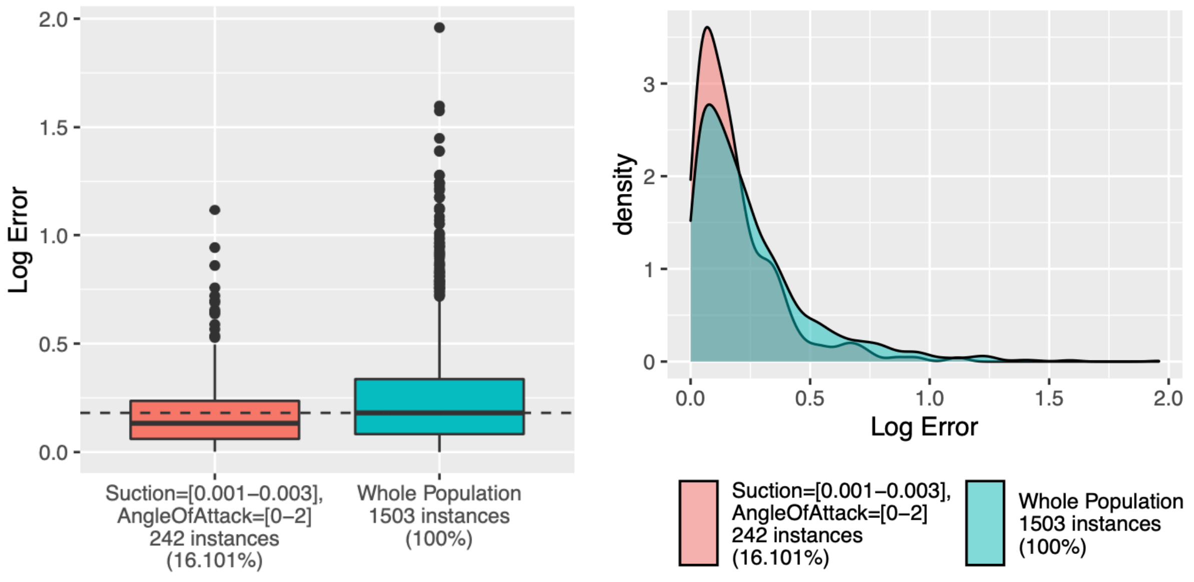
This example utilizes a SVM model trained with dataset *Boston* and setting a minimum support of 5%. By observing some of its EDPs in Figure 34, one can spot some interesting scenarios. For instance,



(a) Feature *AngleOfAttack*.

(b) Feature *Suction*.

Figure 31: EDRs from dataset Airfoild (cf. Table 1) to analyze subgroups $Suction = [0.001 - 0.003]$, and $AngleOfAttack = [0 - 2]$, trained with a RF model (cf. Table 2).



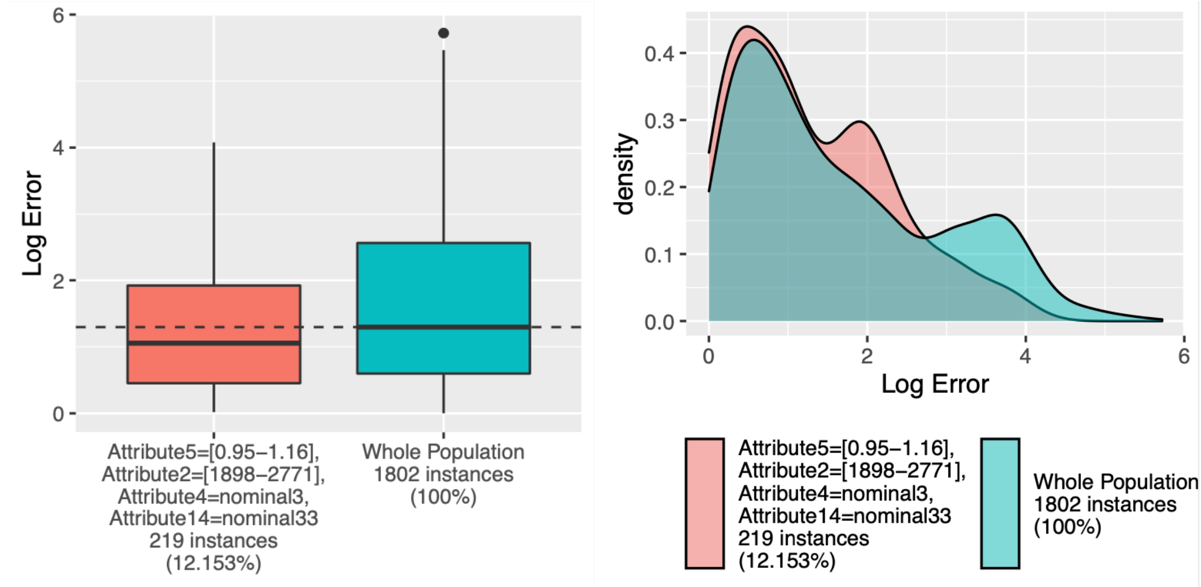
(a) Boxplot.

(b) Density plot.

Figure 32: EDR from dataset Airfoild (cf. Table 1) to analyze subgroup $Suction = [0.001 - 0.003], AngleOfAttack = [0 - 2]$, trained with a RF model (cf. Table 2).

in Figure 34a it is clear that $age = [80.8 - 94.9]$ is similar to the global distribution, despite having a slight tendency for larger errors (Q3 of the distribution is higher than the global one). Conversely, $dis = [2.21 - 3.65]$, in Figure 34b, tends to smaller errors, as the Q3 and Q1 of the distribution are smaller than the global ones. Additionally, its median is also slightly lower. Lastly, in Figure 34c, we see that $tax = [422 - 711]$ has a slight tendency for higher errors, due to having a higher value for its Q3.

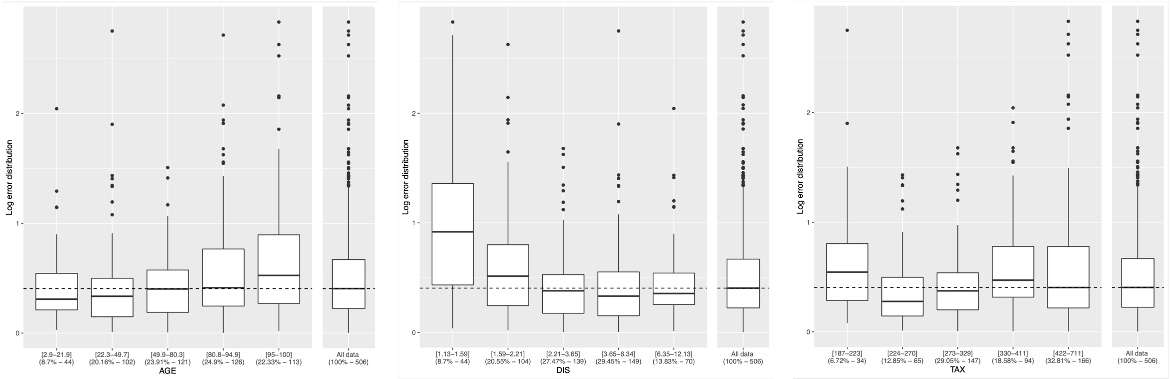
Once again, by using EDRs, we detect a subgroup characterized by $age = [80.8 - 94.9], dis =$



(a) Boxplot.

(b) Density plot.

Figure 33: EDR from dataset AvailPwr (cf. Table 1) to analyze subgroup $attribute5 = [0.95 - 1.16], attribute2 = [1898 - 2771], attribute4 = nominal3, attribute14 = nominal33$, trained with an ANN model (cf. Table 2).



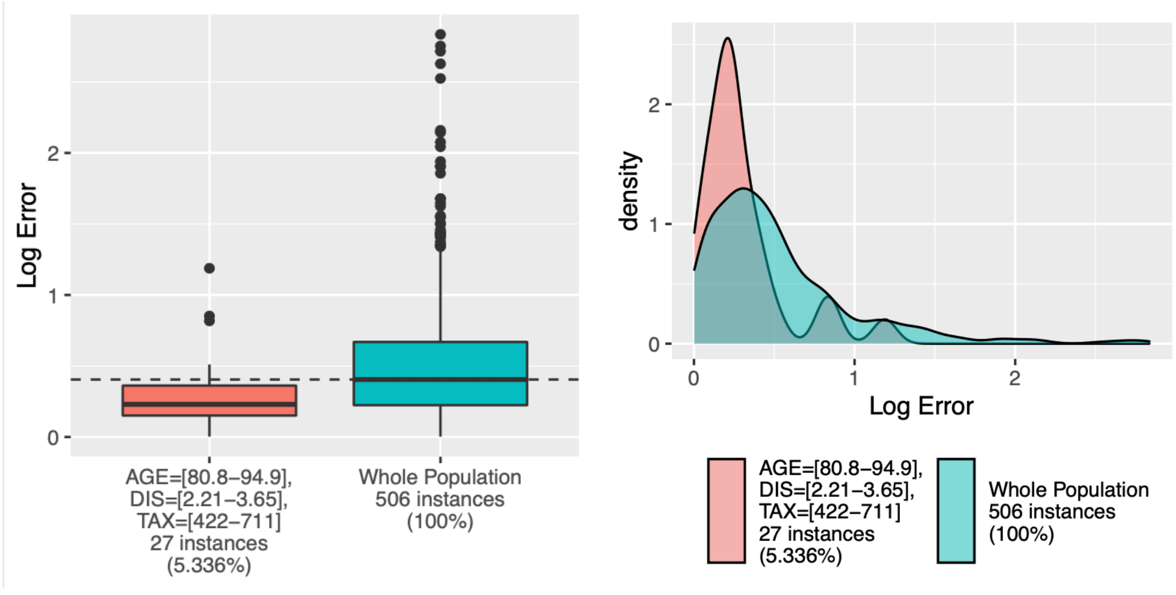
(a) Feature *age*.

(b) Feature *dis*.

(c) Feature *tax*.

Figure 34: EDPs from dataset Boston (cf. Table 1) to analyze features *age*, *dis* and *tax*, trained with a SVM model (cf. Table 2).

$[2.21 - 3.65], tax = [422 - 711]$, as seen in Figure 35. This subgroup represents 5.3% of the complete dataset, 21.4% of $age = [80.8 - 94.9]$, 19.4% of $dis = [2.21 - 3.65]$ and 16.3% of $tax = [422 - 711]$. Additionally, it presents a better performance in comparison with the complete dataset in general, even having a Q3 value below the general median. Thus, a combination that, at first did not seem to deviate much from the overall performance and had two of its individual conditions defined by lower accuracy than expected, ends up showing very low error values.



(a) Boxplot.

(b) Density plot.

Figure 35: EDR from dataset Boston (cf. Table 1) to analyze subgroup $age = [80.8 - 94.9], dis = [2.21 - 3.65], tax = [422 - 711]$, trained with a SVM model (cf. Table 2).

4.5.8.9 ConcreteStrength

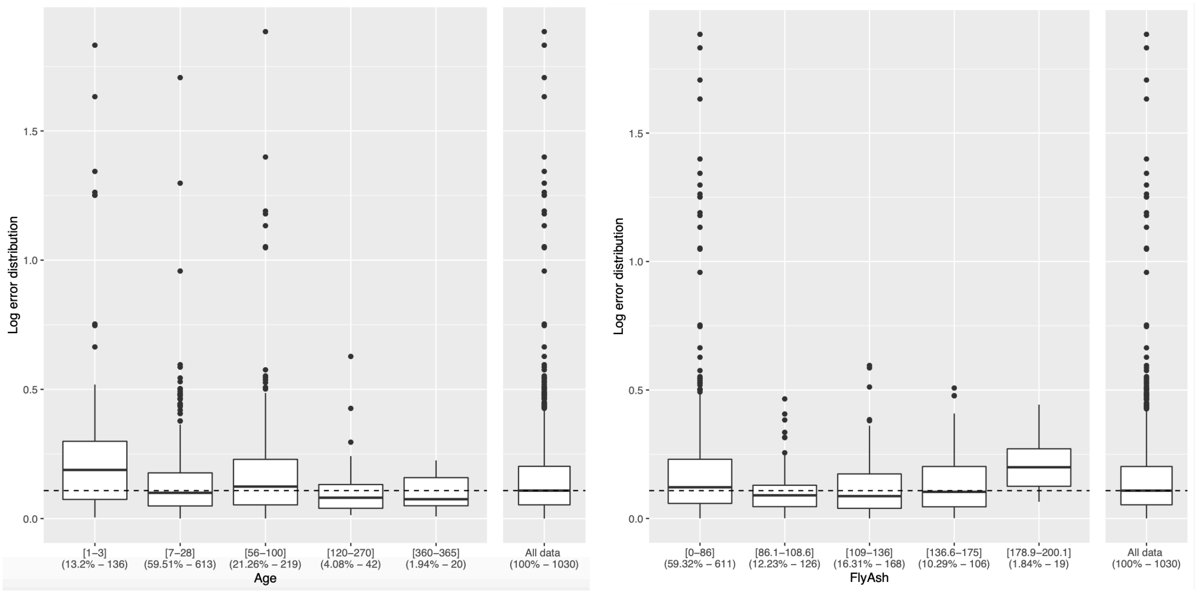
Another interesting example can be seen using dataset *ConcreteStrength*, as it consists of a dataset with considerable dimensions. By analyzing some EDPs produced using a GBM model in Figure 36, we can see that, for instance, $Age = [56 - 100]$ (Figure 36a) and $FlyAsh = [0 - 86]$ (Figure 36b) contain higher values for the median and Q3, leading to a tendency to worse performance than expected. Moreover, their Q1 values are similar to expected and these are characterized by a considerable number of outliers.

Additionally, instances that contain these two attribute values are expected to have worse performance than predicted by the analysis of the EDPs, as seen in Figure 37. This can be seen by the fact that not only the median and Q3 values of the distribution are higher than the global ones, but Q1 as well. Furthermore, it is interesting to notice that every outlier with error values superior to 1.0 present in $Age = [56 - 100]$ (Figure 36a) are also present in the subgroup. Likewise, by viewing the density plot version, it is noticeable that even though the error value where the peaks occur are similar, the peak of the subgroup is smaller and is followed by a less steep descent, explaining, once again, the worse performance.

4.5.8.10 CpuSm

By observing the subgroup defined by $swrite = [71 - 230], exec = [0 - 3]$ in Figure 38, it is visible that the overall performance to be expected is an improvement regarding the global one. In addition, this combination of values represents 40.9% of all data, 52% of $exec = [0 - 3]$ and 74% of $swrite = [71 - 230]$.

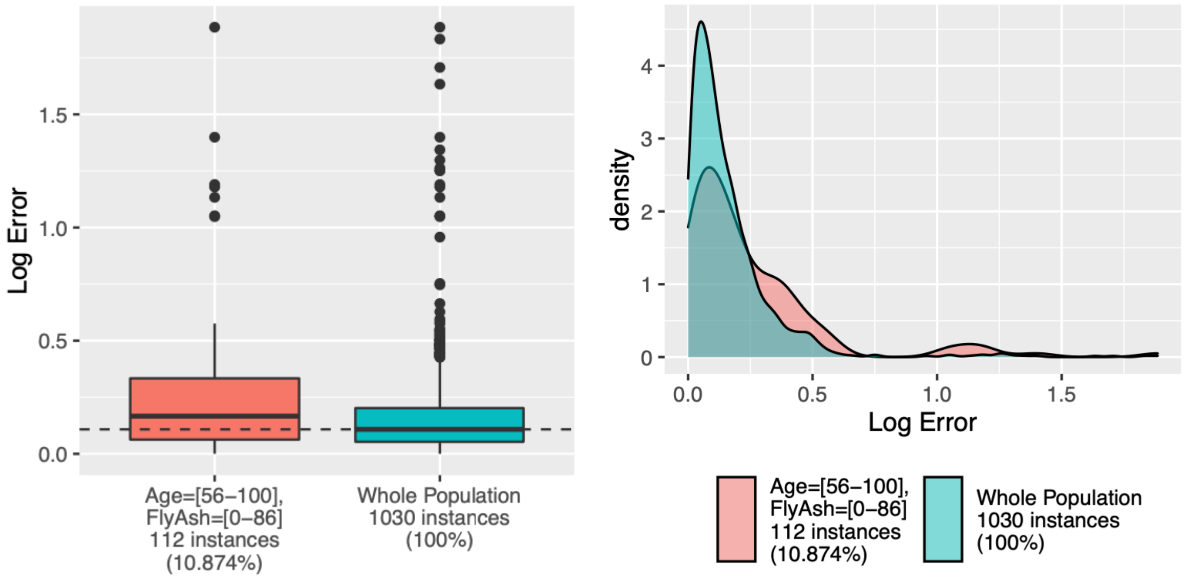
Moreover, its respective EDPs, show that $exec = [0 - 3]$, represented in Figure 39a, has similar error



(a) Feature *Age*.

(b) Feature *FlyAsh*.

Figure 36: EDPs from dataset ConcreteStrength (cf. Table 1) to analyze features *Age* and *FlyAsh*, trained with a GBM model (cf. Table 2).

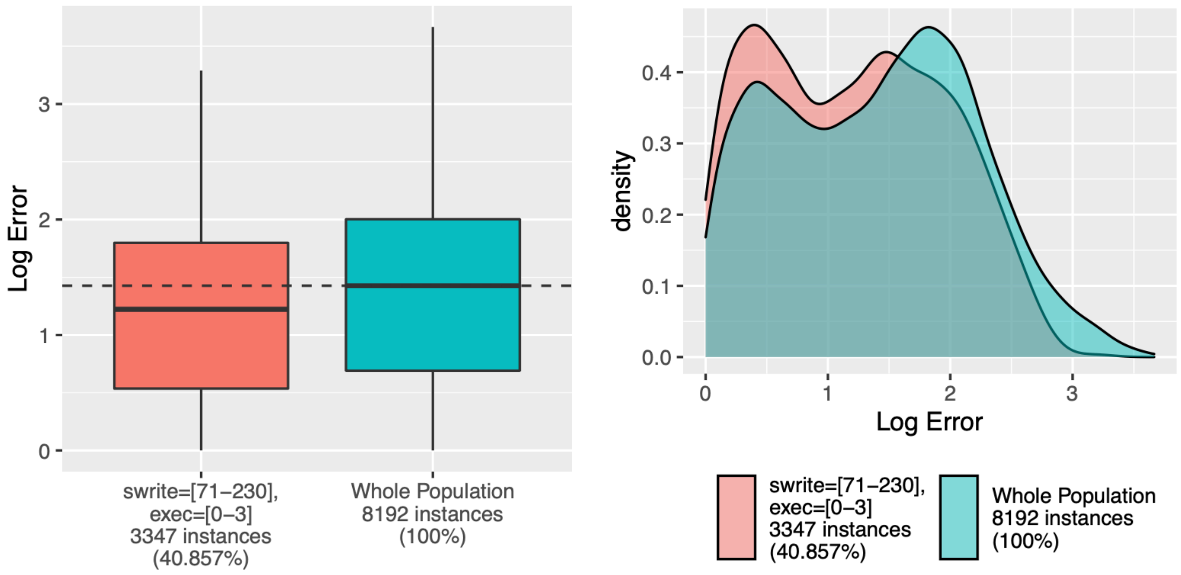


(a) Boxplot.

(b) Density plot.

Figure 37: EDR from dataset ConcreteStrength (cf. Table 1) to analyze subgroup $Age = [56 - 100], FlyAsh = [0 - 86]$, trained with a GBM model (cf. Table 2).

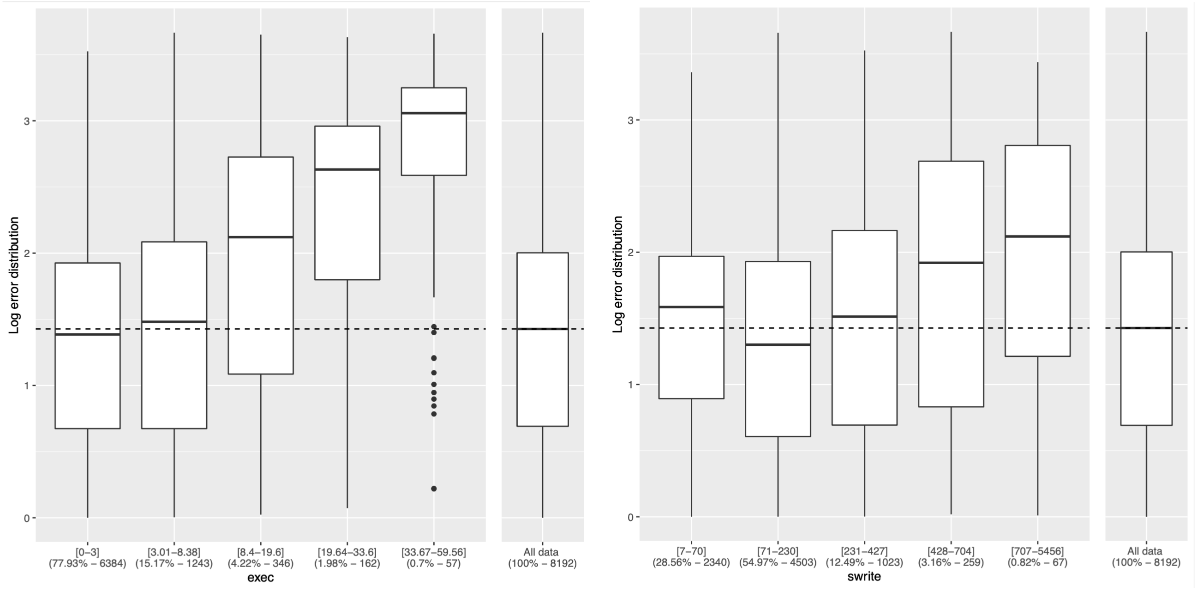
values to the complete dataset. However, Figure 39b, namely $swrite = [71 - 230]$ produces a more interesting result, as the boxplot is quite similar to the subgroup, possibly due to the latter representing almost three quarters of the first.



(a) Boxplot.

(b) Density plot.

Figure 38: EDR from dataset CpuSm (cf. Table 1) to analyze subgroup $swrite = [71 - 230], exec = [0 - 3]$, trained with an ANN model (cf. Table 2).



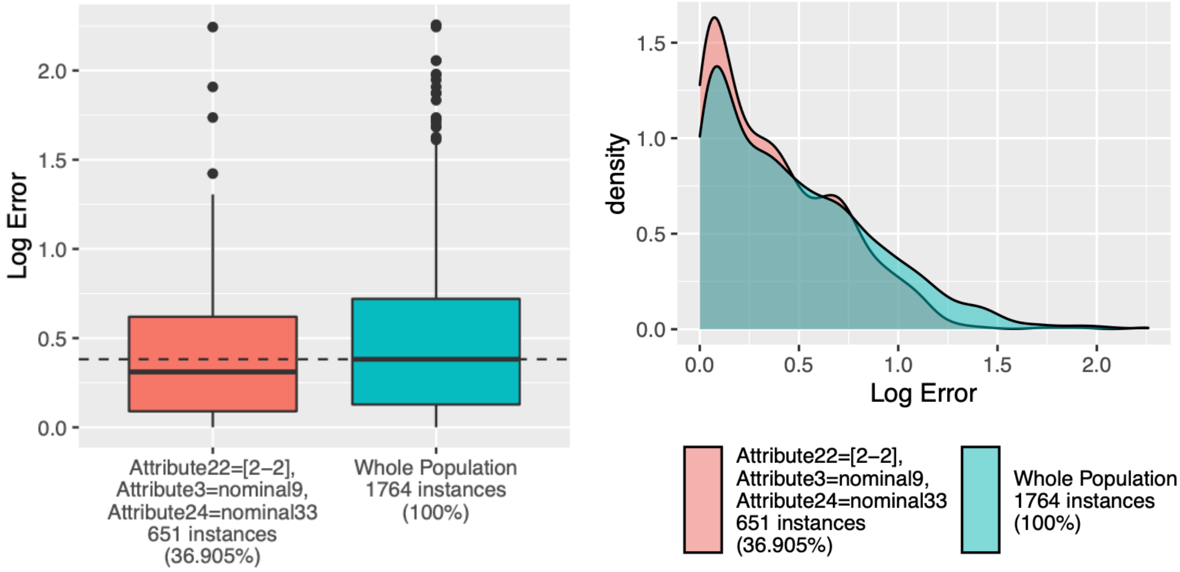
(a) Feature *exec*.

(b) Feature *swrite*.

Figure 39: EDPs from dataset CpuSm (cf. Table 1) to analyze features *exec* and *swrite*, trained with an ANN model (cf. Table 2).

4.5.8.11 FuelCons

In order to identify subgroups with higher density, i.e., with a larger representation, the minimum values of support and pruning filter used were higher. Consequently reducing the number of EDRs discovered. Even so, close to 580 distinct subgroups were generated using a minimum support of 35%, showing the agility and power of the use of rules. An example of these is the subgroup $attribute22 = [2 - 2], attribute3 = nominal9, attribute24 = nominal33$, in Figure 40, identified by smaller errors than expected and a representation of 36.9% of the total data. Note that for this specific case, to find an interaction with three variables using EDPs and generating every plot of all interactions with 3 variables, it would be necessary to generate and analyze $\frac{37!}{3!\times(37-3)!} = 7770$ distinct plots.



(a) Boxplot.

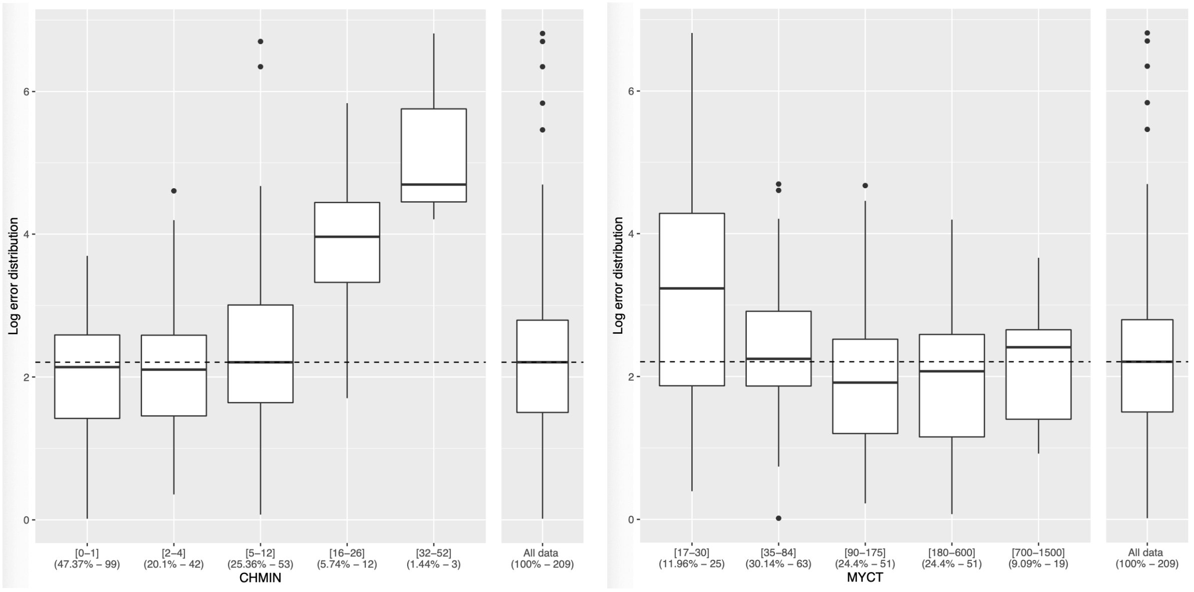
(b) Density plot.

Figure 40: EDR from dataset FuelCons (cf. Table 1) to analyze subgroup $attribute22 = [2-2], attribute3 = nominal9, attribute24 = nominal33$, trained with an ANN model (cf. Table 2).

4.5.8.12 MachineCpu

Lastly, this example consists of a dataset with a relatively small number of instances and features. This hinders the discovery of interesting subgroups of data, as the bins do not have enough support, despite being different in terms of shape. Examples of this are visible in Figure 41, more concretely in the EDP of feature *chmin* in Figure 41a, as $chmin = [16 - 26]$ or $chmin = [32 - 52]$ are considerably different, but not populated enough. Moreover, the bin $chmin = [0 - 1]$ displays a better performance than overall, having smaller values for the quartiles and median than the ones of the global distribution. Similarly, in Figure 41b, $myct = [90 - 175]$ is also characterized by better performance, having minor values for the percentiles in comparison to the first.

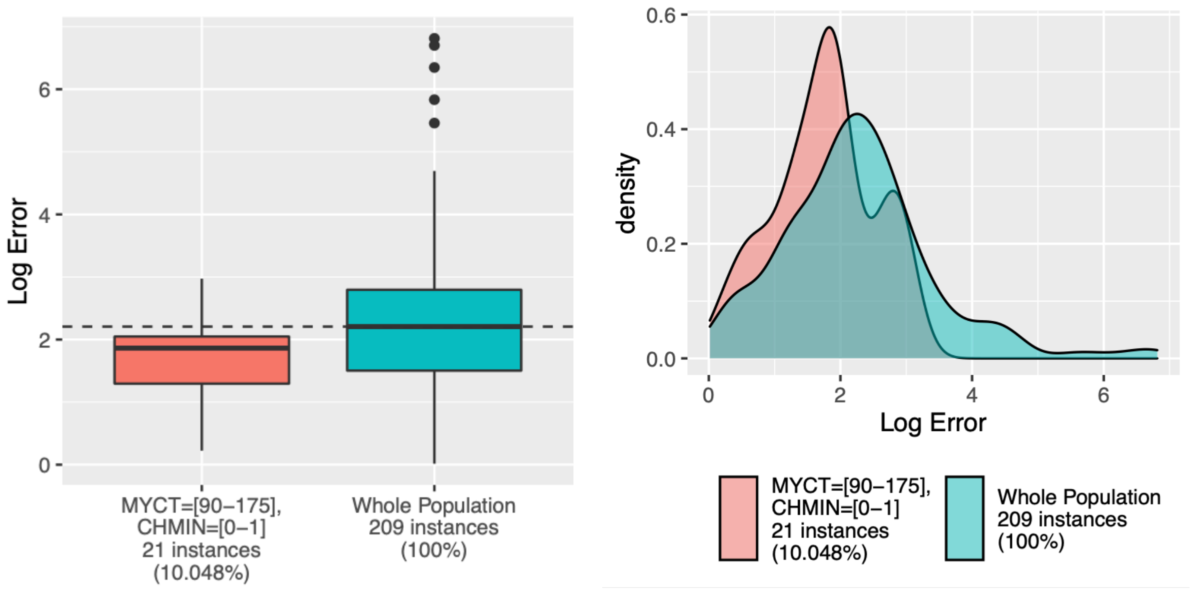
Nevertheless, the combination of both consists of a subgroup with considerably better performance than the general, with the global median being higher than the Q3 of the error distribution of this group, as seen in Figure 42a. In addition, this is visible in the density plot depicted Figure 42b. Here, for error values close to 3.5 the subgroup no longer shows instances, but the global distribution is still in a decreasing phase of density.



(a) Feature *chmin*.

(b) Feature *myct*.

Figure 41: EDPs from dataset MachineCpu (cf. Table 1) to analyze features *chmin* and *myct*, trained with a SVM model (cf. Table 2).



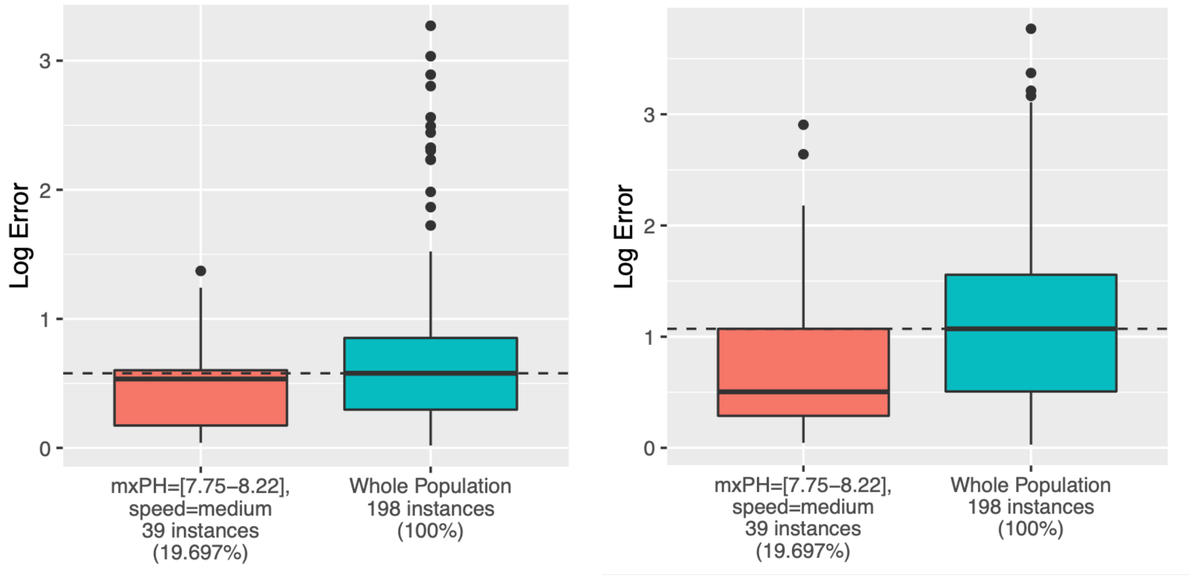
(a) Boxplot.

(b) Density plot.

Figure 42: EDR from dataset MachineCpu (cf. Table 1) to analyze subgroup $myct = [90 - 175], chmin = [0 - 1]$, trained with a SVM model (cf. Table 2).

4.5.9 Comparison of Equal Subgroups on Different Models

One aspect that may be important is the comparison of the behavior of the same subgroups in different models. However, the same subgroups do not always behave differently enough to generate a rule on them, so it is not always possible to perform this comparison. Nevertheless, we can observe an example from dataset *A1*, comparing a GBM model with a RF. Moreover, the discovery process was conducted using a minimum support of 5%. By analyzing Figure 43, we can easily notice that the GBM model has an overall better performance than the RF, due to having lower values of Q1, median and Q3. In addition, the IQR is also smaller, centering the majority of errors in a smaller and lower interval. However, for subgroup $mxPH = [7.75-8.22], speed = medium$ we can see some interesting situations. For the RF model, seen in Figure 43b, similarly to the global error values, the distribution is more spread. Nonetheless, its median is much lower than the median generated by the GBM model, which is close to its Q3, as seen in Figure 43a.



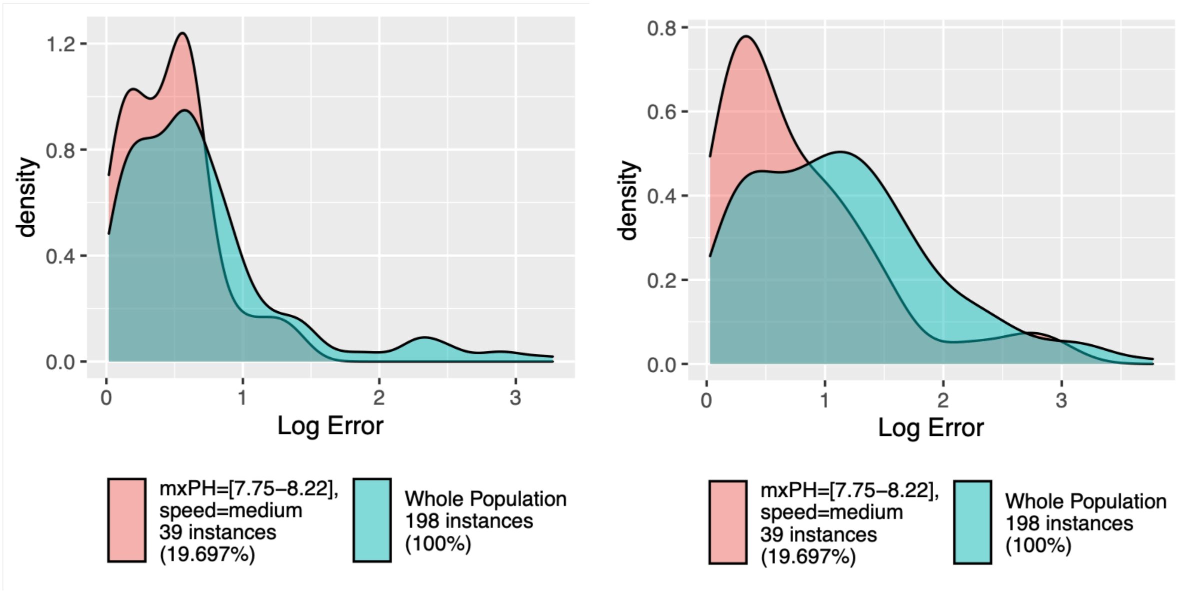
(a) GBM.

(b) RF.

Figure 43: Boxplot EDRs from dataset A1 (cf. Table 1) to analyze subgroup $mxPH = [7.75 - 8.22], speed = medium$, trained with a GBM and a RF models (cf. Table 2).

Moreover, by observing the density plot representation of the subgroup for both models in Figure 44, we can confirm and understand more of the behaviors of these distributions. For instance, for the GBM model, the density of the distributions drops rapidly after reaching its maximum peak. Inversely, for the RF model, the descent is more gradual, leading to the possibility of the existence of higher errors. Global distributions can also be compared, with the GBM model being similar to the subgroup, with a less pronounced drop as seen in Figure 44a. Differently, the distribution from the RF model is much more flat (Figure 44b), explaining the greater IQR.

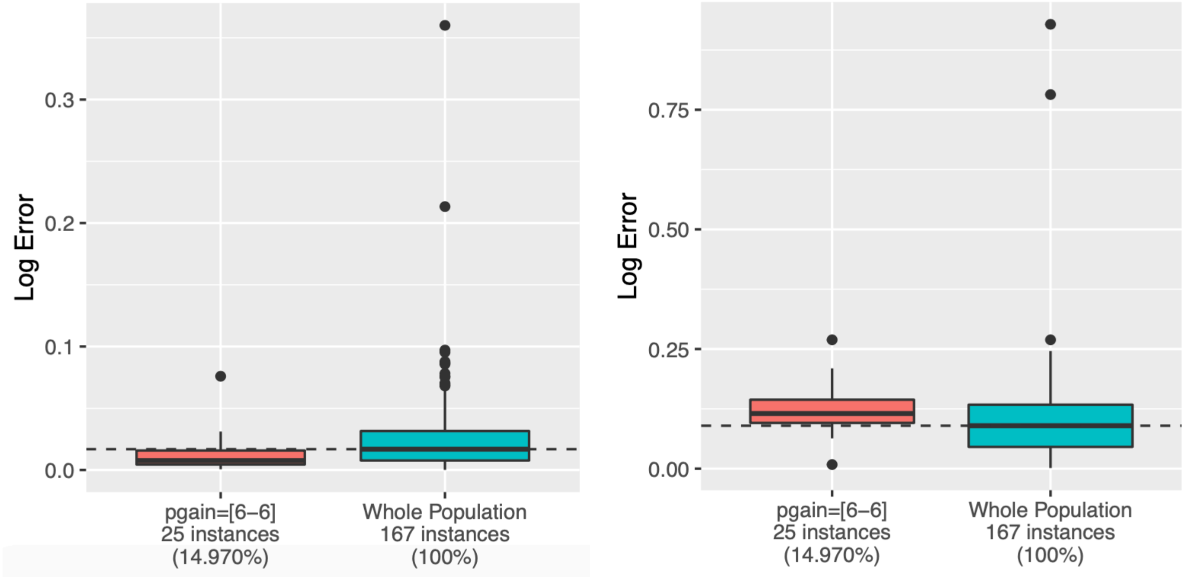
Another curious example can be seen in dataset *Servo*, using the same discovery parameters and trained with an ANN and a SVM models. First, let us analyze the global expected performances in Figure 45. Even though these are relatively similar in shape, the ANN, depicted in Figure 45a, is composed by much lower values of error than the SVM (Figure 45b). Second, we must compare the subgroup defined by $pgain = [6–6]$ in both models. For the first model, it is clear that the subgroup presents a better performance than previously expected by the model itself. However, for the SVM model, the subgroup is now characterized by superior error values than expected, leading to a worse performance than the overall model.



(a) GBM.

(b) RF.

Figure 44: Density plot EDRs from dataset A1 (cf. Table 1) to analyze subgroup $mxPH = [7.75 - 8.22], speed = medium$, trained with a GBM and a RF models (cf. Table 2).



(a) ANN.

(b) SVM.

Figure 45: Boxplot EDRs from dataset Servo (cf. Table 1) to analyze subgroup $pgain = [6–6]$, trained with an ANN and a SVM models (cf. Table 2).

As depicted by these examples, EDRs, like EDPs, are extremely useful to compare the performance of certain subgroups in various models. Although it is not always possible to compare certain subgroups of data between multiple models, as these are not always selected for having a different enough behavior, it may still be useful to compare some combinations of features in order to select a model to use.

4.6 Summary

Various datasets, with variable size and number of instances, both numerical and categorical, were used during the implementation and validation processes. In order to erase any model bias, every dataset was trained with four models, anANN, a GBM, a RF and a SVM.

Further, as numerical features have to be discretized in order to initiate the process of rule discovery, this consisted of a crucial step. Ideally, this procedure is achieved with some level of user or domain specific requirements to divide the values. Due to the number of sets of data being studied, every numerical feature was categorized using quartiles, similar to the default behavior of EDPs, allowing an easy comparison of the results with the latter tool. CAREN, the rule discovery engine used, contains automatic approaches to perform this action. However, the results would not be directly comparable with EDPs, as CAREN does not output the transformed data, only the discovered rules. Additionally, error values were calculated using a CV method, with 10 folds of data, to produce reliable estimates of error.

Thus, we proposed EDRs, a novel tool that utilizes the core ideas of EDPs and DRs to focus on subgroups of data that are interesting to end users. These are composed by three main visualization methods, namely boxplots, histograms and density plots. Another visual approach consists of a network visualization, allowing easy access to the metrics of each rule and to highlight similar rules, i.e., subgroups that share some feature conditions with the selected one. Two other techniques were implemented, performance tables and an extension of counter-factual examples to regression. The first evaluate the values of error on the various cutting points of the distributions, based on reference values, producing one of three possible scenarios: Higher, Equal or Lower. The second practice follows a similar logic, but only focuses on rules that are characterized by a differing behavior in regard to the reference distribution and a EDR that has at least one similarity.

As the proposed tools are based on graphical analysis, producing pleasant results is an important requirement. Consequently, the scale of error values has a wide impact on the results. Three distinct measures were calculated, namely absolute, logarithmic and residual errors. Although the first two always produce the same EDRs, logarithmic errors are easier to analyze. Contrarily, residual error values not only lead to the discovery of a different set of rules, but may produce rules with a contrasting behavior from the ones detected with absolute or logarithmic values. For instance, a rule that is characterized by a better performance than overall for the latter measures, can display higher errors than expected with residual errors. Some illustrative examples comparing EDRs with EDPs for multiple datasets were presented, utilizing logarithmic error values. These examples comprise cases that would not be easily detected using EDPs, situations that reduce drastically the number of plots to evaluate, subgroups with more than three conditions and, consequently, would not be detected with EDPs, among others.

Chapter 5

Case Study

In this chapter, a comprehensive study simulating a real problem of model selection is performed, taking advantage of the proposed tools to weight the advantages and disadvantages of each model. First, the motivation behind the problem is presented, followed by the characterization of the dataset. Then, the methodology applied is also outlined, namely the data discretization process, error calculation and utilized models. Next, a comparison between the results of all models is performed. Initially, this comparison is executed using scalar metrics. Then, with the use of graphical methods. Both procedures evaluate the results globally. After that, the graphical approach is extended, with the usage of EDRs, leading to a regional analysis. This encompasses a general overview of the rules, the analysis of performance tables, counter factual examples and individual graphical examination of subgroups shared by one or more predictive models. Finally, an overall examination of the results is accomplished, culminating in advocating or rejecting factors about the models.

5.1 Motivation

The main motivation behind this case study consisted of applying the develop tools to a relatively common ML problem, i.e., a problem that could be tackled without much domain knowledge. Thus, by selecting a dataset with a clear and understandable context, this was assured.

A frequent problem is the process of application to graduate programs. This process requires meticulous preparation, both in terms of the profiles of the students themselves and in the choice of relevant institutions. Consequently, it is common for students to have difficulties in selecting only a group of institutions, often without being aware of how the minimum requirements of the latter compare to their profiles. Therefore, many students fail admissions, wasting time and resources.

That said, the goal of this study was not to produce the most accurate model, but to analyze how different models compare in terms of error values for certain characteristics. For instance, by selecting a model that is globally better, some data combinations may be penalized with a higher range of error values, leading to less robust predictions. Scenarios like these are important for problems as this one, allowing users to have more information and knowledge about the decisions generated by the predictive models.

Additionally, due to the increase in understanding, end users might select models that were previously dismissed for certain data combinations, in order to compute sturdy results.

5.2 Dataset

The dataset used was originally introduced by Acharya et al. [101] and consists of an Indian perspective of application to Masters programs. Moreover, the set is composed by 500 instances and 7 features, consisting of important parameters of candidate selection. These include scores of exams such as Graduate Record Examination (GRE) or the Test of English as a Foreign Language (TOEFL), if the student has research experience, its undergraduate Grade Point Average (GPA), specifically in the form of Cumulative GPA (CGPA), among others. Table 3 depicts the statistical summary of the numerical features and target variable (*Chance.of.Admit*) of the dataset. Here, one can see that, for instance, the university ratings, the statement of purpose (SOP) and the letter of recommendation (LOR) have values between 1 and 5, in the form of a strength scale. Moreover, due to its categorical nature, feature *Research* is not depicted on the table. However, it is composed by 220 instances with no experience and 280 with, i.e., with values 0 and 1, respectively.

Table 3: Statistical summary of numerical variables of the case study dataset.

	GRE.Score	**TOEFL.Score**	**University.Rating**	**SOP**	**LOR**	**CGPA**	**Chance.of.Admit**
Minimum	290.0	92.0	1.000	1.000	1.000	6.800	0.3400
Q1	308.0	103.0	2.000	2.500	3.000	8.127	0.6300
Median	317.0	107.0	3.000	3.500	3.500	8.560	0.7200
Mean	316.5	107.2	3.114	3.374	3.484	8.576	0.7217
Q3	325.0	112.0	4.000	4.000	4.000	9.040	0.8200
Maximum	340.0	120.0	5.000	5.000	5.000	9.920	0.9700

Additionally, correlation plots between predictors and graphical representations of the distributions of each variable are presented in Appendix A, specifically in Figures 64 and 65.

5.3 Methodology

The process applied to the case study data was similar to the procedure described in Sections 4.3 and 4.4, in order to discretize numeric features and calculate the error values. In other words, numeric features were discretized using quartiles and the error values were calculated using a 10 fold CV method and a logarithmic measure to produce more pleasant graphical representations. Moreover, the parameters of the models used can be, once again, seen in Table 2. Furthermore, the process of rule discovery used CAREN, with minimum support of 5%. Lastly, the pruning filter value used was 1×10^{-14}. This is an improvement filter, meaning that the *p-value* of a more specific rule has to be at least 1×10^{-14} smaller than the *p-value* of its parent rule to be considered.

5.4 Model Comparison

The following section consists of a comprehensive analysis of the performance of models applied to the case study dataset. First, the models are compared using scalar metrics, as the RMSE and MAE. Second, in order to extract more information about the performance of each model, graphical visualizations of predicted values and errors, namely logarithmic and residual, are produced. Then, a comparison using EDRs is performed. This comparison encompasses an overview of all the different rules detected, globally and by each model, the study of performance tables for every predictor, some counter-factual examples and graphical visualizations. The latter comprise EDRs detected on all and only a few models. Lastly, in order to explain how the use of rules can be beneficial in performance analysis, an overview of some pros and cons of the models is executed.

5.4.1 Comparison using Scalar Metrics

The first step, and the most common, is to compare the various models based on a global scalar metric. In this case, two metrics were used, RMSE and MAE, to allow a more comprehensive comparison. As seen in Table 4, the models are defined by very similar values. Nonetheless, some interesting situations can be identified. For instance, the SVM model has slightly smaller values than the remaining models concerning both metrics. Consequently, and only regarding the available metrics, can be seen as the best model of the four. Contrarily, the ANN is defined by the highest metric values and should have the worse performance. Moreover, the GBM and the RF models have relatively similar values. However, the metrics for the RF model are slightly higher, being more notorious by comparing the MAE values for both models.

Table 4: RMSE and MAE values for the different models applied to the case study dataset.

Model	RMSE	MAE
Artificial Neural Network	0.06323443	0.04614278
Gradient Boosting Machine	0.06264165	0.04349683
Random Forest	0.06267704	0.04432247
Support Vector Machine	0.06199665	0.04273785

Thus, by only looking at these scalar metrics, the SVM appears to be the model that performs better for this dataset, followed by the GBM and RF models. Differently, the ANN should be the worst model overall. Nevertheless, the metrics are considerably similar, making this analysis not enough to confidently select a model over another.

5.4.2 Comparison using Global Graphical Metrics

In terms of a global graphical analysis, there are two important comparisons one can study: how the distribution of predicted values directly compares to the real values, and the errors of each model, i.e, by comparing their performances.

Regarding the first, Figure 46 depicts this study. It is clear that all the models fail in identifying smaller target values, as seen through their minimum values compared to the real distribution. Moreover, the ANN has the lowest minimum and the highest maximum of all models, making it the closest to the real distribution concerning these metrics. Additionally, the SVM is the second closest model to the real data in regard to the same points. Furthermore, by analyzing one model at the time, it is possible to see that the ANN also has the closest median to the real values and the IQR is slightly smaller than reality. In regard to the GBM, the IQR appears to be closest to the real values. However, its whiskers are considerably distant from reality. The RF model has the closest Q1 value to the real values and the SVM the nearest Q3, but also the farthest Q1 in comparison to the original data. Therefore, the ANN appears to have produced the closest results to the real data distribution. The GBM and RF models also produced similar distributions, but show a tendency to have higher errors regarding extremes, i.e., have greater differences concerning the minimum and maximum distribution values. Lastly, although the SVM is characterized by very accurate cutting points of the predicted distribution, its difference regarding the Q1 is considerable. Moreover, allied to its marginally high median, it is possible that the model has a tendency to predict higher values than reality.

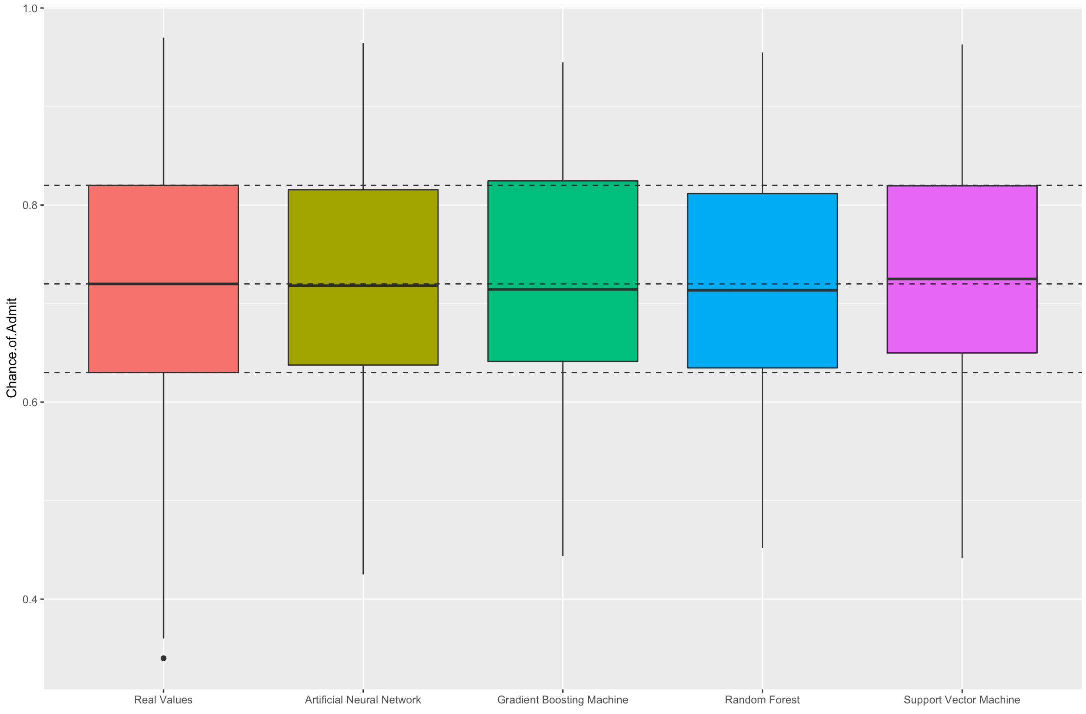


Figure 46: Graphical representation of the predicted values of the models (cf. Table 2) applied to the case study dataset and the real values.

After comparing the predicted values with real ones, it is also important to analyze how the models compare among themselves in terms of actual errors. As mentioned before, for visualization reasons, we opted for the use of logarithmic errors. Figure 47 consists of a comparison of the distributions of logarithmic errors for all models in study. Some interesting aspects can be seen in the fact that the GBM is defined by the lowest Q1, followed by the SVM, RF and, lastly, the ANN. The GBM also has the lowest median of the models. The RF and SVM succeed to it with very close values among them and, once again, the ANN has the highest value for this cutting point. Moreover, the SVM is defined by the lowest Q3 and maximum values. It is important to notice that the GBM has the second lowest Q3 and the second highest maximum value. Additionally, the RF model has the highest values for Q3 and maximum cutting points. Furthermore, all models are characterized by having some outliers, most of which are below 0.2. To summarize, the GBM and the SVM appear to be the better models. Nonetheless, the latter has a smaller IQR, with similar lower bounds, meaning that it has a tendency for lower error values. Moreover, these models have the most distant outliers of the four models, i.e., higher errors, and the outliers of the GBM are less dispersed in comparison to the SVM.

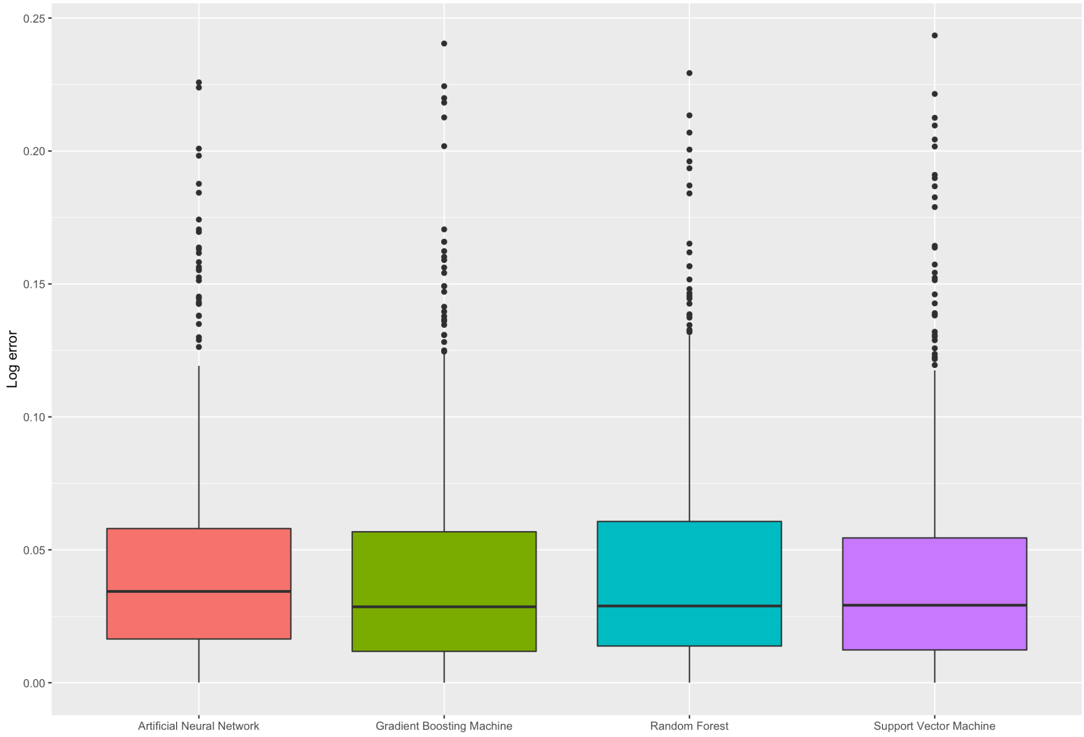


Figure 47: Graphical representation of the overall performance of the models (cf. Table 2) applied to the case study dataset.

As sometimes models may tend to over or under predict, the analysis of residual errors is also an important step of the process, as these depict the actual difference, positive or negative, between predictions and real values. By observing Figure 48, it is apparent that the SVM errors are centered around 0, as its median is approximately 0. Additionally, the remaining models are centered around positive values, meaning that the real values are higher than the predicted ones. There are also many negative outliers, probably due to the models predicting some considerably higher values than supposed to. Moreover, the ANN has the largest IQR of the four models, i.e., a larger variety of errors with considerable density. Similarly, the SVM also has a large IQR and difference between maximum and minimum, possibly leading to higher errors, as seen before, despite seeming to be the better option due being centered around 0. Regarding the remaining models these are more concise in terms of error values, even though they tend to under predict the results.

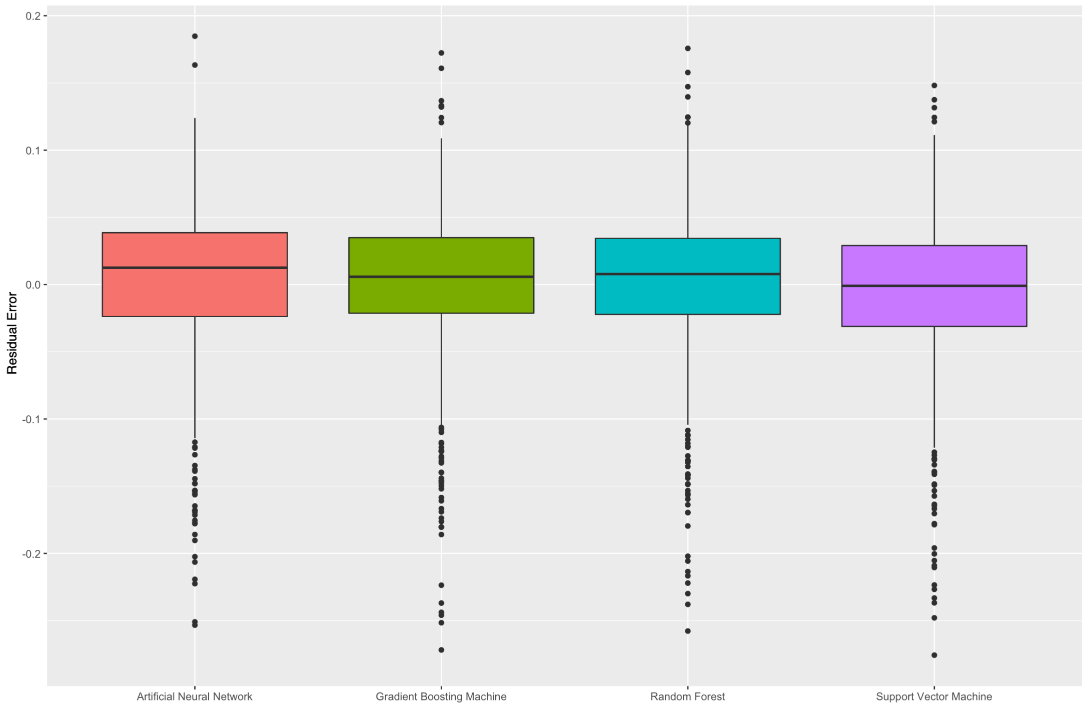


Figure 48: Graphical representation of the overall performance of the models (cf. Table 2) applied to the case study dataset, using residual errors.

5.4.3 Comparison using Rules

After the global scalar and graphical analysis, it is now possible to perform a rule based study of the models. This allows for the identification of specific situations where a model behaves differently from its global performance. Table 5 depicts the number of rules detected for each model and for number of features. It is clear that the ANN is the model with least amount of uncovered rules, followed by the GBM with one more rule detected. Moreover, the rules of these models are characterized for being simpler, containing only one or two variables. Inversely, the RF and SVM models have 49 and 48 EDRs, respectively, including some with three and four predictors. Moreover, the SVM is the model with the most amount of EDRs with more than two variables. It is also important to notice that most of the detected rules have two variables, followed by single-feature rules, meaning that there are few cases with two or four predictors.

Table 5: Number of EDRs discovered per model and number of interactions for the case study dataset.

Model	**One Feature**	**Two Features**	**Three Features**	**Four Features**	**Total**
Artificial Neural Network	15	17	0	0	32
Gradient Boosting Machine	16	17	0	0	33
Random Forest	18	27	3	1	49
Support Vector Machine	16	26	5	1	48
Total	65	87	8	2	

Continuing the primary stages of the study, Table 6 consists of a detailed analysis of all the 67 different subgroups detected. Of these, 19 were discovered on all models, 10 on three, 18 on two and 20 on only one model. Some examples of rules discovered on all models are:

- $CGPA = [8.96 - 9.53]$;
- $GRE.Score = [297 - 308], LOR = [2.5 - 3.5]$;
- $LOR = [4 - 4.5], Research = 1$;
- $TOEFL.Score = [111 - 117]$;
- $University.Rating = [2 - 2], GRE.Score = [309 - 323]$.

This means that these subgroups have a different enough behavior on all models to be classified as interesting cases to analyze. Moreover, it is also important to see what rules are detected on only a subset of models. For example, the subgroup $CGPA = [7.64 - 8.27], SOP = [2.5 - 3.5]$ is only detected on the GBM, while the CGPA condition alone is consistent on all models. Contrarily, $SOP = [2.5 - 3.5]$ is only highlighted on the GBM and RF models. Another example is the subgroup $CGPA = [8.96 - 9.53], GRE.Score = [324 - 335]$, only uncovered for the RF and SVM models. However, the individual conditions, separately, are associated with the four regression models, meaning that for these two, the behavior differs enough from their individual counterparts to be pointed out. Further, the ANN is the only model with the subgroup $Research = 0, LOR = [2.5 - 3.5]$, despite its base variant for *Research* only being highlighted on the RF and SVM models, and the base condition for LOR on all models but the SVM. Another interesting example can be seen in $SOP = [2.5 - 3.5], Research = 1, LOR = [2.5 - 3.5], TOEFL.Score = [102 - 110]$, a subgroup that is only associated with the RF model. While $SOP = [2.5 - 3.5]$ appears on the GBM and the RF itself, $Research = 1$ is not individually detected on any model, but has some variants, similarly to $TOEFL.Score = [102 - 110]$. Additionally, $LOR = [2.5 - 3.5]$ appears on all models, except the SVM.

After knowing which subgroups have an interesting behavior for each model, it is crucial to understand how these actually behave in comparison to the global distributions of their models. The use of performance tables allows this understanding, as they produce a general overview of the performance of each subgroup. Moreover, these comprise a powerful tool to quickly understand how many subgroups have better or worse error values than expected.

Starting with the ANN model, in Figure 49, some aspects can be easily seen. For instance, this model has 18 subgroups with a better performance than expected, in terms of Q1, median and Q3 values, depicted in green. Contrarily, it has 13 subgroups with higher errors than anticipated. Furthermore, there is a group that is characterized for having a higher Q1 value, but lower median, Q3 and maximum values. It is also interesting to notice that there are 5 subsets with minimum errors as low as the whole data and 10 with maximums as high. Thus, of the 32 subgroups detected, the majority, $\frac{19}{32} = 59.4$%, have better performance than expected and $\frac{13}{32} = 40.6$% worse. Moreover, as these tables allow the analysis of specific behavior, one can also see that, e.g., *University.Rating* with value of 5 always produces better results, while the value of 2 for the same attribute is characterized for having worse performance for its detected groups. Other examples can be seen in $TOEFL = [95 - 101]$, characterized by higher errors than overall, or instances with research experience, defined by higher accuracy in the prediction.

Table 6: EDRs discovered per model for the case study dataset. *X* indicates that the subgroup was present for the model.

Subgroup	Artificial Neural Network	Gradient Boosting Machine	Random Forest	Support Vector Machine
CGPA=[6.8-7.6]				X
CGPA=[7.64-8.27]	X	X	X	X
CGPA=[7.64-8.27], LOR=[2.5-3.5]	X	X		X
CGPA=[7.64-8.27], SOP=[2.5-3.5]		X		
CGPA=[8.28-8.95], Research=0, TOEFL.Score=[102-110]				X
CGPA=[8.96-9.53]	X	X	X	X
CGPA=[8.96-9.53], GRE.Score=[324-335]			X	X
CGPA=[8.96-9.53], Research=1			X	X
CGPA=[8.96-9.53], SOP=[4-4.5]			X	X
CGPA=[8.96-9.53], SOP=[4-4.5], Research=1				X
CGPA=[9.54-9.92]	X	X	X	X
CGPA=[9.54-9.92], Research=1			X	
GRE.Score=[297-308]	X	X	X	X
GRE.Score=[297-308], LOR=[2.5-3.5]	X	X	X	X
GRE.Score=[324-335]	X	X	X	X
GRE.Score=[324-335], Research=1	X	X	X	X
GRE.Score=[324-335], SOP=[4-4.5]		X	X	X
LOR=[1.5-2], SOP=[2.5-3.5]		X		
LOR=[2.5-3.5]	X	X	X	
LOR=[2.5-3.5], Research=1, TOEFL.Score=[102-110]			X	
LOR=[2.5-3.5], TOEFL.Score=[102-110]			X	X
LOR=[4-4.5]	X	X		
LOR=[4-4.5], Research=1	X	X	X	X
LOR=[5-5]	X	X	X	X
LOR=[5-5], Research=1	X		X	X
LOR=[5-5], University.Rating=[5-5]	X			
Research=0			X	X
Research=0, LOR=[2.5-3.5]	X			
SOP=[1.5-2]	X		X	X
SOP=[1.5-2], LOR=[2.5-3.5]	X	X	X	X
SOP=[1.5-2], TOEFL.Score=[95-101]			X	X
SOP=[2.5-3.5]		X	X	
SOP=[2.5-3.5], LOR=[2.5-3.5]			X	
SOP=[2.5-3.5], Research=1, LOR=[2.5-3.5]			X	
SOP=[2.5-3.5], Research=1, LOR=[2.5-3.5], TOEFL.Score=[102-110]			X	
SOP=[4-4.5]		X	X	X
SOP=[4-4.5], LOR=[4-4.5]		X		X
SOP=[4-4.5], Research=1	X		X	X
SOP=[5-5]	X	X	X	X
SOP=[5-5], Research=1	X	X	X	X
TOEFL.Score=[111-117]	X	X	X	X
TOEFL.Score=[111-117], CGPA=[8.96-9.53]			X	X
TOEFL.Score=[111-117], Research=1	X		X	X
TOEFL.Score=[111-117], SOP=[4-4.5]		X	X	X
TOEFL.Score=[118-120]	X	X	X	X
TOEFL.Score=[118-120], Research=1	X		X	
TOEFL.Score=[95-101]	X	X	X	X
TOEFL.Score=[95-101], CGPA=[7.64-8.27]	X		X	
TOEFL.Score=[95-101], GRE.Score=[297-308]	X			
TOEFL.Score=[95-101], LOR=[2.5-3.5]		X		
TOEFL.Score=[95-101], Research=0		X		X
University.Rating=[2-2]	X	X	X	X
University.Rating=[2-2], GRE.Score=[309-323]	X	X	X	X
University.Rating=[2-2], GRE.Score=[309-323], SOP=[2.5-3.5], TOEFL.Score=[102-110]				X
University.Rating=[2-2], GRE.Score=[309-323], TOEFL.Score=[102-110]			X	X
University.Rating=[2-2], LOR=[2.5-3.5]			X	
University.Rating=[2-2], Research=1				X
University.Rating=[2-2], Research=1, LOR=[2.5-3.5]				X
University.Rating=[2-2], SOP=[2.5-3.5]		X	X	
University.Rating=[3-3], LOR=[2.5-3.5]			X	X
University.Rating=[3-3], Research=0, LOR=[2.5-3.5]				X
University.Rating=[4-4]			X	
University.Rating=[4-4], GRE.Score=[324-335]	X			X
University.Rating=[4-4], Research=1		X	X	X
University.Rating=[5-5]	X	X	X	X
University.Rating=[5-5], Research=1	X	X	X	X
University.Rating=[5-5], TOEFL.Score=[111-117]			X	X

Figure 50 contains the information about the GBM model. Here, it is clear that of the 33 rules, 16 are defined with smaller errors and 15 with higher, concerning the middle quartiles, i.e., Q1, median and Q3. Additionally, and similarly to the previous model, there are two subgroups with a higher lower bound

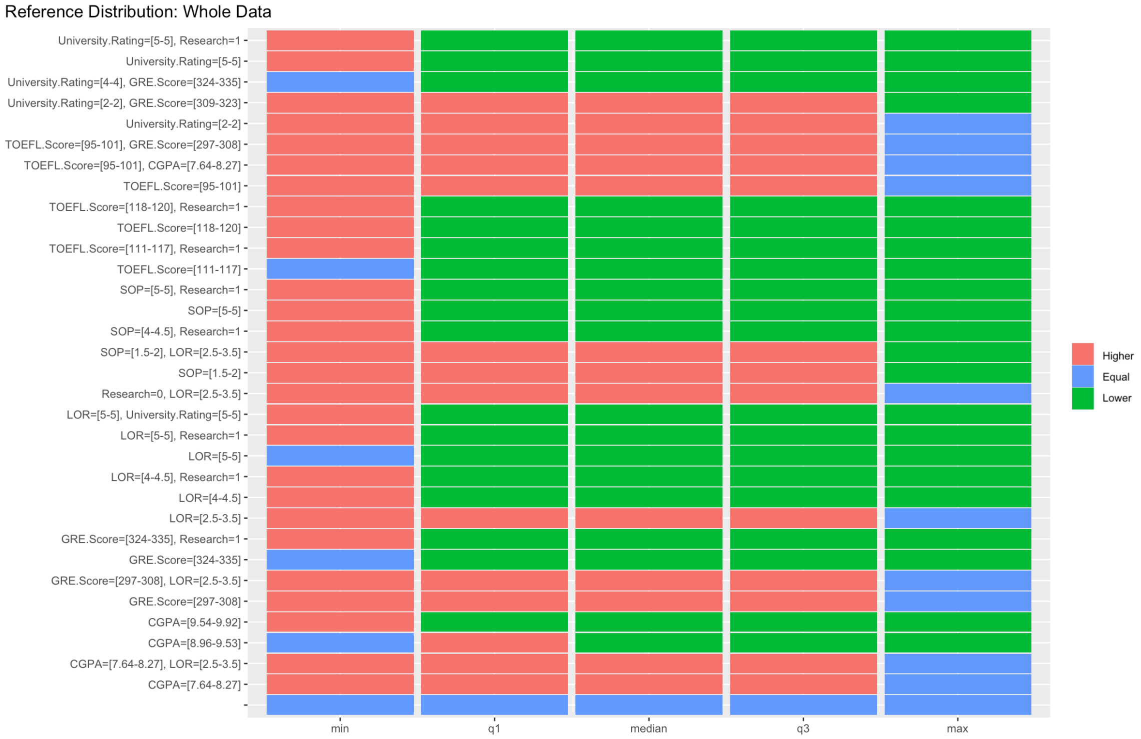


Figure 49: Global performance table from the case study dataset, trained with an ANN model (cf. Table 2).

error value, but smaller median and upper bound. Assuming these as better performance overall, the GBM is defined by $\frac{18}{33} = 54.5\%$ of better performance subgroups and $\frac{15}{33} = 45.5\%$ worse. Furthermore, it is intriguing to see that, while the ANN CGPA with values $[8.96 - 9.53]$ was defined by a higher Q1, despite the remaining cutting points being better than he overall model, for this model, it is CGPA with values $[9.54 - 9.92]$ that follows this behavior.

The performance table for the RF model is presented in Figure 51, containing 27 subgroups with lower errors overall and 22 with higher, i.e., $\frac{27}{49} = 51.1\%$ and $\frac{22}{49} = 44.9\%$, respectively. Moreover, there are 13 sets of data with minimum errors as low as the whole distribution and 6 with maximum values as high as the global data. It is also interesting to notice that this model does not have scenarios where a subgroup is simultaneously better and worse than the overall performance for some cutting points, as seen in the foretold examples. Furthermore, and contrarily to the examples before, this model has some interesting scenarios regarding the combination of conditions leading to opposing performances, e.g, variations of $Research = 1$.

Lastly, there is the SVM model, portrayed in Figure 52. By globally analyzing the EDRs, one may see that the model is characterized by $\frac{27}{48} = 56.25\%$ of better performing subgroups and $\frac{21}{48} = 43.75\%$ of worse. Additionally, the number of subgroups with equal minimum and maximum values to the global distribution are 10 and 5, respectively. Similarly to the previous model, this also contains some counterfactual examples for $Research = 1$.

Then, as there are some subgroups with contradictory performance, the discovery of some counterfactual examples is an important complement to the analysis of the tables. However, many examples only

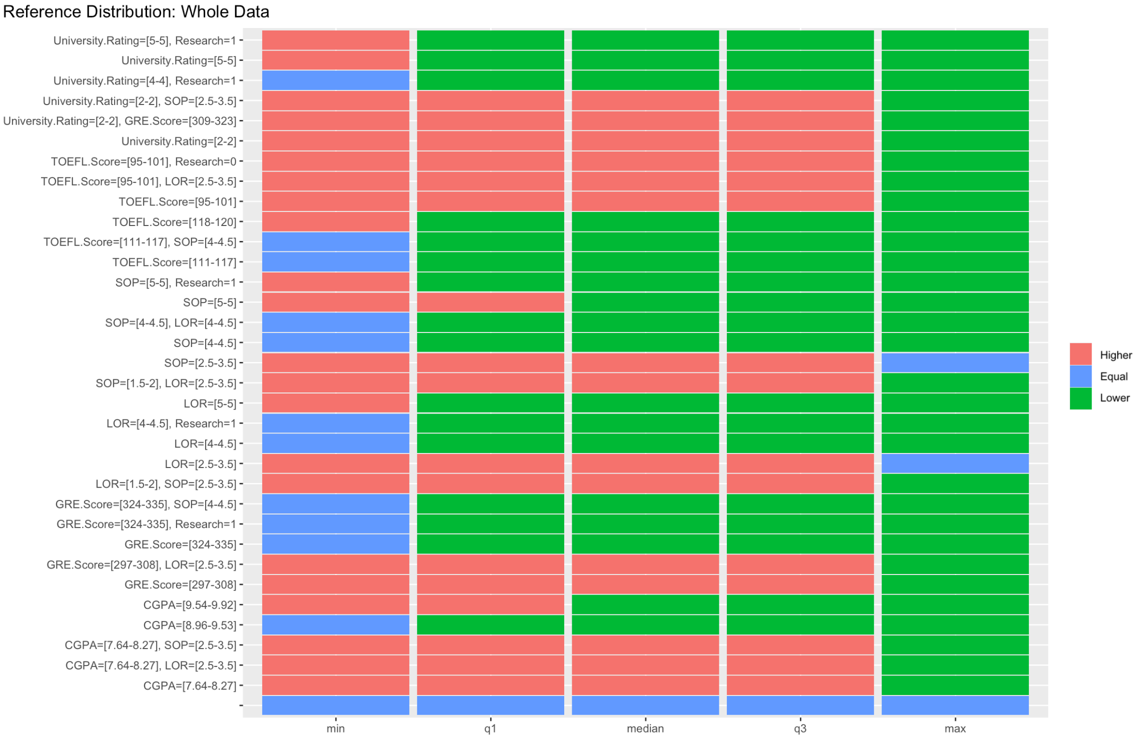


Figure 50: Global performance table from the case study dataset, trained with a GBM model (cf. Table 2).

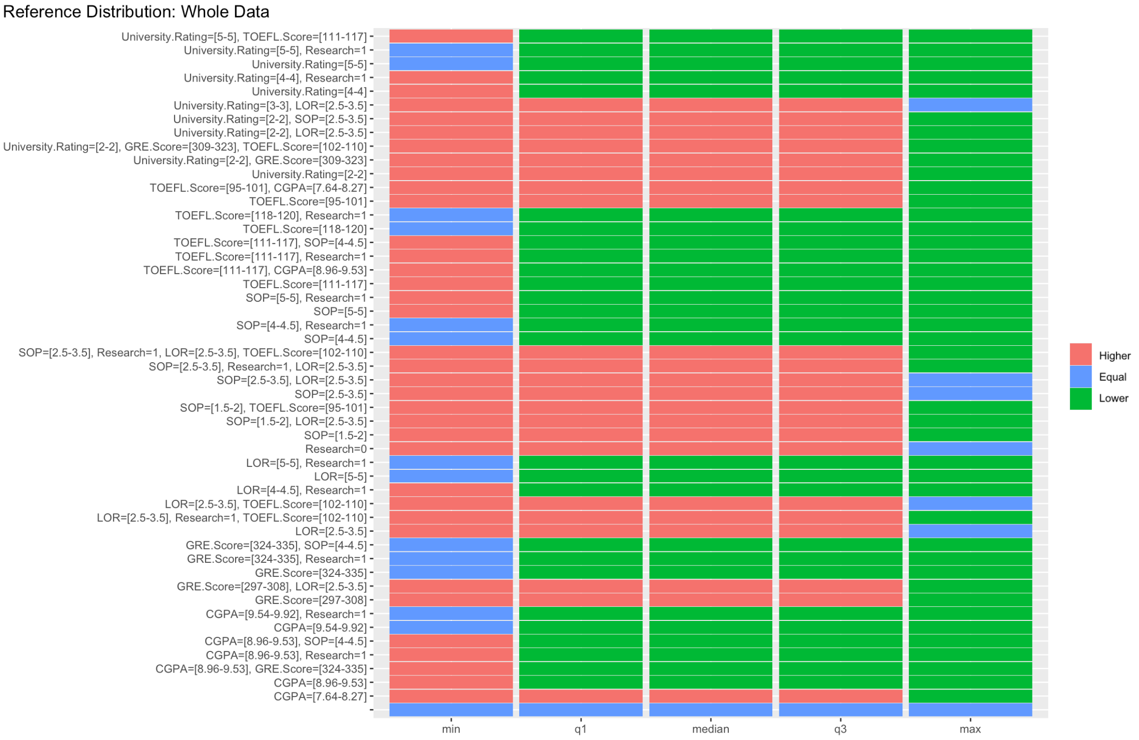


Figure 51: Global performance table from the case study dataset, trained with a RF model (cf. Table 2).

differ on the minimum or maximum values, as in Listing 1. Here, $LOR = [5-5]$ has the minimum error value equal to the global distribution. Nevertheless, by merging the condition with $University.Rating = [5-5]$, the minimum error is higher, being expected to obtain slightly higher error values.

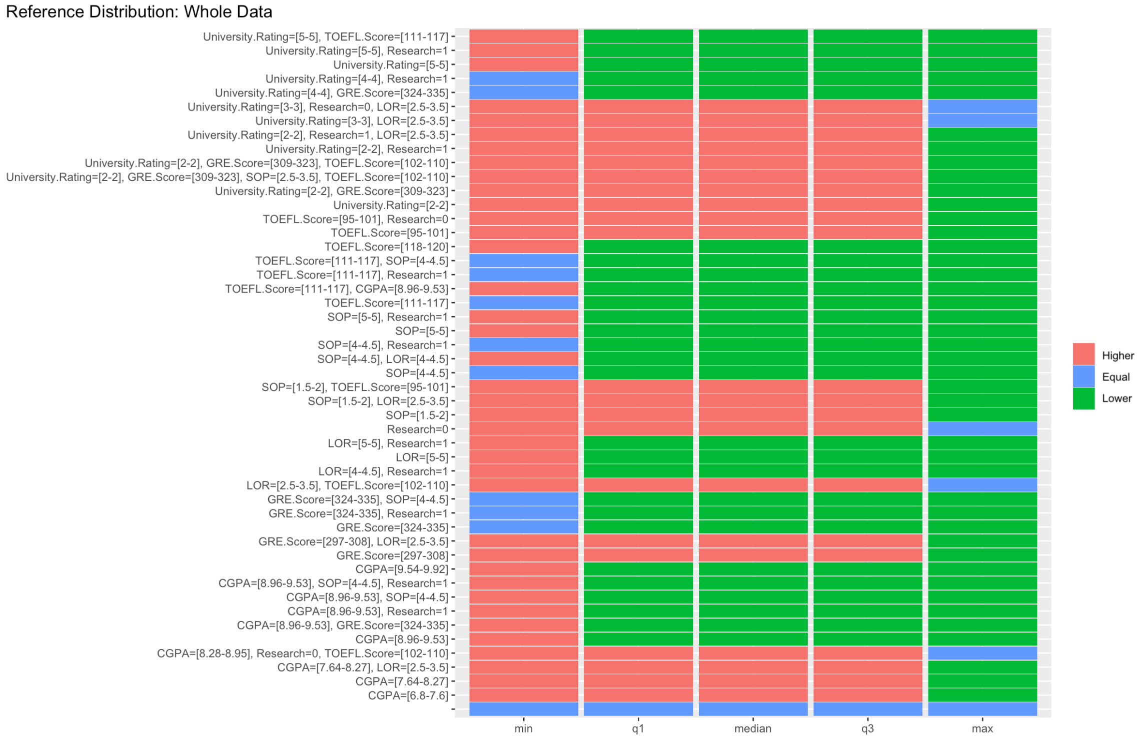


Figure 52: Global performance table from the case study dataset, trained with a SVM model (cf. Table 2).

Listing 1: Counter-factual examples for subgroup LOR = 5, University.Rating = 5 of the case study dataset, trained with an ANN model (cf. Table 2)

```
>> LOR=[5-5], University.Rating=[5-5]
>> Higher Lower Lower Lower Lower

           > LOR=[5-5]
           Equal Lower Lower Lower Lower
```

A more compelling example is available in Listing 2, from the SVM model. For instance, $University.Rating = [2-2], Research = 1$, or $University.Rating = [2-2], Research = 1, LOR = [2.5-3.5]$ have a worse performance overall regarding the whole data, in contrast to $University.Rating = [5-5], Research = 1$, that displays lower errors than expected. Furthermore, some less extreme examples can be seen in $TOEFL.Score = [111-117], Research = 1$ or $GRE.Score = [324-335], Research = 1$, where the minimum error values are lower, i.e., equal to the whole data. In contrast to $University.Rating = [5-5], Research = 1$ that is characterized for having a higher minimum error value than the global data.

Listing 2: Counter-factual examples for subgroup University.Rating = 5, Research = 1 of the case study dataset, trained with a SVM model (cf. Table 2)

```
>> University.Rating=[5-5], Research=1
>> Higher Lower Lower Lower Lower
```

```
        > TOEFL.Score=[111−117], Research=1
        Equal Lower Lower Lower Lower

        > GRE.Score=[324−335], Research=1
        Equal Lower Lower Lower Lower

        > SOP=[4−4.5], Research=1
        Equal Lower Lower Lower Lower

        > University.Rating=[2−2], Research=1, LOR=[2.5−3.5]
        Higher Higher Higher Higher Lower

        > University.Rating=[2−2], Research=1
        Higher Higher Higher Higher Lower

        > University.Rating=[4−4], Research=1
        Equal Lower Lower Lower Lower
```

Another interesting case is depicted in Listing 3, from the RF model. Similarly to the previous examples, we can examine some less impactful patterns, such as $University.Rating = [5{-}5], Research = 1$, or $CGPA = [9.54{-}9.92], Research = 1$, that display lower minimum error values than the compared subgroup. Additionally, it is visible that, for instance, $SOP = [2.5 - 3.5], Research = 1, LOR = [2.5 - 3.5], TOEFL.Score = [102 - 110]$, or $SOP = [2.5 - 3.5], Research = 1, LOR = [2.5 - 3.5]$ are characterized for having a worse performance in terms of Q1, median and Q3 in regard to $SOP = [5 - 5], Research = 1$ and the global distribution of errors.

Listing 3: Counter-factual examples for subgroup SOP = 5, Research = 1 of the case study dataset, trained with a RF model (cf. Table 2)

```
>> SOP=[5−5], Research=1
>> Higher Lower Lower Lower Lower

        > University.Rating=[5−5], Research=1
        Equal Lower Lower Lower Lower

        > GRE.Score=[324−335], Research=1
        Equal Lower Lower Lower Lower

        > CGPA=[9.54−9.92], Research=1
        Equal Lower Lower Lower Lower

        > TOEFL.Score=[118−120], Research=1
        Equal Lower Lower Lower Lower

        > SOP=[4−4.5], Research=1
        Equal Lower Lower Lower Lower
```

```
> LOR=[5 — 5], Research=1
Equal Lower Lower Lower Lower

> SOP=[2.5 — 3.5], Research=1, LOR=[2.5 — 3.5], TOEFL.Score=[102 — 110]
Higher Higher Higher Higher Lower

> LOR=[2.5 — 3.5], Research=1, TOEFL.Score=[102 — 110]
Higher Higher Higher Higher Lower

> SOP=[2.5 — 3.5], Research=1, LOR=[2.5 — 3.5]
Higher Higher Higher Higher Lower
```

Until this point, only pointwise comparisons were performed. Nonetheless, it is also relevant to fully compare distributions of error values. We start by considering some subgroups that are present on all models. Figure 53 depicts the subgroup defined by $CGPA = [7.64 - 8.27]$, composed by 145 instances, exactly 29% of all examples. Moreover, the subgroup has higher error values than expected for the four models. By actually comparing the models themselves, it is visible that the ANN and SVM have the errors centered around a smaller range of values. Nonetheless, the second has a considerable amount of outliers in comparison to the remaining models. Moreover, its Q1 value is relatively high compared to the GBM. Contrarily to the latter, the distributions of the subgroups for the remaining models show resemblances in terms of shape, e.g., the median of the first is closer to the center on the subgroup and close to Q1 on the whole data. Furthermore, the SVM also has the closest median to the global distributions, being slightly higher than the global ANN model.

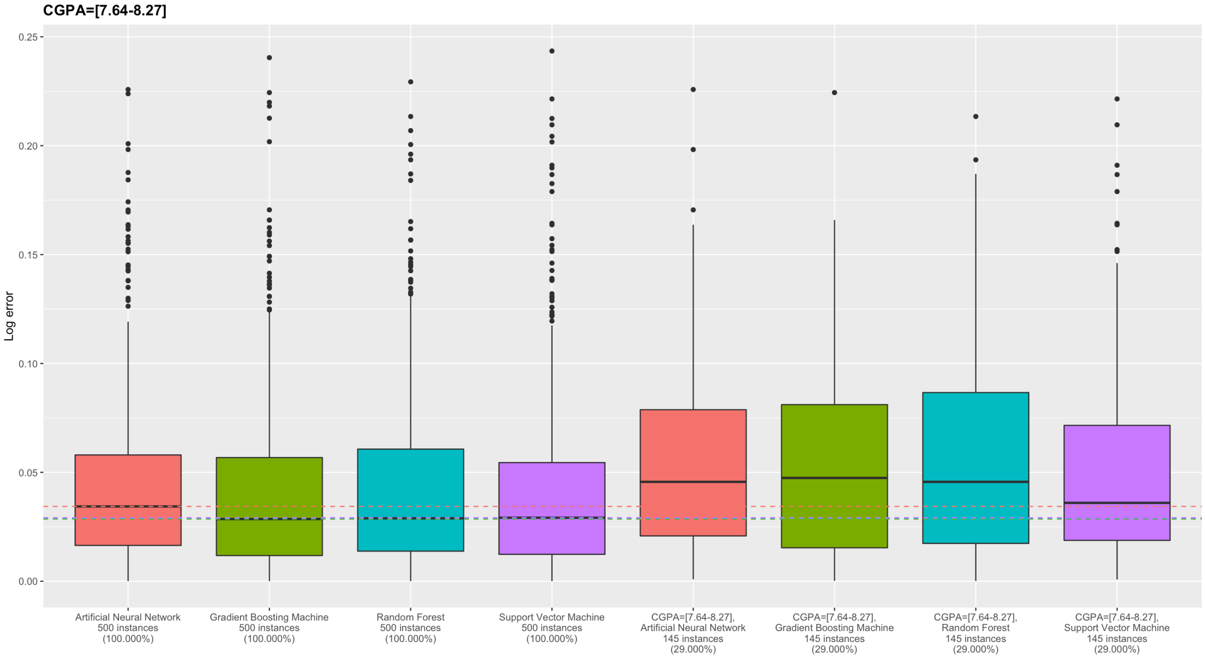


Figure 53: Boxplot EDRs from the case study dataset to analyze subgroup $CGPA = [7.64 - 8.27]$, trained with all models (cf. Table 2).

Another example can be seen in Figure 54, illustrating $GRE.Score = [297 - 308], LOR = [2.5 - 3.5]$. Once again, this subgroup performs worse than expected for all models. Concretely, some important aspects can be seen in the GBM, as this has the lowest Q1 value, or the RF, having the highest Q1, median and Q3 values of all models. Furthermore, the GBM has the most variety of values, i.e., highest IQR, leading to a wider scope of possible errors, from low to high values. The ANN is defined by the highest maximum and outlier of the models. Moreover, the latter and the SVM have, more or less, the same median value, being the smallest of the four. However, the SVM has a much lower Q1 and Q3, focusing the errors around the smallest values, compared to the rest of the models.

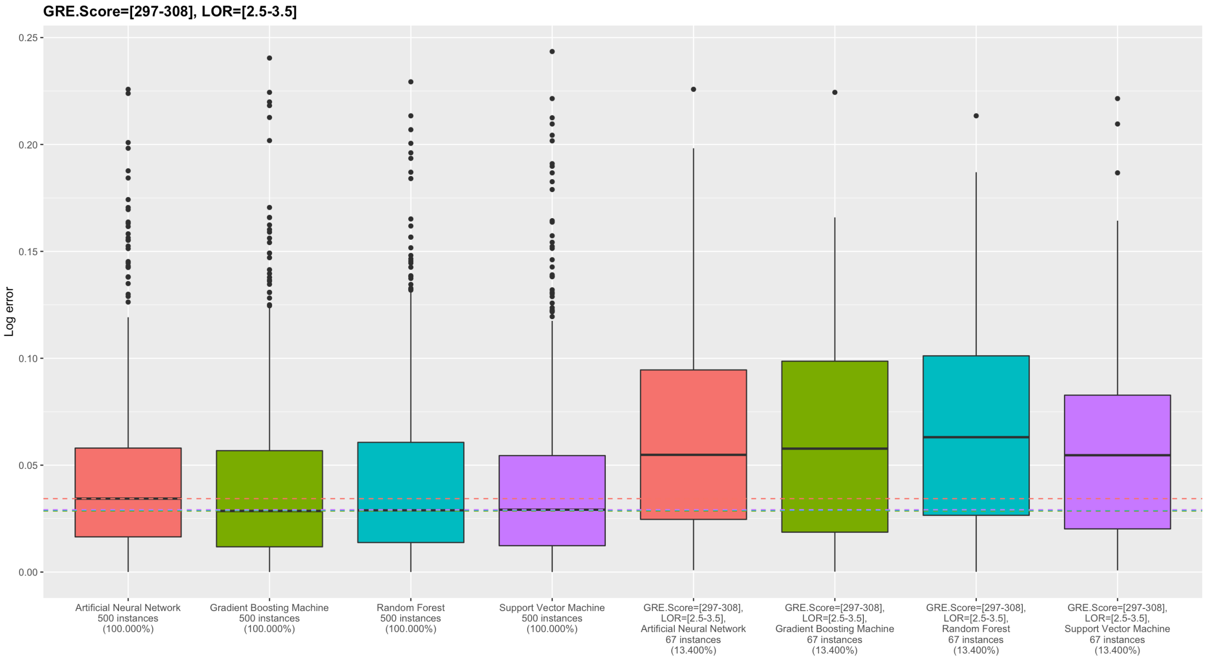


Figure 54: Boxplot EDRs from the case study dataset to analyze subgroup $GRE.Score = [297 - 308], LOR = [2.5 - 3.5]$, trained with all models (cf. Table 2).

Figure 55 contains a graphical representation of the performance of all models for subgroup $LOR = [4 - 4.5], Research = 1$. Oppositely to the former examples, this group is characterized for having smaller errors than the overall behavior of the models. For instance, the ANN is the worst model for this subgroup of the four, as it has the highest errors on all cutting points and the highest IQR. Moreover, the RF has the smallest Q3 and maximum values, followed by the GBM and SVM, respectively. Besides that, the medians of the subgroup for the RF and SVM are extremely close and the lowest of the four models. In short, the best models for this subgroup are the GBM and the SVM, as these have low values for Q1, and their median and Q3 values are similar to the RF model. However, even though the IQR of the SVM is higher than the one of the GBM, the first has a considerable smaller Q1 value, with relatively similar values of median and Q3, being the best choice of the two.

The performance of the subgroup defined by the restrictions $SOP = [1.5 - 2], LOR = [2.5 - 3.5]$ can be seen in Figure 56. Similarly to the example in Figure 54, that is also defined by the same condition for attribute LOR, this subgroup is characterized with worse performance than expected for all models. Nevertheless, this group of conditions has smaller errors than the previous one, indicating more accurate

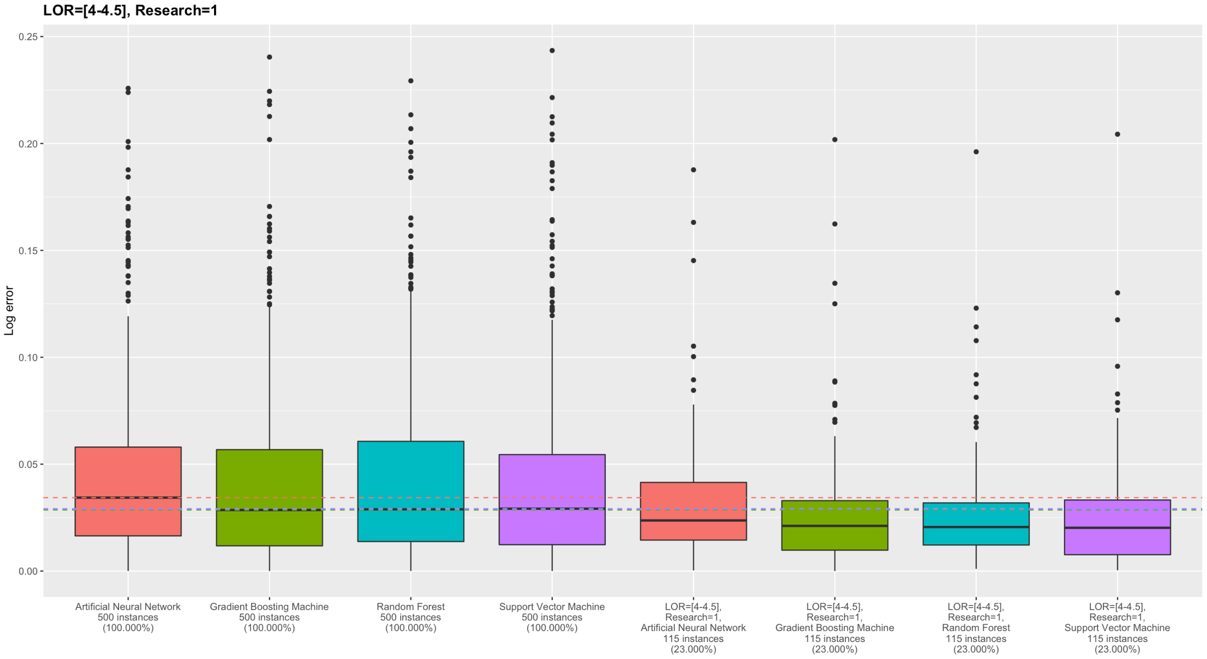


Figure 55: Boxplot EDRs from the case study dataset to analyze subgroup $LOR = [4 - 4.5], Research = 1$, trained with all models (cf. Table 2).

predictions when $SOP = [1.5 - 2]$ coexists with $LOR = [2.5 - 3.5]$, contrarily to $GRE.Score = [297-308]$. Additionally, a greater difference between the values of the Q1 is also seen, as the GBM widely outperforms the other models for this cutting point. However, the SVM model has the lowest median, Q3 and maximum values. Nonetheless, this model contains some outliers, contrarily to the remaining models. It is also interesting to notice that the Q1 of this model is higher than its global median, while, for the remaining models, this does not occur. Overall, the GBM seems capable to predict more accurate results, but the SVM has its errors more centered and concise, possibly being more reliable. Nevertheless, both models largely outperform the ANN and the RF for this subgroup.

One more illustrative case can be observed in Figure 57, concerning the subgroup $TOEFL.Score = [111 - 117]$. Once again, this is a subgroup with overall smaller errors than expected for all models, as in Figure 55. Coincidentally, the ANN model is not as good as the remaining models for this subgroup, as well. Nevertheless, its performance for the group still outperforms its global efficiency. The GBM and the RF exhibit, in some extent, similar values. However, the RF has lower values for its Q1 and Q3 and also the lowest observed maximum value. On the other hand, the GBM has the second lowest median for this subgroup, but has the highest outliers. Overall, the SVM may be seen as the better option in this case, as it has the lowest Q1, median and Q3 values, followed by the RF model. Nonetheless, the GBM also produces interesting results, yet, it tends for higher error values than the two foretold models.

It is also important to evaluate the behavior of subgroups that only appear in certain models, as these are characterized for differing performances, regarding both the global efficiency and that of the remaining subgroups. Figure 58 contains the error values of the subgroup $University.Rating = [2 - 2], Research = 1$, detected on the SVM. This group of instances is associated with higher errors than the overall model. In fact, the Q1 of the subgroup neighbors the global median and its median is

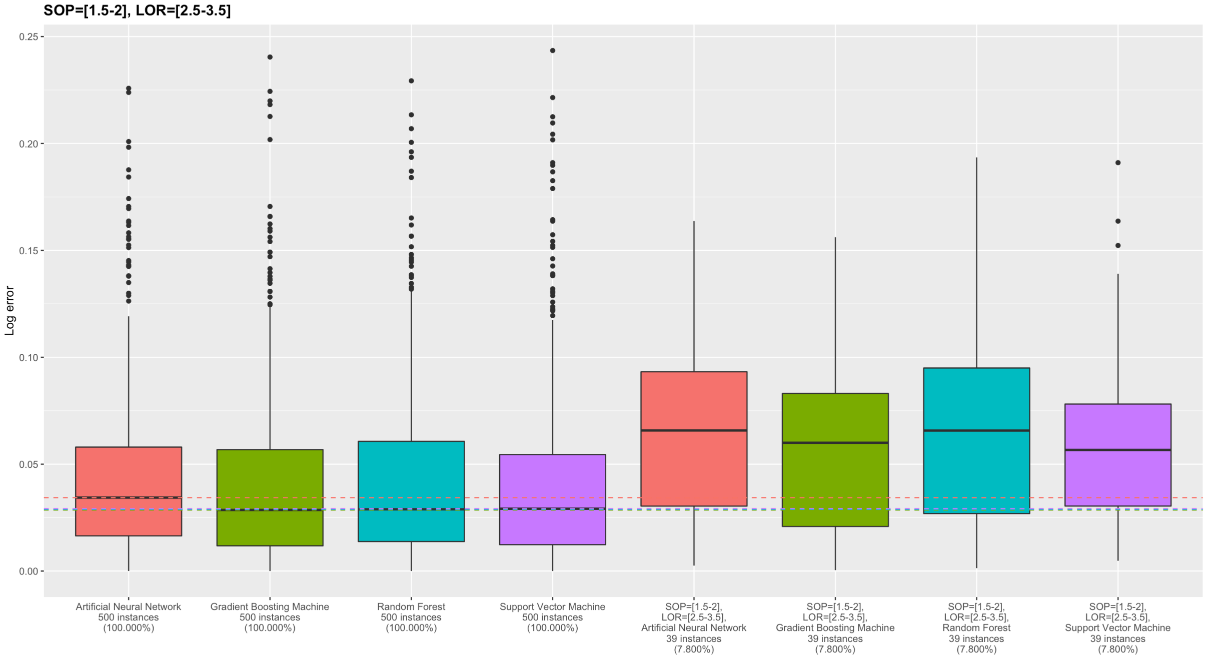


Figure 56: Boxplot EDRs from the case study dataset to analyze subgroup $SOP = [1.5 - 2], LOR = [2.5 - 3.5]$, trained with all models (cf. Table 2).

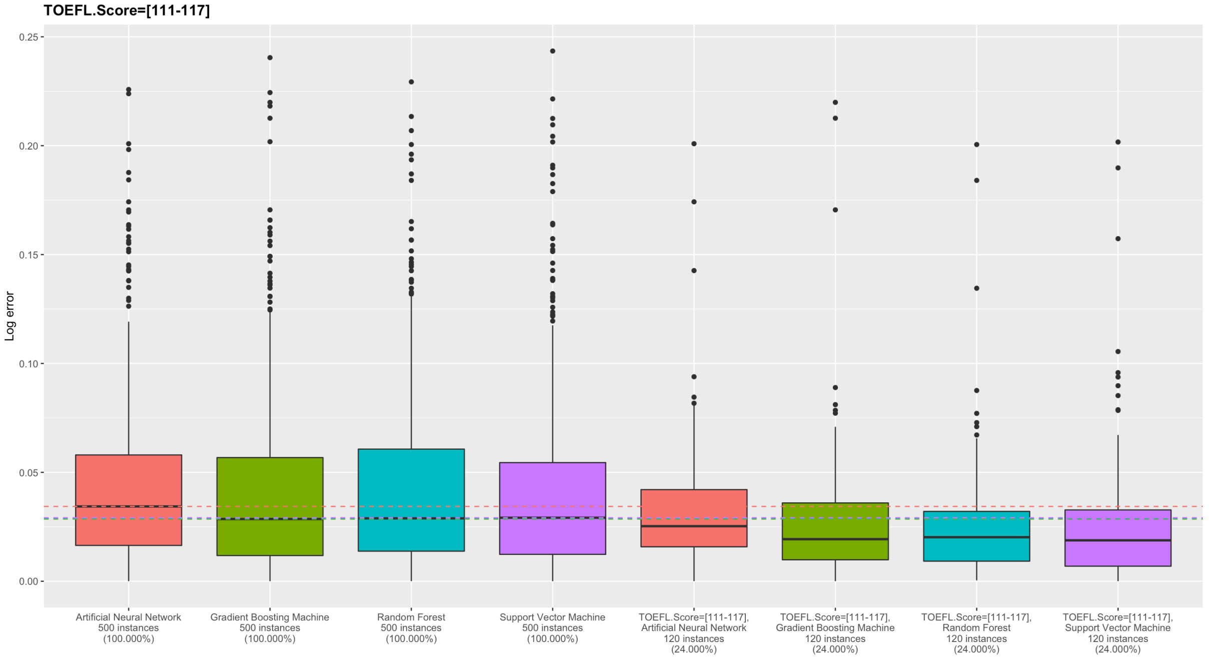
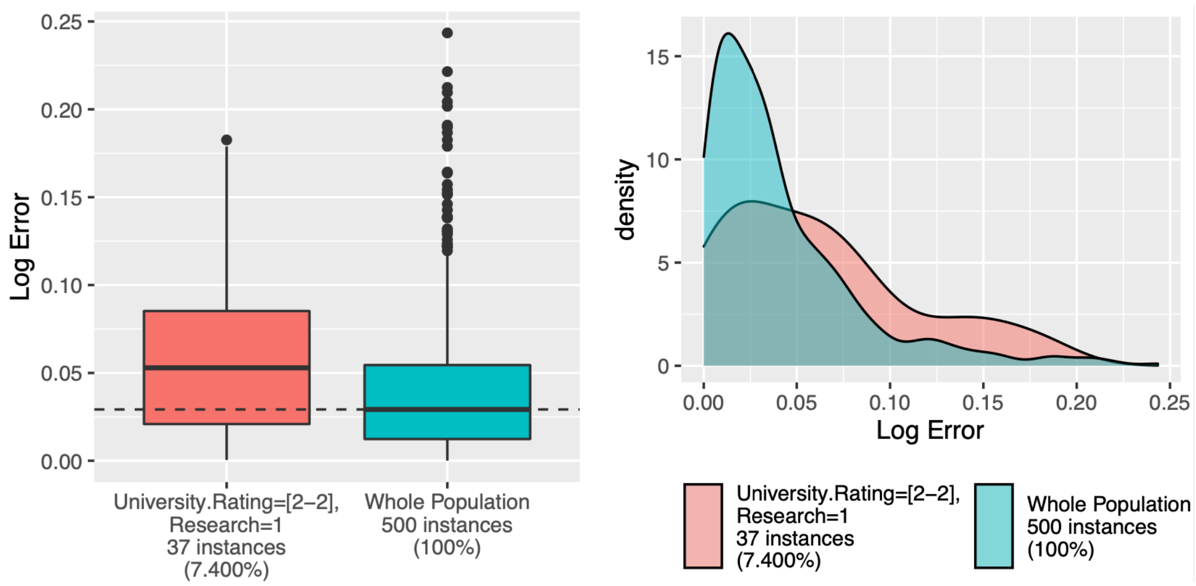


Figure 57: Boxplot EDRs from the case study dataset to analyze subgroup $TOEFL.Score = [111 - 117]$, trained with all models (cf. Table 2).

almost as high as the global Q3 value, as seen in Figure 58a. Moreover, the density plot, in Figure 58b, shows that the density of the subgroup not only has a much smaller peak than the global one, but is also more spread withing the same range, with more uniform density values. Contrarily, the global distribution drops rapidly after reaching its peak.

Another example can seen in the GBM model, specifically in the subgroup $TOEFL.Score = [95 -$



(a) Boxplot.

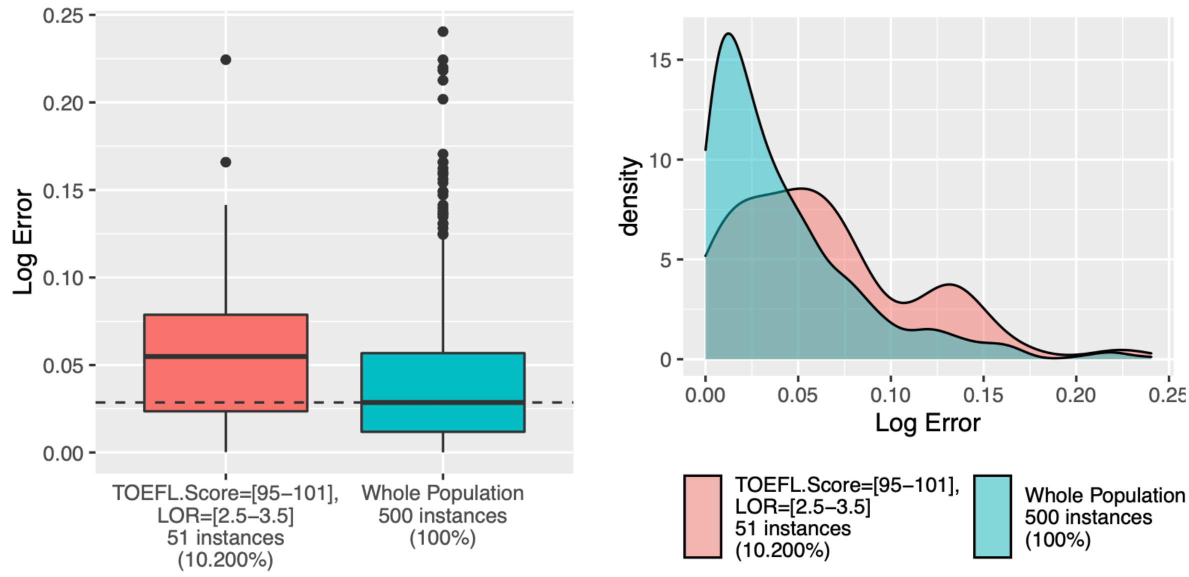
(b) Density plot.

Figure 58: EDR from the case study dataset to analyze subgroup $University.Rating = [2 - 2], Research = 1$, trained with a SVM model (cf. Table 2).

$101], LOR = [2.5 - 3.5]$, depicted in Figure 59. Similarly to the former example, this subgroup is characterized for having higher errors than expected. In fact, both have an identical behavior, as the Q1 of this subgroup is also similar to the global median and its median is almost as high as the global Q3. However, this subgroup has smaller values for the Q3 and maximum cutting points, than the former one. Regarding the density analysis, two peaks are clearly visible for the subgroup. A larger one around 0.05 and another at approximately 0.13. Differently, the global error values gradually descend after hitting the peak. Nonetheless, there seems to be a slight variation on the decline of the descent around the same error value as the second peak of the subgroup appears.

Figure 60 contains graphical representations for the EDR defined by $CGPA = [9.54 - 9.92]$, $Research = 1$. This was only detected on the RF model and is characterized by a range of smaller error values than expected. Furthermore, the subgroup is relatively small, being composed by only 29 instances, i.e., 5.8% of all items. Even though this is a subgroup with a low support, the combination of features was detected as having a better performance than expected. Another interesting aspect is the fact that $CGPA = [9.54 - 9.92]$ was detected on all models, being composed by 30 instances, but only the RF model detected its combination with $Research = 1$. This is possible due to a higher level of difference between the the distribution containing both restrictions and its singular counter-part, namely in terms of its *p-value* used in the pruning stage. Concerning the actual performance of the instances, the boxplot representation shows that the median of the subgroup is approximately equal to the global Q1 and its Q3 is considerably smaller than the global median. Furthermore, the density plot shows two major peaks below 0.05 and a very swift descent for the group, corroborating that this combination of restrictions is defined by extremely small error values.

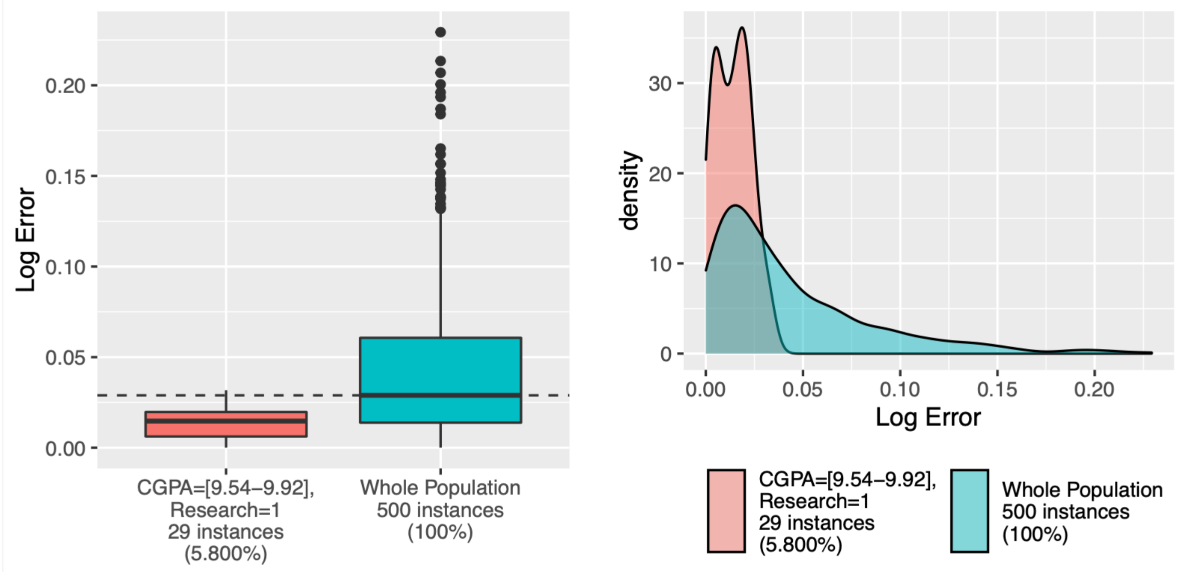
An additional interesting group only detected on a single model is the one defined by $CGPA =$



(a) Boxplot.

(b) Density plot.

Figure 59: EDR from the case study dataset to analyze subgroup $TOEFL.Score = [95 - 101], LOR = [2.5 - 3.5]$, trained with a GBM model (cf. Table 2).

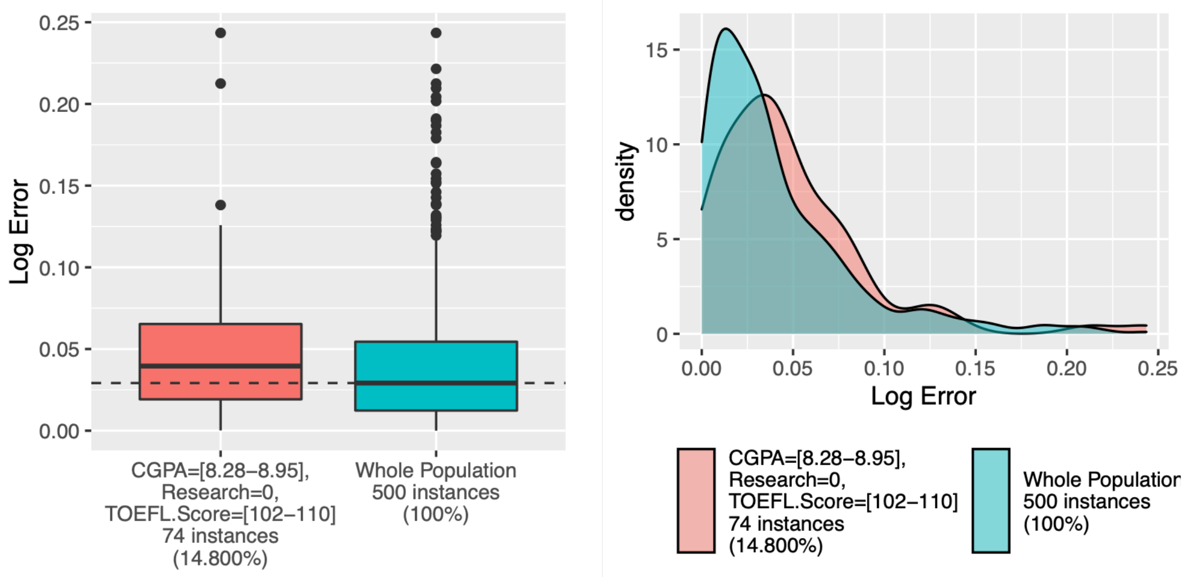


(a) Boxplot.

(b) Density plot.

Figure 60: EDR from the case study dataset to analyze subgroup $CGPA = [9.54 - 9.92], Research = 1$, trained with a RF model (cf. Table 2).

$[8.28 - 8.95], Research = 0, TOEFL.Score = [102 - 110]$, visible in Figure 61. This is associated with the SVM model and has an overall worse performance than previously expected by the general error values, having higher values for all the cutting points of the distribution. Additionally, the distribution of the subgroup is similar to the global one, but it is centered around a higher value, as seen in both the boxplot and density plot representations. Furthermore, this subgroup has a considerable representation, being composed by 74 instances, corresponding to 14.8% of the data.



(a) Boxplot.

(b) Density plot.

Figure 61: EDR from the case study dataset to analyze subgroup $CGPA = [8.28 - 8.95], Research = 0, TOEFL.Score = [102 - 110]$, trained with a SVM model (cf. Table 2).

Another crucial aspect of the analysis is to compare EDRs that are detected on more than one model, but not all. One example of such analysis is presented in Figure 62, illustrating the conditions $CGPA = [8.96 - 9.53], SOP = [4 - 4.5]$. This subgroup was detected on both the RF model and the SVM, being defined by smaller error values than predicted. The error values for both are extremely similar. However, the ones of the RF are slightly smaller, as seen by all the cutting points, except the median. The latter is marginally higher for the RF model than for the SVM. Nonetheless, overall, the RF is the model that predicts more accurate values for this combination of feature values, even though its global performance is worse than the one of the SVM.

The last example is similar to the one before, portraying a subgroup detected on only two models, namely the ANN and the SVM. Figure 63 depicts the subgroup in question, characterized by $University. Rating = [4 - 4], GRE.Score = [324 - 335]$ and by smaller error values than globally expected for both models. In both cases, not only every cutting point has a lower value than its global counterpart, but also their Q3 is relatively similar to their global median values. Nonetheless, the SVM largely outperforms the ANN, as it is composed by a range of smaller error values than the latter model.

5.4.4 Overall Analysis of the Experimental Results

After completing the various stages of the analysis process, some conclusions can be drawn about the evaluated models. For instance, the ANN is probably the model with the highest errors both globally and for the majority of detected subgroups. Moreover, the best overall models are the GBM and the SVM, due to having low error values globally and, again, for the great part of the groups. The remaining model, the RF, is the model in between, as it has usually presents slightly higher errors than the latter two and smaller than the ANN. Nevertheless, the two best models have some key aspects that may be crucial in

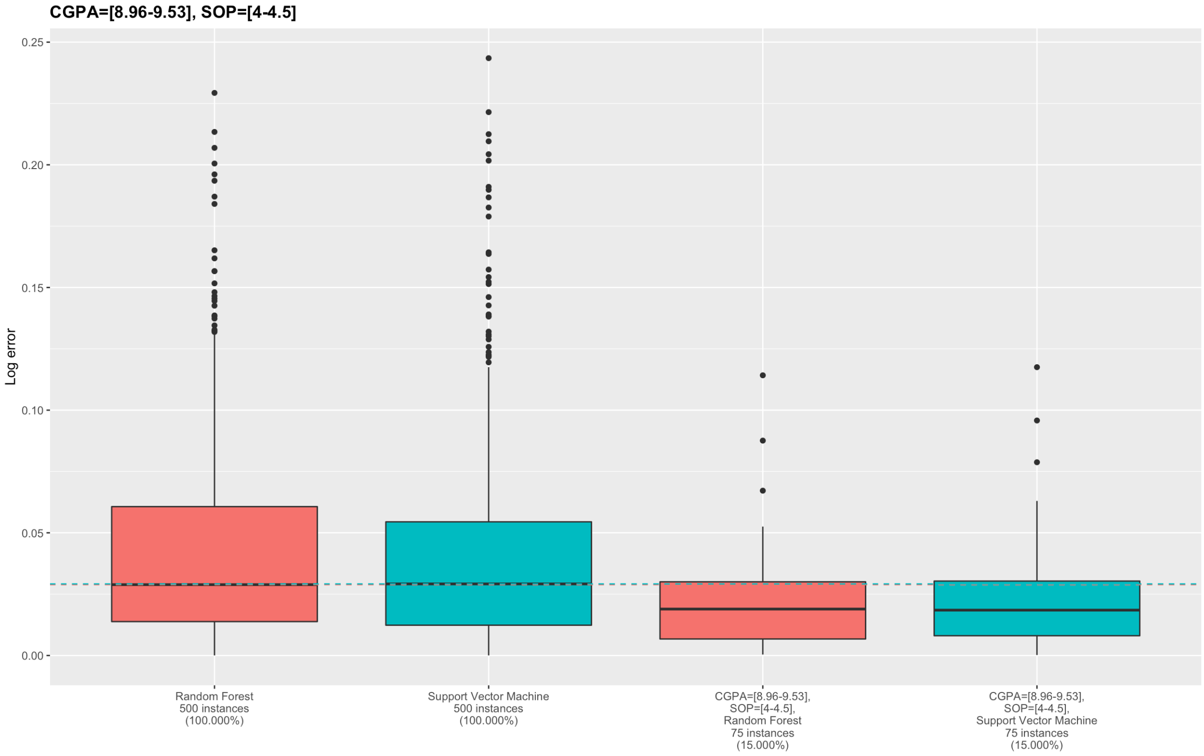


Figure 62: EDR from the case study dataset to analyze subgroup $CGPA = [8.96 - 9.53], SOP = [4 - 4.5]$, trained with a RF and a SVM models (cf. Table 2).

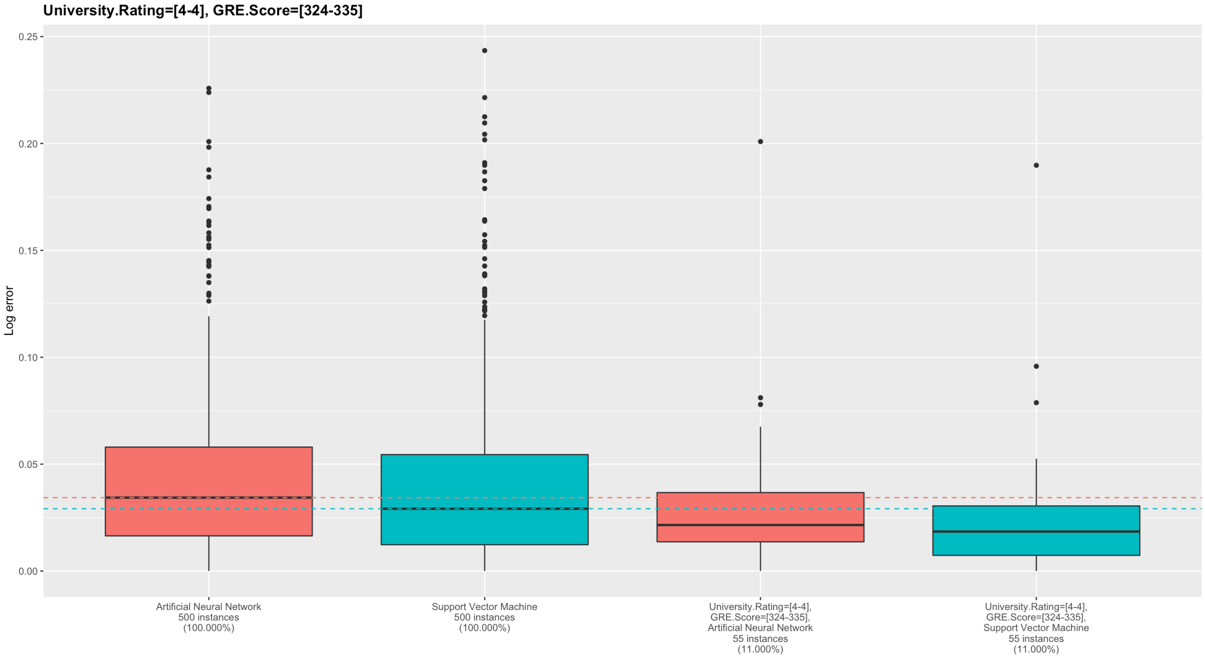


Figure 63: EDR from the case study dataset to analyze subgroup $University.Rating = [4 - 4], GRE.Score = [324 - 335]$, trained with an ANN and a SVM models (cf. Table 2).

the decision. For example, while the GBM has only 33 EDRs, all with one or two conditions of features, the SVM has 48 rules, ranging from one feature, to four. Of these, 54.5% have better performance than expected for the first and 56.25% for the second model.

Furthermore, depending on the needs of the end users, EDRs detected on a model over another may be taken in consideration, as these can be evaluated has having a more distinct behavior than the remaining subgroups. Comparing the two models, some rules that were detected only on the GBM are $CGPA = [7.64 - 8.27], SOP = [2.5 - 3.5]$; $SOP = [2.5 - 3.5]$; or $TOEFL.Score = [95 - 101], LOR = [2.5 - 3.5]$. Contrarily, $CGPA = [6.8 - 7.6]$; $CGPA = [8.28 - 8.95], Research = 0, TOEFL.Score = [102 - 110]$; or $LOR = [2.5 - 3.5], TOEFL.Score = [102 - 110]$ were only identified on the SVM. Curiously, these six subgroups have higher errors than expected by their global models. Another important aspect concerns the shape of the distributions of error values. On the one hand, the SVM tends to have more centered and concise error values. On the other hand, the GBM usually has a wider range of possible errors, from very low, to considerably mid-high values, around or above the ones of the former model.

However, there are some combinations of characteristics where the two models are not the best performing ones. An example of this situation was previously depicted in Figure 62, portraying a subgroup with higher predictive accuracy on the RF model. Moreover, it is also notorious that, even though the four models have similar performance metrics, globally, some are considerably better than others, especially for specific combinations of feature values. Therefore, as some models produce more accurate results for some combinations of features, it is easier for users to select the model that is more adequate to their needs. Another possible outcome of this analytic process is the generation of ensembles, i.e., combinations of models to predict results more accurately, based on the performance of specific subgroups and, consequently, taking advantage of the best aspects of each predictive model.

Lastly, an important note goes to the selection of rule discovery values. As stated before, the support and the pruning filters are extremely important and vary from problem to problem, dictating which rules are to be discovered and emphasized. A problem with a larger scale might rely on higher filtering values, focusing on interactions with a high impact on the overall results. While another one, more concerned about small scale interactions that differ from the expected behavior, may be dependent on smaller filtering values. Therefore, not only the EDRs themselves have to be taken in consideration, but their scale and behavior as well, leading to more pleasant results based on the requirements of the end users.

5.5 Summary

A case study was also considered, allowing the test of the developed tools using a realistic problem. An important condition of this test was the selection of a problem that could be tackled without much knowledge of the domain, leading to the selection of a dataset easily comprehended. This focused on the study of applications to graduate studies from an Indian perspective, based on the profiles of both student and institutions. The dataset is comprised of many factors, such as GRE and TOEFL scores, LOR, CGPA, among others. The methodology used was similar as before for the discretization of numerical features, used models and error calculation.

The models applied to the data were first compared based on scalar metrics, as the MAE and RMSE. Then, using global scalar metrics, examining the difference between the real distribution of values and the predicted ones, and overall errors, both logarithmic and residuals. The last step was to compare the four models with the aid of rules, utilizing the various tools proposed. The multiple comparisons led to some conclusions. For instance, the ANN appears to be the model with higher errors overall, globally and for the majority of the subgroups. The RF can be seen as the model in between, not being characterized by the smaller errors values globally, but being the best model for some specific cases. The remaining models, the GBM and the SVM, can be seen as two best choices for the problem, as is. Nonetheless, there are some important aspects that have to be taken in consideration to make a final decision. The GBM has its errors more spread and usually reaches higher values. Differently, the SVM has a more concise distribution, hitting lower top values than the GBM, but tends to start at greater values than the latter. Moreover, there are some subgroups that are only detected on a model, leading to the need for higher attention based on user requirements. Similarly, not all subgroups perform best on these two models, as stated before. This may be helpful to generate ensembles of models, taking advantage of the best performing models for the various conditions.

Lastly, the filtering values have a high impact on the discovered EDRs and should be taking in consideration early, as these guide the discovery process. On the one hand, higher filtering values lead to the discovery of groups with high impact on the overall. On the other hand, lower values also detect subgroups with small scale, that differ from the overall behavior and can be important to study very specific scenarios.

Chapter 6

Conclusions and Future Work

The main goal of this dissertation comprised the development of a novel approach to study the error values produced by black box regression models. Furthermore, this approach was supposed to follow a drill-down methodology, i.e., to successively analyze levels of data with more detail. Hence, DRs were used in the analytic process, as these use a drill-down approach to identify data groups. The other tool used to guide the process were EDPs, as these provide users with a fast and simple method to study the performance of combinations of data. Nevertheless, EDPs have a few considerable problems. Namely, not allowing the study of feature interactions composed by more than three variables and producing a substantial number of plots to analyze, when considering feature interactions. Note that the authors of this tool addressed the problems by proposing PEPs. Despite the resolution of the foretold problems, PEPs are visually less informative, as these do not show the distribution of values, and are more complex to study due the enormous amount of lines plotted.

In this work, we introduced EDRs, a combination of DRs and EDPs, taking advantage of the best aspects of each. By relying on the discovery process of DRs, EDRs are capable to detect combinations of feature values with more than three variables, eradicating any dimensionality restrictions. Additionally, these can be seen as an extension of EDPs, not being model dependent and producing fewer plots to analyze, due to filtering subgroups that are not interesting to point out, based on user preferences. In order to extend the analytic process, some visualization methods were developed, in addition to boxplots. Each rule can be plotted as a histogram or a density plot, allowing a deeper understanding of the behavior of the distributions. A network visualization of a group of rules was also implemented, allowing users to easily highlight interactions and see overall metrics of a certain combination of feature values. Furthermore, an overall graphical comparison of cutting point values of the distribution values of EDRs was developed, in the form of performance tables. These colorize each cutting point with a performance color based on how it compares to a reference distribution. Lastly, using the results of latter process, an extrapolation of counter-factual analysis to be applied in regression was realized. These provide users with meaningful scenarios where the performance of a subgroup alters considerably based on reference values.

Nevertheless, the usage of EDRs does not exhaust the domain of variable values, due to the filtering nature of their discovery process. Not only groups of data with low representation are filtered, but also subgroups that do not differ significantly enough from their parent subgroups, i.e., subgroups that have less conditions but share them. Consequently, support and the pruning filter values are extremely important, controlling which rules are to be discovered, and vary from problem to problem. Thus, EDRs and EDPs complement each other. On the one hand, the first highlight distributions of data that differ significantly from the reference and from subgroups that share similar conditions. On the other hand, the second show every possible combination of values on interactions up to three variables.

In order to validate the proposed approach, multiple datasets were used to train a diverse group of regression algorithms. By doing so, any model bias was eliminated from the experiments. Moreover, the used datasets have various dimensions, both in terms of number of instances, and number of features, with the latter being comprised of multiple combinations between numerical and categorical values. The experiments performed on these allowed us to detected combinations of feature values that would not be easily detected using EDPs, requiring the usage of the multivariate variants of the latter and producing a considerable number of plots. Moreover, some identified subgroups would never be found with EDPs, due to being composed by more than three variables.

The last study performed simulated the selection of a model on a scenario the closest to reality possible. In this case, the problem selected concerned the prediction of probability of admission on graduate programs, considering aspects of the student applying and the institution itself. By using the proposed tools, it was possible to confirm that the best models overall are not always the best for all combinations of feature values, being easily outperformed, depending on user preferences. Hence, EDRs give users an extra layer of understanding on how each model behaves regarding error values and helps in the selection process, allowing users to compare performance in very specific situations.

As stated throughout this project, EDRs greatly facilitate the performance analysis of black box regression models. However, these are far from perfect, with results highly dependent on the discretization of numerical features. Thus, it would be interesting to study the impact of developing a method to discretize such feature values guided by the discovered rules, instead of using an external algorithm. For instance, applying multiple discretization approaches and selecting the one that guides the process to a desirable effect, e.g., the rules that differ most significantly from a reference. Another important step to further the tool lies on the need to define meta-models to produce ensembles, weighting the models based on the performance these present for each subgroup.

Bibliography

[1] Patrick Hall. *An introduction to machine learning interpretability*. O'Reilly Media, Incorporated, 2019.

[2] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[3] Matt Turek. Explainable artificial intelligence (xai). defense advanced research projects agency. `https://www.darpa.mil/program/explainable-artificial-intelligence`, 2018. Acessed: 2020-10-22.

[4] Recital 71 - profiling | general data protection regulation (gdpr). URL `https://gdpr-info.eu/recitals/no-71/`. Acessed: 2020-10-23.

[5] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[7] Michael Van Lent, William Fisher, and Michael Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the national conference on artificial intelligence*, pages 900–907. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[8] W. R. Swartout and J. D. Moore. Explanation in expert systems: A survey, 1988.

[9] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[10] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

[11] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

[12] Ayanna Howard, Cha Zhang, and Eric Horvitz. Addressing bias in machine learning algorithms: A pilot study on emotion recognition for intelligent systems. In *2017 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pages 1–7. IEEE, 2017.

[13] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An approach to evaluating interpretability of machine learning. *arXiv preprint arXiv:1806.00069*, 2018.

[14] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[15] Mengnan Du, Ninghao Liu, and Xia Hu. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77, 2019.

[16] Matthew Britton. Vine: Visualizing statistical interactions in black box models. *arXiv preprint arXiv:1904.00561*, 2019.

[17] Resources :: Fat ml, principles for accountable algorithms and a social impact statement for algorithms. URL `https://www.fatml.org/resources#principles-for-accountable-algorithms`. Acessed: 2020-12-17.

[18] Oded Maimon and Lior Rokach. Data mining and knowledge discovery handbook. 2005.

[19] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[20] Maxim Vladimirovich Shcherbakov, Adriaan Brebels, Nataliya Lvovna Shcherbakova, Anton Pavlovich Tyukov, Timur Alexandrovich Janovsky, and Valeriy Anatol'evich Kamaev. A survey of forecast error measures. *World Applied Sciences Journal*, 24(24):171–176, 2013.

[21] Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.

[22] Hirotogu Akaike. Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213. Springer, 1998.

[23] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[24] Galit Shmueli, Peter C Bruce, Inbal Yahav, Nitin R Patel, and Kenneth C Lichtendahl Jr. *Data mining for business analytics: concepts, techniques, and applications in R*. John Wiley & Sons, 2017.

[25] Jinbo Bi and Kristin P Bennett. Regression error characteristic curves. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 43–50, 2003.

[26] Luís Torgo. Regression error characteristic surfaces. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 697–702, 2005.

[27] José Hernández-Orallo. Roc curves for regression. *Pattern Recognition*, 46(12):3395–3411, 2013.

[28] Inês Areosa and Luís Torgo. Visual interpretation of regression error. In *EPIA Conference on Artificial Intelligence*, pages 473–485. Springer, 2019.

[29] Inês Areosa and Luís Torgo. Explaining the performance of black box regression models. In *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 110–118. IEEE, 2019.

[30] Alfred Inselberg. The plane with parallel coordinates. *The visual computer*, 1(2):69–91, 1985.

[31] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[32] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[33] Igor Kononenko and Matjaz Kukar. *Machine learning and data mining*. Horwood Publishing, 2007.

[34] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 20(177):1–81, 2019.

[35] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. Visualizing the feature importance for black box models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 655–670. Springer, 2018.

[36] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[37] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

[38] Daniel W Apley and Jingyu Zhu. Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(4): 1059–1086, 2020.

[39] Brandon M Greenwell, Bradley C Boehmke, and Andrew J McCarthy. A simple and effective model-based variable importance measure. *arXiv preprint arXiv:1805.04755*, 2018.

[40] Brandon M Greenwell. pdp: An r package for constructing partial dependence plots. *R J.*, 9(1):421, 2017.

[41] Jerome H Friedman, Bogdan E Popescu, et al. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3):916–954, 2008.

[42] Giles Hooker. Discovering additive structure in black box functions. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 575–580, 2004.

[43] Benjamin P Evans, Bing Xue, and Mengjie Zhang. What's inside the black-box? a genetic programming method for interpreting complex machine learning models. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1012–1020, 2019.

[44] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpretability via model extraction. *arXiv preprint arXiv:1706.09773*, 2017.

[45] Mark W Craven. Extracting comprehensible models from trained neural networks. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1996.

[46] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820*, 2018.

[47] Leo Breiman et al. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001.

[48] Terence Parr and Jeremy Howard Kerem Turgutlu, Christopher Csiszar. Beware default random forest importances. `https://explained.ai/rf-importance/#intro`. Acessed: 2020-12-03.

[49] Carolin Strobl, Anne-Laure Boulesteix, Thomas Kneib, Thomas Augustin, and Achim Zeileis. Conditional variable importance for random forests. *BMC bioinformatics*, 9(1):307, 2008.

[50] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[51] Erik Štrumbelj and Igor Kononenko. A general method for visualizing and explaining black-box regression models. In *International Conference on Adaptive and Natural Computing Algorithms*, pages 21–30. Springer, 2011.

[52] Wayne A Larsen and Susan J McCleary. The use of partial residual plots in regression analysis. *Technometrics*, 14(3):781–790, 1972.

[53] R Dennis Cook. Exploring partial residual plots. *Technometrics*, 35(4):351–362, 1993.

[54] Chih-Ling Tsai, Zongwu Cai, and Xizhi Wu. The examination of residual plots. *Statistica Sinica*, pages 445–465, 1998.

[55] Robert E Weiss and Carlos G Lazaro. Residual plots for repeated measures. *Statistics in Medicine*, 11(1):115–124, 1992.

[56] Richard A Becker, William S Cleveland, and Ming-Jen Shyu. The visual design and control of trellis display. *Journal of computational and Graphical Statistics*, 5(2):123–155, 1996.

[57] JM Chambers and TJ Hastie. *Statistical models in S*. Wadsworth & Brooks/Cole, 1992.

[58] William Cleveland. *Visualizing data*. Hobart Press, 1993.

[59] R Dennis Cook. Graphics for studying net effects of regression predictors. *Statistica Sinica*, pages 689–708, 1995.

[60] Ceteris paribus plots. URL `https://pbiecek.github.io/ceterisParibus/`. Acessed: 2020-11-16.

[61] Przemysław Biecek. Dalex: explainers for complex predictive models in r. *The Journal of Machine Learning Research*, 19(1):3245–3249, 2018.

[62] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " 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*, pages 1135–1144, 2016.

[63] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI*, volume 18, pages 1527–1535, 2018.

[64] Mateusz Staniak and Przemyslaw Biecek. Explanations of model predictions with live and breakdown packages. *arXiv preprint arXiv:1804.01955*, 2018.

[65] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[66] Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523): 1094–1111, 2018.

[67] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

[68] Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617. IEEE, 2016.

[69] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018.

[70] Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317, 1953.

[71] Katsushige Fujimoto, Ivan Kojadinovic, and Jean-Luc Marichal. Axiomatic characterizations of probabilistic and cardinal-probabilistic interaction indices. *Games and Economic Behavior*, 55(1):72–99, 2006.

[72] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[73] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[74] Kacper Sokol and Peter A Flach. Counterfactual explanations of machine learning predictions: opportunities and challenges for ai safety. In *SafeAI@ AAAI*, 2019.

[75] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[76] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th international conference on world wide web*, pages 1171–1180, 2017.

[77] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th international conference on intelligent user interfaces*, pages 126–137, 2015.

[78] Josua Krause, Adam Perer, and Kenney Ng. Interacting with predictions: Visual inspection of black-box machine learning models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 5686–5697, 2016.

[79] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 207–216, 1993.

[80] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules, 1995.

[81] Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 1–12, 1996.

[82] Yonatan Aumann and Yehuda Lindell. A statistical theory for quantitative association rules. *Journal of Intelligent Information Systems*, 20(3):255–283, 2003.

[83] Geoffrey I Webb. Discovering associations with numeric variables. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 383–388, 2001.

[84] Geoffrey I Webb. Opus: An efficient admissible algorithm for unordered search. *Journal of Artificial Intelligence Research*, 3:431–465, 1995.

[85] Shiying Huang and Geoffrey I Webb. Discarding insignificant rules during impact rule discovery in large, dense databases. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 541–545. SIAM, 2005.

[86] Shiying Huang and Geoffrey I Webb. Pruning derivative partial rules during impact rule discovery. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 71–80. Springer, 2005.

[87] Shiying Huang and Geoffrey I Webb. Efficiently identifying exploratory rules' significance. In *Data Mining*, pages 64–77. Springer, 2006.

[88] Alípio M Jorge, Paulo J Azevedo, and Fernando Pereira. Distribution rules with numeric attributes of interest. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 247–258. Springer, 2006.

[89] R Core Team. *R: The R Project for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL `https://www.r-project.org/`. Acessed: 2021-03-17.

[90] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL `http://www.stats.ox.ac.uk/pub/MASS4`. ISBN 0-387-95457-0, Acessed: 2021-03-17.

[91] Brandon Greenwell, Bradley Boehmke, Jay Cunningham, and GBM Developers. *gbm: Generalized Boosted Regression Models*, 2020. URL `https://CRAN.R-project.org/package=gbm`. R package version 2.1.8, Acessed: 2021-03-17.

[92] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3): 18–22, 2002. URL `https://CRAN.R-project.org/doc/Rnews/`. Acessed: 2021-03-17.

[93] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2020. URL `https://CRAN.R-project.org/package=e1071`. R package version 1.7-4, Acessed: 2021-03-17.

[94] Paulo J Azevedo. Caren - class project association rule engine. URL `https://www.di.uminho.pt/~pja/class/caren.html`. Acessed: 2021-03-17.

[95] Paulo J Azevedo. Caren-a java based apriori implementation for classification purposes. Technical report, Université de Mons-Hainaut. Service de Science des Systèmes D'information, 2003.

[96] Paulo J Azevedo. A data structure to represent association rules based classifiers. Technical report, Technical report, Universidade do Minho, Departamento de Informática, 2005.

[97] Usama Fayyad and Keki Irani. Multi-interval discretization of continuous-valued attributes for classification learning. 1993.

[98] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL `https://ggplot2.tidyverse.org`. Acessed: 2021-03-17.

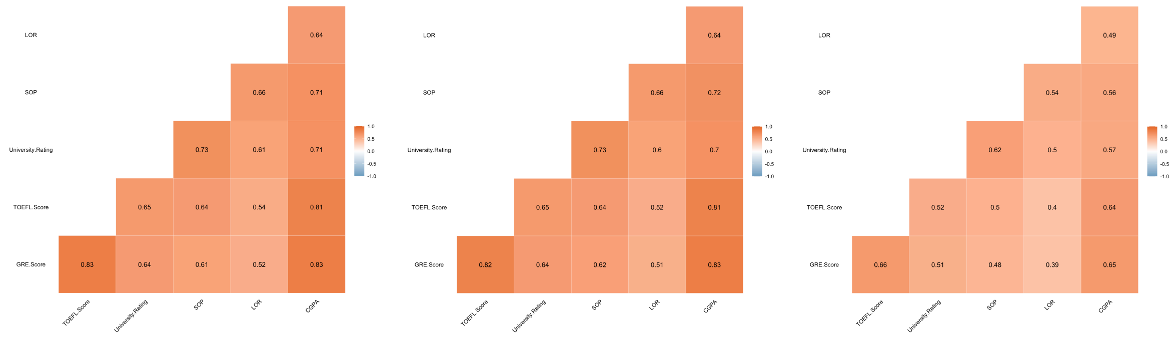
[99] John W Tukey et al. *Exploratory data analysis*, volume 2. Reading, Mass., 1977.

[100] Almende B.V., Benoit Thieurmel, and Titouan Robert. *visNetwork: Network Visualization using 'vis.js' Library*, 2019. URL `https://CRAN.R-project.org/package=visNetwork`. R package version 2.0.9.

[101] Mohan S Acharya, Asfia Armaan, and Aneeta S Antony. A comparison of regression models for prediction of graduate admissions. In *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, pages 1–5. IEEE, 2019.

Appendix A

Case Study Support Analysis

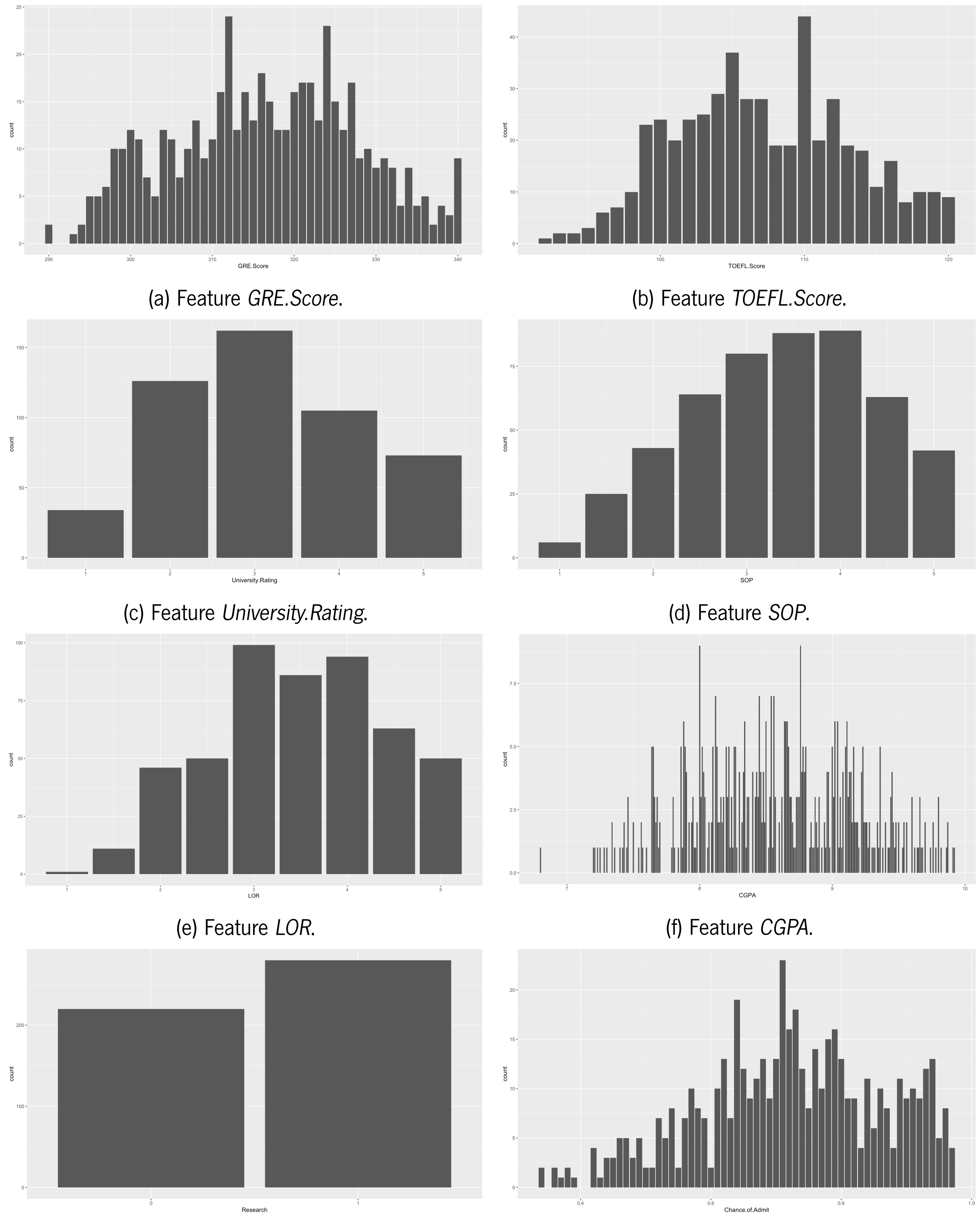


(a) *Pearson* coefficient.

(b) *Spearman* coefficient.

(c) *Kendall* coefficient.

Figure 64: Correlation between numerical predictors for the case study dataset, calculated using multiple coefficients.



(a) Feature *GRE.Score*.

(b) Feature *TOEFL.Score*.

(c) Feature *University.Rating*.

(d) Feature *SOP*.

(e) Feature *LOR*.

(f) Feature *CGPA*.

(g) Feature *Research*.

(h) Target *Chance.of.Admit*.

Figure 65: Graphical representation of the distributions of each variable of the case study dataset.