UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JUSCIMARA GOMES AVELINO

IMBALANCED REGRESSION PIPELINE RECOMMENDATION

Recife

2024

JUSCIMARA GOMES AVELINO

IMBALANCED REGRESSION PIPELINE RECOMMENDATION

Doctoral thesis presented to the Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, as a partial requirement for obtaining the degree of Doctor in Computer Science.

Concentration Area: Computational Intelligence

Advisor: George Darmiton da Cunha Cavalcanti

Co-advisor: Rafael Menelau Oliveira e Cruz

Recife

2024

.Catalogação de Publicação na Fonte. UFPE - Biblioteca Central

Avelino, Juscimara Gomes.
Imbalanced Regression Pipeline Recommendation / Juscimara Gomes Avelino. - Recife, 2024.
138f.: il.

Tese (Doutorado) - Universidade Federal de Pernambuco, Centro de Informática, Programa de Pós-Graduação em Ciência da Computação, 2024.
Orientação: George Darmiton da Cunha Cavalcanti.
Coorientação: Rafael Menelau Oliveira e Cruz.

1. Regressão desbalanceada; 2. Estratégias de reamostragem; 3. Meta-aprendizado. I. Cavalcanti, George Darmiton da Cunha. II. Cruz, Rafael Menelau Oliveira e. III. Título.

UFPE-Biblioteca Central

JUSCIMARA GOMES AVELINO

IMBALANCED REGRESSION PIPELINE RECOMMENDATION

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Aprovada em: 26/08/2024

BANCA EXAMINADORA

_____
Prof. Dr. Ricardo Bastos Cavalcante Prudêncio
Centro de Informática/UFPE

_____
Prof. Dr. Tsang Ing Ren
Centro de Informática/UFPE

_____
Profa. Dra. Ana Carolina Lorena
Instituto Tecnológico de Aeronáutica

_____
Prof. Dr. Péricles Barbosa Cunha de Miranda
Departamento de Computação / UFRPE

_____
Prof. Dr. Alceu de Souza Britto Junior
Programa de Pós Graduação em Informática Aplicada/PUC/PR

ACKNOWLEDGEMENTS

Primeiramente, agradeço a Deus por me conceder força, sabedoria e perseverança ao longo desta jornada. Sem Sua graça, esta conquista não teria sido possível.

À minha família, expresso minha mais profunda gratidão. Aos meus pais, **Juscelino Avelino** e **Verônica Gomes**, agradeço por todo o amor, apoio incondicional e por acreditarem em mim em todos os momentos, mesmo nos mais desafiadores. Ao meu irmão, **Juscelino Jr.**, sou grata por ser uma fonte constante de inspiração, apoio e motivação.

Aos meus orientadores, **George Darmiton da Cunha Cavalcanti** e **Rafael Menelau Oliveira e Cruz**, minha imensa gratidão pelo apoio contínuo ao longo deste processo. A dedicação, paciência e confiança no meu trabalho foram essenciais para o desenvolvimento e conclusão deste trabalho.

Agradeço também ao apoio financeiro da Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (**FACEPE**) e do Conselho Nacional de Desenvolvimento Científico e Tecnológico (**CNPq**), cuja contribuição foi fundamental para a concretização desta tese.

RESUMO

Problemas de desbalanceamento são comuns em diversos cenários do mundo real e apresentam desafios significativos, especialmente para tarefas de regressão, devido à raridade de certos valores-alvo contínuos. Embora essas questões tenham sido amplamente exploradas em tarefas de classificação, elas também afetam a regressão, complicando o desempenho dos modelos. Este trabalho apresenta um estudo experimental extenso envolvendo várias estratégias de balanceamento e modelos de aprendizado, introduzimos uma taxonomia para abordagens de regressão desbalanceada baseada em modelos de regressão, modificação no processo de aprendizado e métricas de avaliação, e destaca novos insights sobre as vantagens de diferentes estratégias. A partir deste estudo, ficou evidente que a escolha do método de reamostragem depende do problema, dos modelos de aprendizado e das métricas, tornando difícil selecionar uma estratégia de reamostragem e um modelo de aprendizado apropriados. Como resultado, é necessário testar a maioria das combinações existentes. Com base nessas descobertas, este trabalho propõe o modelo Meta-learning for Imbalanced Regression (Meta-IR) para enfrentar esses desafios. O Meta-IR recomenda pipelines ideais que consistem em estratégias de reamostragem e modelos de aprendizado para tarefas de regressão desbalanceada. Duas formulações são propostas: Independente, que recomenda separadamente algoritmos de aprendizado e estratégias de reamostragem, e Encadeada, que modela suas interdependências sequencialmente. A abordagem Encadeada demonstrou desempenho superior, sugerindo uma relação significativa entre algoritmos de aprendizado e estratégias de reamostragem. Em comparação com modelos de AutoML e configurações de linha de base, o Meta-IR superou todos, oferecendo uma solução mais eficaz para a regressão desbalanceada e indicando direções para futuras pesquisas.

Palavras-chave: Regressão desbalanceada, Estratégias de reamostragem, Meta-aprendizado

ABSTRACT

Imbalanced problems are common in various real-world scenarios and present significant challenges, especially for regression tasks due to the rarity of certain continuous target values. While these issues have been extensively explored in classification tasks, they also affect regression, complicating model performance. This work presents an extensive experimental study involving various balancing strategies and learning models, introduces a taxonomy for imbalanced regression approaches based on regression models, learning process modification, and evaluation metrics, and highlights new insights into the advantages of different strategies. From this study, it became evident that the choice of resampling method depends on the problem, learning models, and metrics, making it difficult to select an appropriate resampling strategy and learning model. As a result, it is necessary to test the majority of existing combinations. Based on these findings, this work proposes the Meta-learning for Imbalanced Regression (Meta-IR) framework to address these challenges. Meta-IR recommends optimal pipelines consisting of resampling strategies and learning models for imbalanced regression tasks. Two formulations are proposed: Independent, which separately recommends learning algorithms and resampling strategies, and Chained, which models their interdependencies sequentially. The Chained approach demonstrated superior performance, suggesting a significant relationship between learning algorithms and resampling strategies. Compared with AutoML models and baseline configurations, Meta-IR outperformed all, offering a more effective solution for imbalanced regression and indicating directions for future research.

Keywords: Imbalanced Regression, Resampling Strategies, Meta-learning.

LIST OF FIGURES

Figure 1 – A general architecture for meta-learning systems that address the algorithm selection problem. 23

Figure 2 – Distribution and frequency of the target value Y from the FuelCons dataset. 28

Figure 3 – Relevance function of the fuelCons dataset. 34

Figure 4 – Proposed taxonomy for imbalanced regression problems. 36

Figure 5 – Bumps partition of Y with respect to relevance function ϕ and the maximum admissible loss in bumps. Each bump i is characterized by its partition node b^- and by one global maximum b^*. Each bump has a maximum error tolerance defined by the double of the smalles amplitude in the bump between each of one of its bounds and its maximum value. 43

Figure 6 – Distribution of the examples of the FuelCons dataset after applying the resampling strategies, considering t_R=0.8. 56

Figure 7 – Critical difference diagrams for each learning algorithm considering the F1-score metric. 65

Figure 8 – Critical difference diagrams for each learning algorithm considering the SERA metric. 66

Figure 9 – Percentage of increase/decrease in the training set for each resampling strategy. 67

Figure 10 – Evolution of the *F1-score* with datasets sorted by percentage of rare cases. 68

Figure 11 – Evolution of the *F1-score* with datasets sorted by number of rare cases. . . 69

Figure 12 – Evolution of the *F1-score* with datasets sorted by size. 69

Figure 13 – Evolution of the *F1-score* with datasets sorted by number of attributes. . . 70

Figure 14 – Evolution of the *F1-score* with datasets sorted by imbalance ratio. 70

Figure 15 – Distribution and frequency of the target value Y from the FuelCons dataset. Stronger colors indicate less represented examples (rare examples), while lighter colors indicate more represented examples (normal examples). . . . 76

Figure 16 – Comparison between our proposed meta-learning-based model Meta-IR and traditional AutoML. 78

Figure 17 – Distribution of the examples of the FuelCons dataset after applying the resampling strategies, considering t_R=0.8. 83

Figure 18 – The proposed Meta-IR framework. (I) **Meta-dataset construction:** the meta-features ($\mathbf{f}$) are extracted, and the pipelines are evaluated to construct the meta-dataset $\mathbf{M}$. (II) **Meta-classifiers training:** the two meta-classifiers (λ_L and λ_R) are trained. (III) **Recommendation:** Given the meta-features ($\mathbf{f_G}$) of an imbalanced regression dataset $\mathbf{G}$, this phase recommends a learning model ($l^* \in \mathbb{L}$) and a resampling strategy ($r^* \in \mathbb{R}$). . 92
Figure 19 – Training of the meta-classifiers λ_L and λ_R. 93
Figure 20 – Meta-IR performance for each type of training (Independent, Model First, and Strategy First) compared with the baselines Random and Majority. . . 98
Figure 21 – Number of wins over all datasets: Meta-IR *versus* each pipeline (resampling strategy and regressor). 100
Figure 22 – Critical Difference diagrams obtained from the recommendations given by the models. 102
Figure 23 – Critical Difference diagrams obtained from the recommendations given by the models. 105
Figure 24 – Time and performance of each model. 107
Figure 25 – Feature Importance for recommending the Learning model (a-c) and the Resampling Strategy (d-f), considering F1-scoreR metric as optimization function. 109
Figure 26 – Feature Importance for recommending the Learning model (a-c) and the Resampling Strategy (d-f), considering Squared error-relevance area (SERA) metric as optimization function. 110
Figure 27 – Frequency of resampling strategies and learning models used as meta-target. 138

LIST OF ALGORITHMS

Algorithm 1 – pchip(S): Piecewise Cubic Hermite Interpolating Polynomials . . . 33

Algorithm 2 – check_slopes (Φ, Δ) (FRITSCH; CARLSON, 1980) 35

Algorithm 3 – SmoteR 47

Algorithm 4 – Generating synthetic cases 48

Algorithm 5 – Random over-sampling 49

Algorithm 6 – Random under-sampling 49

Algorithm 7 – Introduction of Gaussian Noise 51

Algorithm 8 – SMOGN 53

Algorithm 9 – WEighted Relevance-based Combination Strategy (WERCS) 54

LIST OF TABLES

Table 1 – Control points of LNO2 concentration thresholds according to Directive 2008/50/EC. 33

Table 2 – Predictions of two artificial models. 40

Table 3 – Performances of two artificial models 41

Table 4 – Characteristics of the 30 datasets used in the experiments. N: number of cases; p.total: number of attributes; p.nom: number of nominal attributes; p.num: number of numeric attributes; nRare: number of rare cases; Imbalance ratio (IR): $\frac{|D_R|}{|D_N|}$; %Rare: $100 \times nRare/N$. Datasets are arranged in descending order regarding the percentage of rare cases (%Rare). 59

Table 5 – Resampling strategies, hyperparameters, and packages used 60

Table 6 – Number of times each algorithm and resampling strategy achieved the best result according to the F1-score metric. 62

Table 7 – Number of times each algorithm and resampling strategy reached the best result according to the SERA metric. 62

Table 8 – Best and worst results for each dataset based on the F1-score metric 63

Table 9 – Best and worst results for each dataset based on the SERA metric 64

Table 10 – Average ranking (F1-score). 71

Table 11 – Average ranking (SERA). 71

Table 12 – Meta-learning related works 87

Table 13 – Related works. 88

Table 14 – Pairwise comparison between the performances of the Meta-IR *versus* each pipeline (resampling strategy and regressor) over all datasets. The values represent the p-value of the Wilcoxon signed-rank test using the F1-scoreR metric. All results with $\alpha < 0.05$ are in bold. 101

Table 15 – Pairwise comparison between the performances of the Meta-IR *versus* each pipeline (resampling strategy and regressor) over all datasets. The values represent the p-value of the Wilcoxon signed-rank test using the SERA metric. All results with $\alpha < 0.05$ are in bold. 101

Table 16 – Pairwise comparison between the performances of the Meta-IR *versus* AutoML frameworks at each level of percentage of rare cases (High, Medium and Low). The values represent the p-value of the Wilcoxon signed-rank test using the SERA metric. All results with $\alpha < 0.05$ are in bold. 103
Table 17 – Win/tie/loss of the Meta-IR versus AutoML frameworks. 103
Table 18 – Meta-IR as a pre-processing step and Comparison with Baseline 104
Table 19 – Data sets used to calculate time. 106
Table 20 – Datasets ordered by the percentage of rare cases. (n.examples: Number of examples; n.attributes: Number of attributes; n.rare: Number of rare cases; p.rare: $100 \times n.raro/n.exemplos$ 123
Table 21 – Characteristic, acronym, aggregation functions and description of meta-features. 136
Table 22 – Performances (Accuracy) achieved by the meta-models for each type of training. Best results are in bold. 137
Table 23 – Performances (Accuracy) achieved by the meta-models for each type of training. Best results are in bold. 137

LIST OF ABBREVIATIONS AND ACRONYMS

AS	Algorithm Selection
ASP	Algorithm Selection Problem
AutoML	Automated Machine Learning
BG	Bagging
CASH	Combined algorithm selection and hyperparameter optimization
F1-scoreR	F1-score for Regression
GN	Introduction of Gaussian Noise
HPO	Hyperparameter optimization
Meta-IR	Meta-Learning for Imbalanced Regression
MLP	Multilayer Perceptron
MSE	Mean Squared Error
MtL	Meta-Learning
REBAGG	REsampled BAGGing
RF	Random Forest
RO	Randon Over-sampling
RT	Regression Tree
RU	Randon Under-sampling
SERA	Squared error-relevance area
SG	SMOGN
SMOGN	SmoteR with Gaussian Noise
SMT	SmoteR
SVR	Support Vector Regressor
WERCS	WEighted Relevance based Combination Strategy

XG XGBoost

LIST OF SYMBOLS

D - Original dataset

D_R - Rare data subset

D_N - Normal data subset

$\phi()$ - Relevance function

t_R - Relevance threshold

y_i - True value of example i

$\hat{y}_i$ - Predicted value of example i

$U^p_\phi()$ - Utility function

$\Gamma_B()$ - Benefit function

$\Gamma_C()$ - Cost function

l - Learning model

r - Resampling strategy

f - Meta-features

$\mathbb{D}$ - Set of datasets

$\mathbf{G}$ - Test data

λ_L - Meta-model for recommending the learning model

λ_R - Meta-model for recommending the resampling strategy

CONTENTS

1 INTRODUCTION 17
1.1 MOTIVATION AND MAIN CONTRIBUTIONS 18
1.2 OBJECTIVE AND HYPOTHESES 19
1.3 THESIS ORGANIZATION 20
2 BASIC CONCEPTS 22
2.1 META-LEARNING 22
2.1.1 Dataset characterization 24
2.1.2 Forms of Recommendation 25
2.2 AUTOMATED MACHINE LEARNING (AUTOML) 25
3 RESAMPLING STRATEGIES FOR IMBALANCED REGRESSION: A SURVEY AND EMPIRICAL ANALYSIS 27
3.1 INTRODUCTION 27
3.2 BASIC CONCEPTS AND PROPOSED TAXONOMY 31
3.2.1 Relevance Function 32
3.2.2 Proposed Taxonomy 36
3.3 RESAMPLING STRATEGIES 45
3.3.1 SmoteR 46
3.3.2 Random Over-sampling 48
3.3.3 Random Under-sampling 49
3.3.4 Introduction of Gaussian Noise 50
3.3.5 SmoteR with Gaussian Noise 51
3.3.6 WEighted Relevance based Combination Strategy 53
3.3.7 Advantages and Disadvantages of Strategies 54
3.4 RESEARCH METHODOLOGY 57
3.4.1 Datasets 57
3.4.2 Algorithms 58
3.4.3 Model Evaluation 58
3.5 RESULTS 61
3.6 LESSONS LEARNED 72
3.7 CONCLUSION 73

4 IMBALANCED REGRESSION PIPELINE RECOMMENDATION . 75
4.1 INTRODUCTION . . . 76
4.2 BACKGROUND . . . 80
4.2.1 Imbalanced Regression . . . 80
4.2.1.1 Relevance Function . . . 80
4.2.1.2 Resampling Strategies . . . 81
4.2.2 Meta-Learning . . . 84
4.2.3 Automated Machine Learning . . . 87
4.3 PROPOSED METHOD . . . 90
4.3.1 Problem Definition . . . 90
4.3.2 Meta-learning for Imbalanced Regression (Meta-IR) . . . 91
4.3.2.1 Meta-dataset construction phase . . . 91
4.3.2.2 Meta-classifiers training phase . . . 92
4.3.2.3 Recommendation Phase . . . 93
4.4 EXPERIMENTAL METHODOLOGY . . . 94
4.4.1 Datasets . . . 94
4.4.2 Meta-features . . . 94
4.4.3 Learning Algorithms . . . 94
4.4.4 Resampling strategies . . . 95
4.4.5 Meta-Models . . . 95
4.4.6 Evaluation Methodology . . . 95
4.5 RESULTS AND DISCUSSION . . . 97
4.5.1 Meta-level analysis . . . 97
4.5.2 Base-level analysis . . . 99
4.5.2.1 Comparison with each pipeline . . . 99
4.5.2.2 Comparison with AutoML frameworks . . . 101
4.5.2.3 Meta-IR as a pre-processing step for AutoML frameworks . . . 104
4.5.3 Execution time analysis . . . 106
4.5.4 Meta-features analysis . . . 106
4.6 CONCLUSIONS . . . 111
5 GENERAL CONCLUSION . . . 112
5.1 LIMITATIONS AND FUTURE WORK . . . 113
REFERENCES . . . 114

1 INTRODUCTION

In machine learning, imbalanced data poses significant challenges for analysis and model development (HE; GARCIA, 2009). This issue arises when certain classes or values are underrepresented, leading to biased models that perform inadequately on minority classes or rare values. Both classification and regression problems can suffer from imbalance. In classification, this manifests as one class (the minority class) having significantly fewer instances than others (the majority classes). In regression, where target values are continuous, imbalance can occur in more complex forms. To address this, Ribeiro (2011) introduced the concept of a relevance function that assesses the importance of continuous target values, distinguishing between rare and normal examples. This approach helps in identifying and rectifying imbalances to ensure a more balanced and fair analysis.

While the issue of imbalance in classification problems has been widely studied, it is also a significant but overlooked problem in regression tasks (BRANCO; RIBEIRO; TORGO, 2016). Learning algorithms often struggle with underrepresented values, focusing on more common value ranges and neglecting rare cases. This leads to poor performance on these specific cases. Additionally, evaluating model performance can be challenging, as some metrics may not capture what is most important to users (BRANCO; TORGO; RIBEIRO, 2019). Metrics that effectively capture the imbalanced problem include Precision, Recall, F1-score (TORGO; RIBEIRO, 2009), and the SERA metric (RIBEIRO; MONIZ, 2020).

One of the most common solutions to address imbalanced regression is the application of various resampling strategies. These strategies aim to balance the training data by altering the distribution of examples (GALAR et al., 2011). Key strategies include SmoteR (TORGO et al., 2013), Random Under-sampling (TORGO et al., 2013), Random Over-sampling (BRANCO; TORGO; RIBEIRO, 2019), Introduction of Gaussian Noise (BRANCO; TORGO; RIBEIRO, 2019), SmoteR with Gaussian Noise (SMOGN) (BRANCO; TORGO; RIBEIRO, 2017), and WEighted Relevance based Combination Strategy (WERCS) (BRANCO; TORGO; RIBEIRO, 2019). However, choosing the best solution among these strategies is complex, as it depends on the specific combination of resampling strategy and learning model for each dataset, which is a classic example of the algorithm selection problem.

The algorithm selection problem (RICE, 1976) involves determining the most effective algorithm for a given dataset based on its specific characteristics (BRAZDIL et al., 2022b). The

process of selecting the right solution can be achieved through the use of Meta-Learning (MtL) models. Meta-learning relies on meta-features to learn new tasks more efficiently (VANSCHOREN, 2019). Using a set of prior tasks, a meta-model is trained to correlate these meta-features with meta-labels, representing the best model for specific tasks. When faced with new tasks, the meta-model can then suggest suitable algorithms based on the meta-features of the new problem, effectively addressing the algorithm selection problem.

Based on this concept, this work introduces the Meta-Learning for Imbalanced Regression (Meta-IR) framework. Specifically designed to address challenges in imbalanced regression, Meta-IR employs meta-classifiers to recommend optimal pipelines that integrate both resampling strategies and learning models. The framework is presented in two distinct formulations: Independent and Chained. The Independent formulation separately recommends learning algorithms and resampling strategies, while the Chained formulation systematically models the relationship between resampling strategies and learning algorithms, recommending them in a sequential manner.

This chapter is organized as follows: Section 1.1 motivates our research and present the main contributions; Section 1.2 presents the objectives and hypotheses of this thesis; and Section 1.3 outlines the organization of this document.

1.1 MOTIVATION AND MAIN CONTRIBUTIONS

Imbalance problems frequently arise in various real-world applications such as healthcare (HE; MUNASINGHE, 2021), finance (RUDD; HUO; XU, 2022), and atmospheric phenomena (GHIMIRE et al., 2022; SALCEDO-SANZ et al., 2022). Most proposed studies for imbalanced regression focus on resampling strategies that alter the data distribution with the aim of balancing and are independent of the learning model. Although resampling strategies are applicable to any dataset and are independent of learning models, each problem tends to have its own preferred combination of resampling strategy and learning model. Furthermore, the learning model can be positively or negatively affected depending on the resampling strategy used. Given the existence of various resampling strategies and learning models, making this choice is not trivial. To address this issue, we propose a meta-learning-based model to recommend an appropriate resampling strategy and learning model for imbalanced regression datasets. To the best of our knowledge, while several studies have explored meta-learning in other scenarios (e.g., (MONIZ; CERQUEIRA, 2021; ROSSI et al., 2014; ROSSI et al., 2021; AMORIM; CAVALCANTI;

CRUZ, 2024; AGUIAR et al., 2019)), no existing proposals of meta-learning-based models have been developed specifically for imbalanced regression. This highlights the originality of our work, as it is the first to address this gap.

The main contributions of this work are:

- A comprehensive review of the main strategies used for imbalanced regression tasks with an extensive experimental study comparing the performance of state-of-the-art resampling strategies and their effects on multiple learning algorithms and novel performance metrics proposed in the literature.
- A new taxonomy for imbalanced regression tasks according to the regression model, learning strategy and metrics.
- An analysis of the impact of dataset characteristics (e.g., dataset size and the number of rare cases) on the model's predictive performance.
- A meta-learning method for recommending pipelines for imbalanced regression problems with two meta-learning recommendation approaches: Independent and Chained.
- An analysis of meta-feature importance, identifying key meta-features crucial for the meta-model's predictive ability in recommending learning models and resampling strategies for imbalanced regression problems.

In addition, the following two papers were written:

- AVELINO, J. G.; CAVALCANTI, G. D.; CRUZ, R. M. Resampling strategies for imbalanced regression: a survey and empirical analysis. Artificial Intelligence Review, Springer, v. 57, n. 4, p. 82, 2024
- AVELINO, J. G.; CAVALCANTI, G. D.; CRUZ, R. M. Imbalanced Regression Pipeline Recommendation. *Under review*

1.2 OBJECTIVE AND HYPOTHESES

The primary goal of this study is to introduce a meta-learning method for addressing imbalanced regression problems. The aim is to improve the process of selecting learning models and resampling strategies, which is currently complex and time-consuming. To achieve this, we

started by conducting an analysis of resampling strategies proposed in the existing literature for imbalanced regression. Our hypotheses and the corresponding chapters in which they are discussed are outlined below.

- Chapter 2:

    1. It is beneficial to use resampling strategies for improving predictive performance in imbalanced regression problems.
    2. Certain resampling strategies have a greater impact on predictive performance compared to others.
    3. The effectiveness of the best resampling strategy is influenced by the specific problem, the learning model employed, and the metrics used for evaluation.
    4. The number of training examples generated by each resampling strategy affects the predictive results.
    5. Data features such as the percentage of rare cases, number of rare cases, dataset size, number of attributes, and imbalance ratio have a significant impact on the predictive performance of the models.

- Chapter 3:

    1. Meta-learning can be used to recommend resampling strategies and learning models for imbalanced regression tasks.
    2. Employing specialized metrics and methods for imbalanced regression is crucial for improving the performance and reliability of machine learning models.
    3. The selection of a resampling strategy influences the choice of the learning model, and vice versa.
    4. Meta-IR is more effective for imbalanced regression tasks compared to AutoML methods.
    5. Resampling strategies improve the performance of AutoML methods.

1.3 THESIS ORGANIZATION

This thesis is structured around two papers written during the Ph.D. program. Firstly, we introduce the basic concepts in chapter 2. The papers compose Chapters 3 and 4 of the thesis,

presenting the core research and findings. Chapter 5 consolidates the principal conclusions drawn from these works, summarizing the insights gained throughout the research period.

- Chapter 2: **Basic Concepts**

  It presents the basic concepts to provide the foundation for understanding the proposed method, including meta-learning (MtL), algorithm selection problem (ASP), and automated machine learning (AutoML).

- Chapter 3: **Resampling strategies for imbalanced regression: a survey and empirical analysis**

  In this study, we performed a comprehensive experimental analysis on imbalanced regression tasks. We assessed different strategies for resampling and predictive models, and introduced a taxonomy based on regression models, learning processes, and evaluation metrics.

- Chapter 4: **Imbalanced Regression Pipeline Recommendation**

  This work introduces the Meta-learning for Imbalanced Regression (Meta-IR) framework, which utilizes meta-learning-based model to recommend optimal pipelines of resampling strategies and learning models for imbalanced regression tasks.

- Chapter 5: **Conclusion**

  We present the main findings of the work, discuss its limitations, and outline directions for future research.

2 BASIC CONCEPTS

In this section, we present the basic concepts applied in this thesis: Meta-learning (MtL), Algorithm Selection Problem (ASP) and Automated Machine Learning (AutoML). These concepts provide the foundation and context necessary for understanding the methodologies and approaches proposed.

2.1 META-LEARNING

Meta-learning is a field that focuses on learning from previous experiences, involving using metadata to adapt algorithms on new tasks more efficiently (BRAZDIL et al., 2022b). This approach encompasses techniques enabling machine learning systems to effectively adjust to new datasets and tasks (VANSCHOREN, 2018). Meta-learning systems can solve some types of problems. In Brazdil et al. (2022b), the types are presented; the main ones are Algorithm Selection (AS), Hyperparameter optimization (HPO), Combined algorithm selection and hyperparameter optimization (CASH), and Workflow synthesis (pipeline).

Algorithm selection involves identifying the most suitable algorithm for a given dataset from a set of available algorithms, while hyperparameter optimization focuses on fine-tuning the settings of an algorithm's hyperparameters to improve its performance. When combining these approaches, combined algorithm selection and hyperparameter optimization (CASH) tackles both selecting the best algorithm and optimizing its hyperparameters simultaneously. Furthermore, Workflow (pipeline) synthesis extends CASH by designing a series of steps or a pipeline of multiple algorithms to a dataset.

In this work, we proposed an algorithm selection method based on meta-learning to select the best resampling strategy and learning model for imbalanced regression problems. Therefore, the following section describes this topic, exploring methodologies and strategies for algorithm selection.

Algorithm Selection Problem

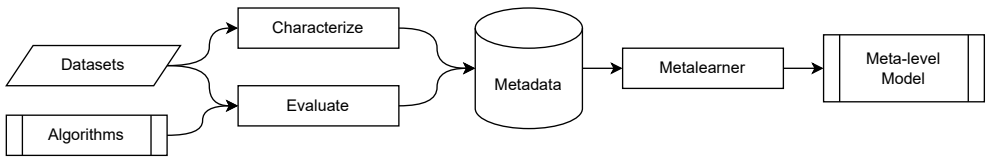
Initially formulated by Rice (1976), the Algorithm Selection Problem ASP can be described as the problem of selecting the most suitable algorithm to solve a given problem instance based

on its characteristics. This problem is crucial because the performance of machine learning algorithms can vary significantly across different datasets, and selecting the wrong algorithm can lead to suboptimal results.

Thus, Rice's formulation defines ASP in terms of four main components: problem space, algorithm space, feature space, and performance measure (RICE, 1976). The problem space is the data sets that need to be solved, which may include classification problems, regression problems, and time series problems. The algorithm space is the set of algorithms available to solve these problems. This includes a range of machine learning algorithms such as decision trees, support vector machines, neural networks, and others. The feature space is characterized by a set of characteristics that describe the problems. These characteristics, or features, may include statistical measures, data complexity measures, and other properties of the data set. Lastly, the performance measure maps a pair (problem, algorithm) with a performance value. The metric quantifies the effectiveness of the algorithm in solving the problem. With these components, metadata is built, where each example is represented by a set of characteristics and how each algorithm performed for the problem.

The algorithm selection problem is one of the central problems addressed by meta-learning. Figure 1 presents a general architecture for meta-learning systems that address the algorithm selection problem. First, the metadata is constructed, including the dataset characterization (**meta-features**), algorithms (e.g., machine learning pipelines), and performance information (**meta-target**). Finally, a **meta-learner** is generated from this data and can recommends algorithms for each dataset. The recommendation can be made in different ways, such as the best algorithm from a set, a subset of the top algorithms, a linear ranking, a nearly linear (weak) ranking, and an incomplete ranking for the user.

Figure 1 – A general architecture for meta-learning systems that address the algorithm selection problem.



Source: Adapted from Brazdil et al. (2022b).

In the following sections, the main meta-features used for regression problems (Section 2.1.1) and the recommendation methods (Section 2.1.2) are presented.

2.1.1 Dataset characterization

Characterizing the data is a crucial step in building a meta-learning model, as the recommendations rely on its effectiveness. There are various ways to extract features from the data. Brazdil et al. (2022a) presented a review of the main meta-features used in classification, regression, time series, and clustering tasks. Below, we present the commonly used meta-features in regression problems.

- **Simple and statistical:** These features are extracted directly from the datasets and are similar to those used in classification problems, except for those involving the target variable, which need to be adapted for the continuous target value. Simple meta-features include measures such as the number of examples, the number of attributes, the proportion of discrete attributes, the proportion of missing values, and the proportion of outliers. Statistical measures can include calculations such as skewness, kurtosis, correlation, and covariance of instances.

- **Complexity-based:** Various studies have utilized data complexity measures as meta-features. For classification problems, this category is investigated in (CAVALCANTI; REN; VALE, 2012; LEYVA; GONZÁLEZ; PEREZ, 2014; GARCIA; CARVALHO; LORENA, 2016; MORÁN-FERNÁNDEZ; BOLÓN-CANEDO; ALONSO-BETANZOS, 2017; GARCIA et al., 2018), and for regression in (LORENA et al., 2018). Complexity measures have also been adapted for imbalanced classification problems (BARELLA et al., 2018; BARELLA; GARCIA; CARVALHO, 2020; BARELLA et al., 2021).

- **Model-based:** This measure provides information obtained from learning models. The measures of mean absolute error and the variation of residuals from a linear regressor were described in Lorena et al. (2018) as complexity-based measures, but since they are derived from models, they can also be considered model-based measures.

- **Smoothness measures:** In this category, meta-features are based, for example, on estimating the similarity of examples in the output space using the idea of a minimum spanning tree to connect the most similar examples in the input space, weighting the edges with Euclidean distance, and then calculating the average distance between target values.

- **Non-linearity measures:** The meta-features in this category are adaptations of those defined for classification problems. The non-linearity of a linear or nearest neighbor regressor is calculated by considering two examples with similar outputs that are interpolated to generate a new test example. The linear regressor is then trained with the original data and tested with the new test set. The Mean Squared Error (MSE) obtained is used as a meta-feature.

2.1.2 Forms of Recommendation

According to Brazdil et al. (2022b), the system can recommend the best algorithm in a set, a subset of the top algorithms, a linear ranking, a quasi-linear (weak) ranking, and an incomplete ranking to the user. The best algorithm in a set is a single top-performing algorithm. Alternatively, it can suggest a subset of the top algorithms, offering a few high-performing options. Another type of recommendation is a linear ranking, where algorithms are ordered from best to worst. A quasi-linear (or weak) ranking provides a less strict ordering, indicating relative performance without a precise order, used when two or more algorithms are tied. Lastly, an incomplete ranking suggests a partial order of some top algorithms without necessarily including all algorithms in the set.

2.2 AUTOMATED MACHINE LEARNING (AUTOML)

Automated machine learning (AutoML) aims to automate the process of applying machine learning to real-world problems. The goal of AutoML methods is to make machine learning accessible to non-experts and to improve the efficiency and effectiveness of machine learning pipelines for experts (BAHRI et al., 2022; BARATCHI et al., 2024). These systems reduce the need for extensive human intervention and can significantly accelerate the development of machine learning models.

The AutoML systems can include data pre-processing, feature engineering, model selection, hyperparameter optimization, and model evaluation (HUTTER; KOTTHOFF; VANSCHOREN, 2019). The data pre-processing process involves cleaning the data, addressing missing values, encoding categorical variables, and normalizing numerical features. Feature engineering includes creating new features or modifying existing ones to enhance the performance of machine learning models. Model selection is the stage where the best machine learning algorithm

for the given task and dataset is determined. Hyperparameter tuning focuses on optimizing the hyperparameters of the chosen model to improve its performance, and model evaluation assesses the model's performance using appropriate metrics and validation techniques.

AutoML systems have three main components (HUTTER; KOTTHOFF; VANSCHOREN, 2019): 1) search space, 2) search strategy, and 3) performance evaluation:

1. **Search space**: The search space defines all possible solution choices. This can include a list of pre-processing strategies, learning models, and hyperparameters. The search strategy specifies the method that will be used to explore the search space and optimize the performance of the selected models.

2. **Search strategies**: Several search strategies are employed by AutoML models, including Bayesian optimization (HUTTER; HOOS; LEYTON-BROWN, 2011; SNOEK; LAROCHELLE; ADAMS, 2012; GARNETT, 2023), Evolutionary algorithms (BÄCK; FOGEL; MICHALEWICZ, 1997; SIMON, 2013), Gradient-based optimization (BENGIO, 2000), Random search (BERGSTRA; BENGIO, 2012), and Meta-Learning (BRAZDIL et al., 2022b). In AutoML models, meta-learning is often used to warm-start the search process by recommending good initial hyperparameter settings (BARATCHI et al., 2024). This approach leverages prior knowledge from similar tasks to speed up and improve the optimization process. For example, this method is utilized in tools like Auto-sklearn (FEURER et al., 2015).

3. **Performance evaluation**: The search strategies provide some candidate options, and it is necessary to evaluate the performance of these solutions. A commonly used method to address this situation involves dividing the dataset utilized by the AutoML system into a training set and a validation set. The search algorithm of the AutoML system trains any considered model on the training set and evaluates it on the validation set (BARATCHI et al., 2024). This process is typically carried out using a nested cross-validation technique (VARMA; SIMON, 2006).

3 RESAMPLING STRATEGIES FOR IMBALANCED REGRESSION: A SURVEY AND EMPIRICAL ANALYSIS

AUTHORS

Juscimara G. Avelino *Universidade Federal de Pernambuco, Recife, Brazil*
George D. C. Cavalcanti *Universidade Federal de Pernambuco, Recife, Brazil*
Rafael M. O. Cruz *École de technologie supérieure, Montreal, Canada*

ABSTRACT

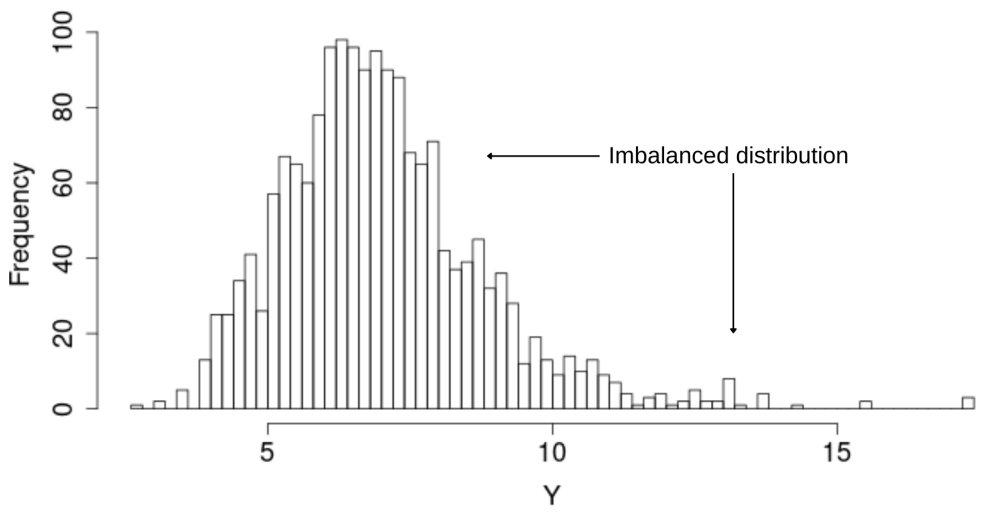
Imbalanced problems can arise in different real-world situations, and to address this, certain strategies in the form of resampling or balancing algorithms are proposed. This issue has largely been studied in the context of classification, and yet, the same problem features in regression tasks, where target values are continuous. This work presents an extensive experimental study comprising various balancing and predictive models, and wich uses metrics to capture important elements for the user and to evaluate the predictive model in an imbalanced regression data context. It also proposes a taxonomy for imbalanced regression approaches based on three crucial criteria: regression model, learning process, and evaluation metrics. The study offers new insights into the use of such strategies, highlighting the advantages they bring to each model's learning process, and indicating directions for further studies. The code, data and further information related to the experiments performed herein can be found on GitHub: <https://github.com/JusciAvelino/imbalancedRegression>.

3.1 INTRODUCTION

Imbalanced datasets are often encountered in multiple real-world applications. For classification tasks, such an issue has been studied (HAIXIANG et al., 2017; KRAWCZYK, 2016; JOHNSON; KHOSHGOFTAAR, 2019). Nonetheless, it is also present in regression tasks (BRANCO; RIBEIRO; TORGO, 2016). Branco, Torgo e Ribeiro (2017) define imbalanced problems based on the simultaneity of two factors: i) a disproportionate preference of the user at the domain of the target variable, and ii) insufficient representation of the data available in the most relevant cases for the user. In classification tasks, an imbalanced dataset is determined through the

presence of a class having a smaller representation (minority class) than another one (majority class). However, in regression problems, the target value is continuous, thus representing a complex definition, because the target value is not constrained to a limited set of discrete values, unlike in classification problems where the target value represents specific categories or classes. Figure 2 presents the distribution and frequency of examples drawn from an imbalanced dataset (FuelCons) with target values ranging from 2.7 to 17.3. To analyze this range, we employed a bin width of approximately 0.2, resulting in a total of 74 bins. The values at the chart's edges show little frequency and are considered rare examples. In this context, Ribeiro (2011) proposes the concept of a relevance function which determines the relevance of continuous target values in defining certain examples as rare and others as normal. This definition allows to verify an imbalanced between instances considered rare and those seen as normal.

Figure 2 – Distribution and frequency of the target value Y from the FuelCons dataset.



Source: Prepared by the author.

Standard regression tasks assume that all values of the domain are of equal importance, and are typically evaluated based on the performance of the most frequent values. However, values that are little represented are often extremely relevant, not only to the user, but also in the prediction process. For example, in the context of software engineering prediction mistakes in large projects are associated with higher development costs (RATHORE; KUMAR, 2017a), whereas during temperatures prediction in a meteorological application, errors that surface while predicting extreme conditions (e.g., very high temperatures) are even much more costly (RIBEIRO; MONIZ, 2020). This scenario presents particular difficulties for learning algo-

rithms, which tend to follow the interval of values in greater quantity while neglecting the rare ones in the distribution. Hence, failing to obtain a good prediction performance for these particular examples.

Studies looking at solutions for imbalanced regression problems have faced relatively little scrutiny when compared to those related to classification problems (HAIXIANG et al., 2017). The most common approach used to address this gap has been to modify the distribution of examples by balancing the training data before the actual learning process begins. Some of these strategies are Random Under-sampling (TORGO et al., 2013), which removes examples from intervals having greater quantities, Random Over-sampling (BRANCO; TORGO; RIBEIRO, 2019), which replicates rare values in the dataset, and the WEighted Relevance-based Combination Strategy (WERCS) (BRANCO; TORGO; RIBEIRO, 2019), which creates a weighted combination biased versions of the under- and over-sampling strategies. In addition, several real-world imbalanced regression problems rely on resampling strategies to properly deal with rare and extreme cases, such as in software defect prediction ((BAL; KUMAR, 2018), (BAL; KUMAR, 2020), (RATHORE; KUMAR, 2017a) and (RATHORE; KUMAR, 2017b)) and Enzyme Optimum Temperature prediction (GADO; BECKHAM; PAYNE, 2020), as well as to assist in detecting arsenic concentration in soil using satellite imagery (AGRAWAL; PETERSEN, 2021). Hence, the variety of problems and increased interest in this field demonstrates the need for studies on imbalanced regression techniques.

Another difficulty encountered in such scenarios is related to the fact that traditional performance metrics, such as the Mean Squared Error (MSE) and the Mean Absolute Error (MAE), do not adequately capture user-defined criteria (BRANCO; TORGO; RIBEIRO, 2019). Additionally, recent works have proposed new performance metrics for evaluating the performance of regression models under imbalanced target distributions, and place greater emphasis on errors occurring in rare cases. In these cases, Precision, Recall, and F1-score metrics, as described for regression tasks (TORGO; RIBEIRO, 2009), and the squared error-relevance area (SERA) metric proposed in Ribeiro e Moniz (2020), are commonly used. Nevertheless, a comparison between multiple imbalanced regression strategies under these performance metrics, and of how they differ in their approach to assessing the model's performance, is still an open question.

Therefore, our main goal is to analyze the effects of resampling strategies for dealing with imbalanced regression problems from different perspectives. To this end, we conduct an extensive experimental study employing different resampling strategies and learning algorithms. In addition, we use metrics that can assess the models' performance in imbalanced regression

tasks, such as the F1-score for regression and SERA (RIBEIRO; MONIZ, 2020). To the best of our knowledge, this is the first work that performs a comprehensive empirical analysis of resampling techniques for imbalanced regression tasks. In contrast, for imbalanced classification tasks, numerous surveys and empirical studies have evaluated resampling algorithms in different scenarios, such as binary problems (GARCÍA et al., 2020; KOVÁCS, 2019; WOJCIECHOWSKI; WILK, 2017; ROY et al., 2018; ALI et al., 2019; RIO; BENÍTEZ; HERRERA, 2015; DÍEZ-PASTOR et al., 2015; MONIZ; MONTEIRO, 2021), multiclass classification (CRUZ et al., 2019; SÁEZ; KRAWCZYK; WOŹNIAK, 2016), and data streams (AGUIAR; KRAWCZYK; CANO, 2022; ZYBLEWSKI; SABOURIN; WOŹNIAK, 2019).

The broad scope of our experimental analysis, which considers multiple resampling strategies, regression models, and performance metrics, is at the core of the uniqueness of our research since it allowed us to assess the relationship among these three variables. Our study thus differs from Branco, Torgo e Ribeiro (2016), which addresses only theoretical aspects of imbalanced problems in general. Moreover, regarding the performance metrics, using the SERA metric (RIBEIRO; MONIZ, 2020) is highlighted since no other work has evaluated all resampling strategies using it specifically.

The following research questions guide this study: i) Is it worth using resampling strategies? ii) Which resampling strategies influence predictive performance the most? iii) Does the choice of best strategy depend on the problem, the learning model, and the metrics used? iv) Does the number of training examples resulting from each strategy influence the results? v) Do the features of the data (percentage of rare cases, number of rare cases, dataset size, number of attribues and imbalance ratio) impact the predictive performance of the models? The experimental analysis revealed that resampling strategies are beneficial to the vast majority of regression models. The best strategies include Gaussian Noise Introduction, Random Oversampling and WERCS. Another important point is that choosing the best strategy depends on the dataset, the regression model, and the metric used when evaluating the system's performance. Furthermore, we found that the dataset size, the number of rare cases, the number of attribute and the imbalance ratio significantly influence the results. The smallest datasets and those with the fewest rare cases are the most challenging. Models demonstrate superior performance in datasets with fewer features. Lastly, concerning the imbalance ratio, regression models encounter more significant challenges with a higher imbalance ratio.

Contributions

- We propose a novel taxonomy for imbalanced regression tasks according to the regression model, learning strategy and metrics.
- We review the main strategies used for imbalanced regression tasks.
- We conduct an extensive experimental study comparing the performance of state-of-the-art resampling strategies and their effects on multiple learning algorithms and novel performance metrics proposed in the literature.
- We analyze the impact of dataset characteristics (e.g., dataset size and the number of rare cases) on the model's predictive performance.

This work is organized as follows: Section 3.2 presents the basic concepts and proposes a taxonomy for imbalanced regression problems. Section 3.3 describes the resampling approaches evaluated in this study highlighting their advantages and disadvantages. Section 3.4 presents the experimental methodology by describing the data, algorithms, parameters, and performance metrics used in this work. Results are shown in Section 3.5. Section 3.6 presents the lessons learned by revisiting and answering the research questions. Finally, Section 3.7 brings our conclusions.

3.2 BASIC CONCEPTS AND PROPOSED TAXONOMY

Some fundamental concepts must be grasped in order to understand the notion of imbalanced regression. In this context, the concept of relevance function is presented herein and a taxonomy is proposed to organize the strategies required. The relevance function is a fundamental concept in imbalanced regression, as it defines the importance of each sample in the dataset. Finally, a taxonomy is proposed to categorize the approaches used to address imbalanced regression problems, providing a way to understand the existing literature. Based on this taxonomy, we review the main strategies for dealing with imbalanced regression problems.

3.2.1 Relevance Function

The concept of relevance function is crucial when it comes to understanding the imbalanced regression problem and some strategies for dealing with it. Proposed by Ribeiro (2011), the relevance function ($\phi : Y \rightarrow [0, 1]$) determines the relevance of the examples in each dataset using an automatic method. The relevance value determines the examples that are normal and those that are rare, with the rare ones being the least represented in the dataset. The intuition of the relevance function is to automatically set the significance of data points within a dataset by assigning relevance scores. In this way, the relevance function serves as the foundation for evaluating models in the context of imbalanced regression, as well as for data resampling. Consequently, using a different relevance function alters both the model evaluation and data resampling.

To the best of our knowledge, this definition of relevance function is unique in the literature. In Ribeiro (2011) and Ribeiro e Moniz (2020), the relevance function is showcased using the Piecewise Cubic Hermite Interpolating Polynomials (pchip) and cubic spline methods. However, it was noted that cubic spline interpolation cannot provide precise control over the function. It fails to confine the relevance function within the specified [0, 1] interval scale. This limitation is rectified by the pchip method, employing suitable derivatives at control points, thereby ensuring properties like positivity, monotonicity, and convexity. Consequently, Ribeiro (2011) proposed relevance function utilizes the pchip method and aligned with this, the works in the field utilize this function.

The relevance function (ϕ) is calculated using Piecewise Cubic Hermite Interpolating Polynomials (pchip) (DOUGHERTY; EDELMAN; HYMAN, 1989) over a set of control points (Algorithm 1). The algorithm receives as input the control points (S) with their respective relevance values ($\varphi(y_k)$) and derivative ($\varphi'(y_k)$). The condition $y1 < y2 < ... < ys$ ensures that the data points are ordered in ascending order of their y-values. This ordering is fundamental for properly functioning the pchip algorithm. As a result, the algorithm produces a separate $\phi(y)$ polynomial for each interval $[y_k, y_{k+1}]$, with coefficients calculated based on the control points and their derivatives within that specific interval, where the variable k represents the index for the input set S control points.

Algorithm 1 pchip(S): Piecewise Cubic Hermite Interpolating Polynomials

Input:
$S = \{\langle y_k, \varphi(y_k), \varphi'(y_k)\rangle\}_{k=1}^{s}$, with $y_1 < y_2 < ... < y_s$, relevance values $\varphi(y_k)$ and preliminary derivative $\varphi'(y_k)$
Output:
$\phi(y)$: a Piecewise Cubic Hermite Interpolating Polynomial

for $k \leftarrow 1$ **to** $s-1$ **do**
  $h_k \leftarrow y_{k+1} - y_k$
  $\delta_k \leftarrow (\varphi(y_{k+1}) - \varphi(y_k))/h_k$
  $a_k \leftarrow \varphi(y_k)$
end for
$\{b_k\}_{k=1}^{s-1} \leftarrow$ *check_slopes* $(\{\varphi'(y_k)\}_{k=1}^{s-1}, \{\delta_k\}_{k=1}^{s-1})$ ▷ Monotone Cubic Spline
for $k \leftarrow 1$ **to** $s-1$ **do**
  $c_k \leftarrow (3\delta_k - 2b_k + b_{k+1})/h_k$
  $d_k \leftarrow (b_k - 2\delta_k + b_{k+1})/h_k^2$
end for
return $\phi(y) = a_k + b_k(y - y_k) + c_k(y - y_k)^2 + d_k(y - y_k)^3, y \in [y_k, y_{k+1}]$

The control points can be defined based on domain knowledge or provided by an automated method. When control points are defined based on domain knowledge, selecting them is guided by the expertise and understanding of the specific problem or dataset. This approach relies on the insights and experience of individuals familiar with the data and its context. Ideally, access to domain knowledge for defining control points would be preferred. However, this knowledge is often unavailable or nonexistent (RIBEIRO; MONIZ, 2020). Therefore, the utilization of an automatic method for control point definition becomes necessary. An example of defining control points of the NO2 emissions problem based on domain knowledge is presented in Table 1. Control points are determined based on Directive 2008/50/EC. The objective is to maintain the $LNO2$ (target) hourly concentration values below a limit equal to $ln(150\mu g/m^3) \approx 5.0$, indicating maximum relevance, and the annual average guideline of $ln(40\mu g/m^3) \approx 3.7$, indicating minimal relevance. And the lowest $LNO2$ concentration value $ln(3\mu g/m^3) \approx 1.1$ is attributed minimal relevance.

Table 1 – Control points of LNO2 concentration thresholds according to Directive 2008/50/EC.

y_k : $LNO2$ concentration values		$\phi(y_k)$	$\phi'(y_k)$
Low concentration:	$ln(3\mu g/m^3) \approx 1.1$	0.0	0.0
Annual mean guideline:	$ln(40\mu g/m^3) \approx 3.7$	0.0	0.0
Limit threshold:	$ln(150\mu g/m^3) \approx 5.0$	1.0	0.0

Source: Ribeiro e Moniz (2020).

In this work, we employ the automatic method, proposed by Ribeiro (2011), to define the control points. This method is based on Tukey's boxplot (TUKEY, 1970). The Tukey's boxplot is a graphical representation used to display the distribution of a dataset through its five summary statistics: The adjacent limits adj_L (Eq. 3.1) and adj_H (Eq. 3.2), first quartile ($Q1$), third quartile ($Q3$) and median $\tilde{Y}$ (Eq. 3.3). In turn, the control points are defined by the adjacent limits and the median value. The input to the pchip algorithm consists of control points, their relevance and derivatives. For this purpose, to the adjacent values (adj_L, adj_H) maximum relevance is assigned, which equals 1, and the median value ($\tilde{Y}$) with relevance value equal to zero. All control points are initialized with derivative $\phi'(y_k)$ equal to 0. In addition to defining the control points using Tukey's boxplot, Ribeiro e Moniz (2020) proposes the utilization of the adjusted boxplot, as proposed by Hubert e Vandervieren (2008).

$$adj_L = Q1 - 1.5 \cdot IQR \tag{3.1}$$

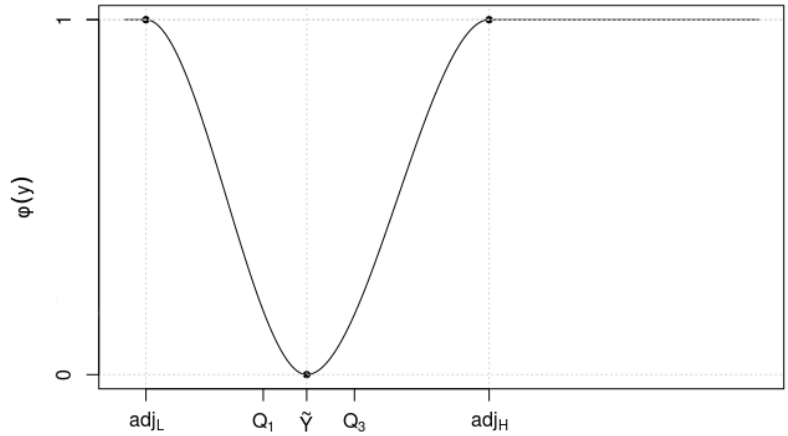
$$adj_H = Q3 + 1.5 \cdot IQR \tag{3.2}$$

$$\tilde{Y} = \text{median of } Y \tag{3.3}$$

where $Q1$ and $Q3$ are the first and third quartile, respectively, and $IQR = Q3 - Q1$.

Figure 3 illustrates the relevance function resulting from the pchip algorithm, for the fuelCons dataset. The points approaching $\tilde{Y}$ have negligible relevance, whereas points that move away from $\tilde{Y}$ and approach adj_L or adj_H have maximum relevance.

Figure 3 – Relevance function of the fuelCons dataset.



Source: Prepared by the author.

Algorithm 2 check_slopes (Φ, Δ) (FRITSCH; CARLSON, 1980)

Input:
$\Phi = \{\varphi'(y_k)\}_{k=1}^{s-1}$, $\Delta = \{\delta_k\}_{k=1}^{s-1}$

Output:
Φ: Modified derivative values

1: **for** $k \leftarrow 1$ **to** $s-1$ **do**
2: **if** $\delta_k = 0$ **then** ▷ Initialize the derivatives
3: $\varphi'(y_k) \leftarrow \varphi'(y_{k+1}) \leftarrow 0$
4: **else**
5: $\alpha \leftarrow \varphi'(y_k)/\delta_k$
6: $\beta \leftarrow \varphi'(y_{k+1})/\delta_k$
7: **if** $\varphi'(y_k) \neq 0 \wedge \alpha < 0$ **then** ▷ select an underset to preserve monotonicity
8: $\varphi'(y_k) \leftarrow -\varphi'(y_k)$
9: $\alpha \leftarrow \varphi'(y_k)/\delta_k$
10: **end if**
11: **if** $\varphi'(y_{k+1}) \neq 0 \wedge \beta < 0$ **then** ▷ select an underset to preserve monotonicity
12: $\varphi'(y_{k+1}) \leftarrow -\varphi'(y_{k+1})$
13: $\beta \leftarrow \varphi'(y_{k+1})/\delta_k$
14: **end if**
15: $\tau_1 \leftarrow 2\alpha + \beta - 3$
16: $\tau_2 \leftarrow \alpha + 2\beta - 3$
17: **if** $\tau_1 > 0 \wedge \tau_2 > 0 \wedge \alpha(\tau_1 + \tau_2) < \tau_1\tau_2$ **then** ▷ modifying the derivative values
18: $\tau \leftarrow 3\delta_k/\sqrt{\alpha^2 + \beta^2}$
19: $\varphi'(y_k) \leftarrow \alpha\tau$
20: $\varphi'(y_{k+1}) \leftarrow \beta\tau$
21: **end if**
22: **end if**
23: **end for**
24: **return** $\Phi = \{\varphi'(y_k)\}_{k=1}^{s-1}$

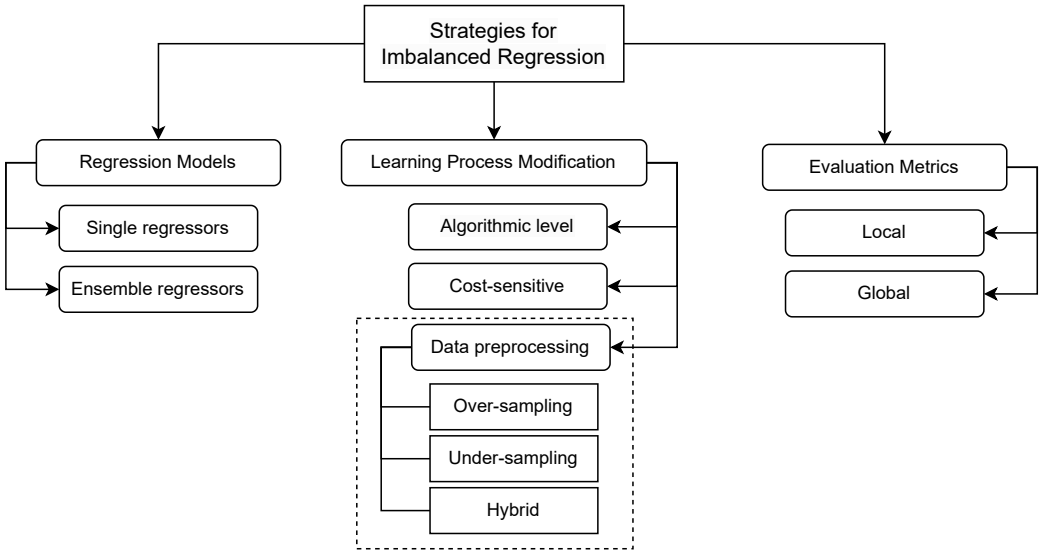
The interpolation generates a function that crosses the control points. One of the main goals is to learn the correct slopes in the data points such that the interpolant is monotonic by parts. To this end, a method that implements the Monotone Cubic Spline (FRITSCH; CARLSON, 1980) (line 6) is used. The check_slopes method (Algorithm 2) ensures that the derivative is zero when the control point for a maximum or minimum local (RIBEIRO; MONIZ, 2020).

A relevance threshold (t_R) defined by the user is employed to divide the data into rare (D_R) and normal (D_N) values. Given a dataset D, the sets D_R and D_N are defined considering the superior and inferior thresholds as follows: $D_R = \{\langle \mathbf{x}, y\rangle \in D : \phi(y) \geq t_R\}$ and $D_N = \{\langle \mathbf{x}, y\rangle \in D : \phi(y) < t_R\}$.

3.2.2 Proposed Taxonomy

In the context of class imbalanced problems, solutions are often classified into four groups: Algorithmic level, Cost-sensitive, Ensemble learning, and Data preprocessing (GALAR et al., 2011; LÓPEZ et al., 2013). However, one problem with this classification is that there is a significant overlap between the ensemble learning, data preprocessing, and cost-sensitive groups. Ensemble learning approaches can be used in conjunction with any other approaches by learning the base models, accounting to target imbalance at the algorithmic level, or applying data preprocessing prior to training each base model in the ensemble. Therefore, to better understand the different approaches for dealing with imbalanced regression problems, we can categorize the strategies into three main groups: i) Regression Models, ii) Learning Process Modification, and iii) Evaluation Metrics.

Figure 4 – Proposed taxonomy for imbalanced regression problems.



Source: Prepared by the author.

The first group of strategies comprises regression models, such as single models and ensembles, which can be used to address imbalanced regression problems. However, their performance can be further improved by incorporating data preprocessing, cost-sensitive learning, and algorithmic-level modifications. The second group describes these additional strategies which can help adjust the learning process to deal with the target imbalance, thus leading to better results when compared to using the models alone. The third group comprises the evaluation metrics and is divided into local and global subgroups. The local metrics require a relevance threshold to distinguish extreme values and conduct a local evaluation, and thus, cases with

a relevance score lower than the threshold are disregarded. Conversely, global metrics do not require a relevance threshold, making a global evaluation, considering all the examples. To conclude, categorizing these strategies into three groups can provide a better understanding of the approaches and enable the selection of the most suitable strategy for dealing with imbalanced regression problems. As shown in Figure 4, data preprocessing takes the spotlight, which is the main focus of this work. Herein, we explore and compare different data preprocessing techniques to improve the performance of regression models (single models and ensembles) in imbalanced regression problems.

Regression Models

Regression models such as MLPRegressor, Linear Support Vector Regression (SVR) and decision trees can be used to solve problems with imbalanced regression data, but they may not perform well due to the imbalance. In such cases, it may be necessary to utilize other techniques such as data preprocessing or cost-sensitive learning, or to modify the algorithm, to address the issue. In the same perspective, ensemble models, such as bagging, boosting, and random forest, can also be utilized in addressing these problems. Solutions based on ensemble learning combined with data preprocessing strategies and cost-sensitive were proposed. In Branco, Torgo e Ribeiro (2018) the REsampled BAGGing (REBAGG) model was proposed in a bid to integrate data resampling strategies with bagging, and had the advantage of generating a diverse set of models taking into account the different ways training data are resampled using the Random Under-sampling, Random Over-sampling and SmoteR strategies. SMOTEBoost (MONIZ et al., 2018) includes a resampling step when boosting, where SmoteR is used to direct the distribution of data towards rare cases. In the same context, Moniz, Branco e Torgo (2017) carried out a performance study of ensemble methods in regression tasks with imbalanced datasets.

Learning Process Modification

Learning Process Modification refers to the techniques used to modify the training process of machine learning algorithms to take into account rare cases. These techniques include algorithmic level modification, as well as the cost-sensitive and data preprocessing methods. At an algorithmic level, a model is introduced in Torgo e Ribeiro (2003) with new division criteria

for the regression trees that allow to induce trees at extreme and rare predicted values. Yang et al. (2021) proposed methods aimed at favoring the similarity between near targets by applying a kernel distribution to soften the distribution in the target and space of attributes. Ribeiro (2011) then addressed a utility-based algorithm involving cost-sensitive learning designed with a set of rules extracted from the generation of different regression trees aimed at obtaining accurate and interpretable predictions for imbalanced regression. Steininger et al. (2021) proposed a density-based weighting approach to address the issue of imbalanced regression, building on the cost-sensitive method. This approach assigns higher weights to rare cases by taking into account their local densities. Finally, one of the most common approaches for treating imbalanced issues is data preprocessing, also known as resampling or balancing algorithms, which precede the learning process, altering the examples distribution. The method works by either removing samples from common cases (i.e., under-sampling) or generating synthetic samples for rare events (i.e., oversampling). Data processing techniques have the advantage of allowing the use of just about any learning algorithm concurrently, without affecting the explicability of the model (BRANCO; TORGO; RIBEIRO, 2019).

Different resampling strategies have been proposed to deal with imbalanced regression problems. Most such techniques are based on existing resampling strategies proposed for classification problems. That is the case, for example, of the SmoteR algorithm, which is a variation of the Smote (CHAWLA et al., 2002) algorithm, with the following main adaptations made to adjust to the issue of regression: i) the definition of rare cases, ii) the creation of synthetic examples, and iii) the definition of target values for newly generated examples. Also on the basis of the Smote algorithm, Camacho, Douzas e Bacao (2022) proposed Geometric SMOTE, which generates synthetic data points along the line connecting two existing data points. Other strategies adapted from imbalanced classification are: Random Under-sampling (TORGO et al., 2013), based on the idea of Kubat, Matwin et al. (1997); Random Over-sampling (BRANCO; TORGO; RIBEIRO, 2019), proposed for the classification in Batista, Prati e Monard (2004), and Introduction of Gaussian Noise (BRANCO; TORGO; RIBEIRO, 2019), adapted from Lee (1999), Lee (2000). In contrast, the SMOGN (SmoteR with Gaussian Noise) (BRANCO; TORGO; RIBEIRO, 2017) and the WERCS (WEighted Relevance-based Combination Strategy) (BRANCO; TORGO; RIBEIRO, 2019) strategies were originally proposed for handling imbalanced regression problems. Furthermore, Song, Dao e Branco (2022) introduced a distributed version of the SMOGN called DistSMOGN. The method uses a weighted sampling technique to generate synthetic samples for rare cases, in addition to considering the data distribution in each node of the

distributed system. For the imbalanced data streams in regression models context, Aminian, Ribeiro e Gama (2021) introduced two sampling strategies (ChebyUS, ChebyOS) based on the Chebyshev inequality to improve the performance of existing regression methods on imbalanced data streams. The approaches use a weighted learning strategy that assigns higher weights to rare cases in order to balance the training process.

Each strategy resamples data differently. However, they appear to be based on the same principles: reducing normal examples and/or increasing rare examples. Under-sampling, which reduces normal examples, is the basis of the Random Under-sampling strategy. In contrast, over-sampling, which increases rare examples, can have a simple performance, as in Random Over-sampling, or by generating synthetic cases, as in the SmoteR Algorithm and Introduction of Gaussian Noise. Other strategies are based on the aforementioned models. Examples include the SmoteR with Gaussian Noise (SMOGN), which combines the Random Under-sampling strategy with the SmoteR and Introduction of Gaussian Noise over-sampling strategies. Also, the WEighted Relevance-based Combination Strategy (WERCS) combines the Random Under-sampling and Random Over-sampling strategies by using weights to perform the resampling without establishing a relevance threshold.

In our study, we analyze a variety of data preprocessing techniques to optimize the performance of single and ensemble regression models in addressing imbalanced regression problems. Our objective is to compare the effectiveness of different approaches in identifying the most suitable strategies for this situation. By carefully assessing these techniques, we aim to provide guidance as to how to increase the success rate of regression models using data preprocessing techniques in imbalanced regression tasks.

Evaluation Metrics

The choice of assessment metrics is fundamental in an imbalanced datasets scenario. Some metrics, such as the MSE, may fool users when the focus is on the accuracy of rare values of the target variable (MONIZ; TORGO; RODRIGUES, 2014) since it does not consider the relevance of each testing example. To show the limitations of the MSE metric and how the scores obtained by different metrics can significantly differ, we present a synthetic example (Table 2). For 10 examples in the *FuelCons* dataset, we present hypothetical predictions for two artificial models: M_1 and M_2. The True row represents the true target for each instance in the dataset, directly obtained from the FuelCons dataset. The ϕ row is the relevance value of each example.

Meanwhile, the M_1 and M_2 rows showcase predictions generated by the respective models for individual test examples. In parallel, the M_1 and M_2 loss rows quantify the differences between the true target and the predictions made by the models for each test example. The example shows that M_1 generates more accurate predictions for the less relevant examples, which are less represented in the dataset, while M_2 performs better for more relevant examples, which are more frequently represented. Nonetheless, if the models' performances are assessed using the MSE metric, there will be no difference in scores between them. This is because the MSE metric considers all examples as having the same relevance (ϕ). Therefore, for the imbalanced data scenario, where each example has a particular relevance, it is more interesting to use metrics that consider the relevance of each particular example.

Table 2 – Predictions of two artificial models.

	Test examples									
True	2.70	3.20	3.50	4.10	4.50	4.70	5.20	5.70	9.20	17.30
ϕ	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.02**	**0.57**	**1.00**	**1.00**	**1.00**
M_1	2.66	3.14	3.40	3.80	4.00	3.80	4.10	4.40	7.70	15.50
M_1 Loss	**0.04**	**0.06**	**0.10**	**0.30**	**0.50**	**0.90**	**1.10**	**1.30**	**1.50**	**1.80**
M_2	0.90	1.70	2.20	3.00	3.60	4.20	4.90	5.60	9.14	17.26
M_2 Loss	**1.80**	**1.50**	**1.30**	**1.10**	**0.90**	**0.50**	**0.30**	**0.10**	**0.06**	**0.04**

Source: Prepared by the author.

True - Target values
ϕ - Relevance values
M1 and M2 - Model predictions
M1 Loss and M2 Loss - Prediction errors

Other metrics consider each example as having a particular relevance score, such as Precision, Recall, and the F1-score, which were proposed for regression applications in Torgo e Ribeiro (2009). In addition, the Squared error-relevance area (SERA) metric, which was specifically created for imbalanced regression, was proposed by Ribeiro e Moniz (2020). This metric aims to effectively assess the model's performance for predictions of extreme values while being robust to model bias. Table 3 presents the MSE, F1-score, and SERA values for the example presented in Table 2. As earlier mentioned, for the MSE, the models are regarded as equals since they both have the same error amplitude. Nonetheless, for the F1-score and SERA, which consider each example's relevance, M_2 is the best model as it presents a lower error in the most important examples.

The Precision, Recall, and F1-score metrics require that a relevance threshold be defined

Table 3 – Performances of two artificial models

	Estimated Performance		
	MSE	**F1-score**	**SERA**
M_1	0.955	0.598	7.885
M_2	0.955	0.983	0.076

Source: Prepared by the author.

to determine extreme values. Thus, a local evaluation is performed, since examples below the threshold are ignored. Furthermore, these metrics use the concept of a utility-based framework (TORGO; RIBEIRO, 2007), (RIBEIRO, 2011). Such a structure uses the numeric error of the prediction and the relevance of the actual and predicted values. The utility of predicting a value $\hat{y}$ for y is calculated from the notions of costs and benefits of numeric predictions (BRANCO; TORGO; RIBEIRO, 2019), and thus, the utility function $U^p_\phi(\hat{y}, y)$ is given by Equation 3.4, where $\hat{y}$ is the predicted value and y is the actual value.

$$U^p_\phi(\hat{y}, y) = B_\phi(\hat{y}, y) - C^p_\phi(\hat{y}, y) \ = \phi(y) \cdot (1 - \Gamma_B(\hat{y}, y)) - \phi^p(\hat{y}, y) \cdot \Gamma_C(\hat{y}, y) \tag{3.4}$$

The utility is given by the difference between the prediction benefit ($B_\phi(\hat{y}, y)$) and cost ($C^p_\phi(\hat{y}, y)$) of prediction $\hat{y}$ for y. The benefit is defined as a proportion of the relevance of the actual value according to the following equation: $\phi(y) \cdot (1 - \Gamma_B(\hat{y}, y))$, where $\Gamma_B(\hat{y}, y)$ is the bounded loss function (Equation 3.5). This equation defines a loss function, $\Gamma_B(\hat{y}, y)$, which quantifies the loss incurred when making a prediction $\hat{y}$ for the actual value y (Equation 3.6). This loss function operates on a scale from 0 to 1, where 0 represents no loss, and 1 represents maximum loss.

$$\Gamma_B(\hat{y}, y)) = \begin{cases} L(\hat{y}, y)/\dot{L}_B(\hat{y}, y), & \text{if } L(\hat{y}, y) < \dot{L}_B(\hat{y}, y) \\ 1, & \text{if } L(\hat{y}, y) \geq \dot{L}_B(\hat{y}, y) \end{cases} \tag{3.5}$$

L is a “standard" loss function (e.g., absolute deviation (Equation 3.6)) and $\dot{L}_B$ is the benefit threshold function, (Equation 3.7). The benefit threshold function identifies the point at which the predicted value ceases to provide a benefit. This can happen because of two conditions: (i) surpassing the maximum acceptable loss of the bump or (ii) being situated on a different bump (RIBEIRO, 2011).

$$L(\hat{y}, y) = |\hat{y} - y| \tag{3.6}$$

$$\dot{L}_B(\hat{y}, y) = min\{b^{\Delta}_{\gamma(y)}, \ddot{L}_B(\hat{y}, y)\} \tag{3.7}$$

where $b^{\Delta}_{\gamma(y)}$ is the maximum admissible loss, defined in Equation 3.8. The maximum admissible loss is calculated for each bump i. A bump refers to a interval of the domain, denoted as $B \subseteq Y$ (RIBEIRO, 2011). b^{-} is the mean value at which the target variable reaches the minimum relevance before reaching its maximum value, and b^{*} is the mean value at which the target variable reaches the maximum relevance. The reason for this definition is that this function is contingent upon the smallest discrepancy concerning the target variable when transitioning from the most pertinent value within a bump (b^{*}_{i}) to an alternative bump. The smallest differences regarding the target variable can have two effects on model performance. On the positive side, it can make the model more accurate by focusing on the areas where predictions must be very close to the actual values. This is useful when you need high accuracy in specific parts of the data. Conversely, the model might become too fixated on the training data, making it sensitive to unusual data points and not very good at handling new data, leading to overfitting. Consequently, this implies that when dealing with “narrow" bumps, our sensitivity to prediction errors is heightened, whereas for broader bumps, we are more inclined to deem larger disparities between the actual and forecasted values as acceptable (RIBEIRO, 2011).

$$b^{\Delta}_{\gamma(y)} = 2 \cdot \text{min}\{\mid b^{-}_{i} - b^{*}_{i} \mid, \mid b^{*}_{i} - b^{-}_{i+1} \mid\} \tag{3.8}$$

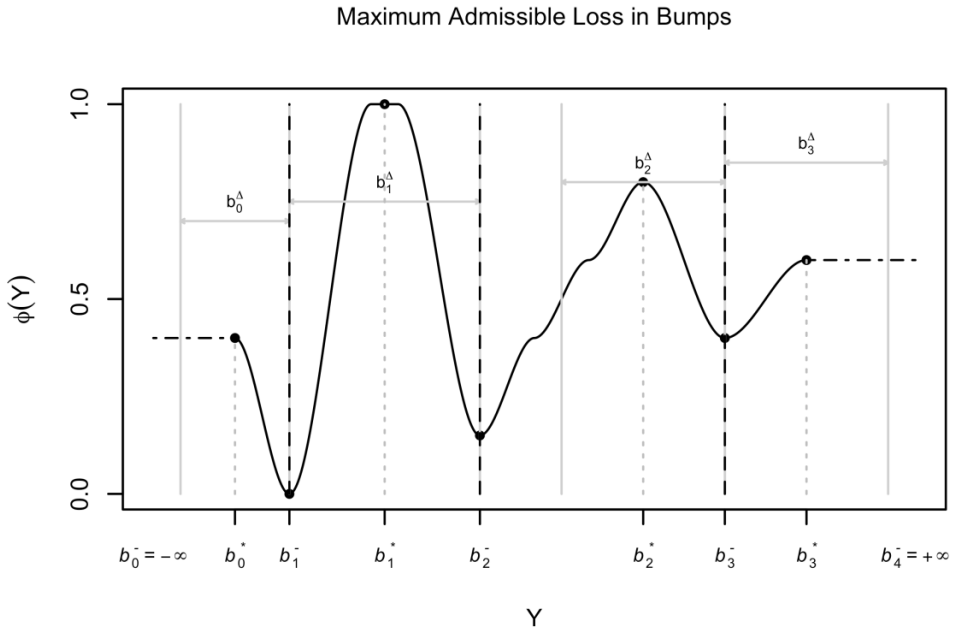
Figure 5 shows the bump partition obtained for a relevance function and the maximum admissible loss for each bump. This arbitrary relevance function, defined in the context of non-uniform utility regression, has four quite different bumps.

And $\ddot{L}_B(\hat{y}, y))$ (Equation 3.9) is defined as follows:

$$\ddot{L}_B(\hat{y}, y)) = \begin{cases} \mid y - b^{-}_{\gamma(y)} \mid, & \text{if } \hat{y} < y) \\ \mid y - b^{-}_{\gamma(y)+1} \mid, & \text{if } \hat{y} \geq y) \end{cases} \tag{3.9}$$

This definition satisfies two essential conditions: (1) The initial component within the min function addresses the maximum allowable error range within the true value's context, guaranteeing a level of reasonable accuracy in the prediction; (2) The subsequent component

Figure 5 – Bumps partition of Y with respect to relevance function ϕ and the maximum admissible loss in bumps. Each bump i is characterized by its partition node b^- and by one global maximum b^*. Each bump has a maximum error tolerance defined by the double of the smalles amplitude in the bump between each of one of its bounds and its maximum value.



Source: Ribeiro (2011).

within the min function evaluates whether the predicted value aligns with the correct action by considering its proximity to the boundaries of the context associated with the true value.

The cost is given by the mean of weighted relevance ($\phi^p(\hat{y}, y)$) (Equation 3.10), where the parameter p is used to define the weights between the two relevances and $\Gamma_C(\hat{y}, y)$ is the bounded loss function in the scale [0;1]. This equation calculates the weighted relevance of the predicted value $\hat{y}$ and the actual value y. The parameter p defines the weights between these two relevances. The intuition here is to balance the predicted value's importance and the utility function's actual value.

$$\phi^p(\hat{y}, y) = (1 - p)\phi(\hat{y}) + p\phi(y) \tag{3.10}$$

The cost function $\Gamma_C(\hat{y}, y)$ is calculated according to Equation 3.11.

$$\Gamma_C(\hat{y}, y)) = \begin{cases} L(\hat{y}, y)/\dot{L}_C(\hat{y}, y), & \text{if } L(\hat{y}, y) < \dot{L}_C(\hat{y}, y) \\ 1, & \text{if } L(\hat{y}, y) \geq \dot{L}_C(\hat{y}, y) \end{cases} \tag{3.11}$$

where L is the standard loss function, and $\dot{L}_C$ is the cost threshold function (Equation 3.12):

$$\dot{L}_C(\hat{y}, y) = min\{b^{\Delta}_{\gamma(y)}, \ddot{L}_C(\hat{y}, y)\} \tag{3.12}$$

and $\ddot{L}_C(\hat{y}, y))$ is defined as follows:

$$\ddot{L}_C(\hat{y}, y)) = \begin{cases} \mid y - b^*_{\gamma(y)-1} \mid, & \text{if } \hat{y} < y) \\ \mid y - b^*_{\gamma(y)+1} \mid, & \text{if } \hat{y} \geq y) \end{cases} \tag{3.13}$$

Captured using the utility function, the Precision and Recall metrics are defined by Equations 3.14 and 3.15, respectively.

$$Precision = \frac{\sum_{\phi(\hat{y}_i)>t_R}(1 + U^p_\phi(\hat{y}_i, y_i))}{\sum_{\phi(\hat{y}_i)>t_R}(1 + \phi(\hat{y}_i))} \tag{3.14}$$

$$Recall = \frac{\sum_{\phi(y_i)>t_R}(1 + U^p_\phi(\hat{y}_i, y_i))}{\sum_{\phi(y_i)>t_R}(1 + \phi(y_i))} \tag{3.15}$$

The relevance of the actual value y_i is defined by $\phi(y_i)$, as defined in Section 3.2.1, and $\phi(\hat{y}_i)$ is the relevance of the predicted value $\hat{y}_i$. t_R is a threshold defined by the user for the relevance values, and $U^p_\phi(\hat{y}_i, y_i)$ is the utility function previously described.

The Precision and Recall metrics can be aggregated in compound measures, such as F1-score, defined by Equation 3.16:

$$\textit{F1-score} = \frac{(\beta^2 + 1) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{3.16}$$

where $0 \leq \beta \leq 1$ controls the relative importance of the Recall for the Precision. These compound measures have the advantage of allowing comparisons between models by providing a single score (TORGO; RIBEIRO, 2009).

These metrics require the definition of an ad-hoc relevance threshold and do not consider examples below the threshold for model evaluation (RIBEIRO; MONIZ, 2020). To address this, Ribeiro e Moniz (2020) proposed the SERA metric.

SERA metric can assess models' efficacy and optimize them for predicting rare and extreme cases. This metric does not require a definition of a relevance threshold and thus performs a global evaluation since all data points are considered. The Squared error-relevance is obtained in relation to a cutting t achieved based on a relevance function $\phi : Y \rightarrow [0, 1]$. A subset $D^t = \{\langle \mathbf{x}, y\rangle \in D : \phi(y) \geq t\}$ formed based on the cutting t is considered for this estimate, such as in Equation 3.17:

$$SER_t = \sum_{i \in D^t} (\hat{y}_i - y_i)^2 \tag{3.17}$$

The Squared error-relevance area (SERA) represents the area below the curve SER_t, obtained through integration presented in Equation 3.18:

$$SERA = \int_0^1 SER_t \, dt = \int_0^1 \sum_{i \in D^t} (\hat{y}_i - y_i)^2 \, dt \tag{3.18}$$

The SER_t curve offers a broad view of prediction errors in the domain at various relevance cutoff values. Therefore, a smaller area under the curve (SERA) indicates a better model. It is noteworthy that assuming uniform preferences with $\phi(y) = 1$, SERA is comparable with the sum of squared errors.

3.3 RESAMPLING STRATEGIES

The most common way to deal with imbalanced datasets is to use resampling strategies changing the data distribution to balance the targets (MONIZ; BRANCO; TORGO, 2017). Such strategies are concentrated on the following three main approaches: i) over-sampling, ii) under-sampling, and iii) a combination of these two approaches. In over-sampling, rare cases are generated to compensate for the imbalanced distribution. The Random Over-sampling technique (BRANCO; TORGO; RIBEIRO, 2019) is an example of such a technique, which works by replicating rare cases prior to training. However, it is also possible to perform over-sampling by generating synthetic cases, as in the SmoteR (TORGO et al., 2013) and Introduction of Gaussian Noise strategies (BRANCO; TORGO; RIBEIRO, 2019).

Conversely, under-sampling techniques aim to exclude larger quantity data (i.e., normal examples). The Random Under-sampling algorithm (TORGO et al., 2013) uses this notion. Some strategies employ a combination of approaches, such as the SmoteR and Introduction of Gaussian Noise, which generates synthetic cases and uses under-sampling, WEighted Relevance based Combination Strategy (BRANCO; TORGO; RIBEIRO, 2019) , thus combining the approaches of under-sampling and over-sampling. The SMOGN (BRANCO; TORGO; RIBEIRO, 2017) uses the generation of synthetic cases with SmoteR and GN and under-sampling.

Sections 3.3.1-3.3.6 provide an overview of the resampling strategies evaluated in this work. These strategies were selected based on their wide adoption in the literature. Conversely, other strategies were disregarded due to an absence of publicly available source code for them, limited

reproducibility, and infrequent utilization by researchers for diverse problem domains. Finally, Section 3.3.7 critically analyzes the resampling strategies with a visual example.

3.3.1 SmoteR

The SMOTE for regression (SmoteR) algorithm was proposed in Torgo et al. (2013) (Algorithm 3). Like the other methods addressing imbalanced regression issues, it requires a relevance function ($\phi(y)$) and a relevance threshold (t_R). The relevant or unimportant examples are defined from such a function. The algorithm removes the least relevant examples (lines 4 to 7), which are considered “normal", and then generates synthetic examples based on the most relevant examples (line 8). The generation process basically follows the idea in the SMOTE, namely, first selecting one rare case from the dataset as the seed case and one of its K-Nearest Neighbors to generate a new data point between the reference and its selected neighbor. Algorithm 4 presents the procedure for generating the synthetic cases using SmoteR. First the number of synthetic examples that is generated from a selected rare case, ng, is determined based on the percentage of over-sampling o determined by the user and the dataset cardinality $|D|$ (line 3). Then, for each rare case c that will be used as a reference in the generation process, its K-Nearest Neighbors are computed (Line 5) nns. After the set of neighbors are obtained, the algorithms execute multiple iterations to generate ng synthetic examples by picking one of the examples in the nns set at random and interpolating with the reference one. This generation process is presented from lines 8 to 15, which show how attribute values for the synthetic case are generated. If the attributes are numeric, the difference between the attributes of the two seed cases is calculated (line 10). Subsequently, (line 11) multiplies this difference by a random number between 0 and 1, and then adds to the example's attribute. Otherwise, a random selection between the values of the seed cases is performed. On lines 16 to 18, the value of the target is generated, calculated by the weighted average of the two cases. The weights are obtained by the distance between the new case and the two seed cases (lines 16 and 17). In Branco (2018), this strategy is extended, and is able to handle any number of either normal or rare cases.

Algorithm 3 SmoteR

Input:

D - original dataset
t_R - relevance threshold
o - percentage of over-sampling
u - percentage of under-sampling
k - the number of neighbors used in case generation

Output:

$newD$ - a new modified dataset

1: $Bins_R \leftarrow \{D_i \in D : \forall (x, y) \in D_i, \phi(y) \geq t_R\}$ ▷ partitions into relevant bins
2: $Bins_N \leftarrow D \backslash Bins_R$ ▷ partitions into normal bins
3: $newD \leftarrow Bins_R$
4: **for each** $B \in Bins_N$ **do**
5: $selNormCases \leftarrow$ random sample of $u \times |B|$ cases of B ▷ under-sampling procedure
6: $newD \leftarrow newD \bigcup selNormCases$
7: **end for**
8: $newCases \leftarrow GENSYNTHCASES(Bins_R, o, k)$ ▷ Generation of the attribute values
9: $newD \leftarrow newD \bigcup newCases$
10: **return** $newD$

Algorithm 4 Generating synthetic cases

Input:
D - original dataset
o - percentage of over-sampling
k - the number of neighbors used in case generation

1: **function** GENSYNTHCASES(D, o, k)
2: $newCases \leftarrow \{\}$
3: $ng \leftarrow (o-1) \times |D|$ ▷ number of new cases to generate for each existing case
4: **for all** $case \in D$ **do**
5: $nns \leftarrow kNN(k, case, D_R \backslash \{case\})$ ▷ k-Nearest Neighbours of case
6: **for** $i \leftarrow 1$ to ng **do**
7: $x \leftarrow$ randomly choose one of the nns
8: **for each** $a \in$ attributes **do** ▷ generation of the attribute values
9: **if** ISNUMERIC(a) **then**
10: $diff \leftarrow case[a] - x[a]$
11: $new[a] \leftarrow case[a] + RANDOM(0,1) \times diff$
12: **else**
13: $new[a] \leftarrow$ randomly select among case$[a]$ and $x[a]$
14: **end if**
15: **end for** ▷ generation of the target value
16: $d_1 \leftarrow DIST(new, case)$
17: $d_2 \leftarrow DIST(new, x)$
18: $new[y] \leftarrow \frac{d_2 \times case[y] + d_1 \times x[y]}{d_1 + d_2}$
19: **end for**
20: $newCases \leftarrow newCases \bigcup new$ ▷ add the new synthetic case
21: **end for**
22: **return** $newCases$
23: **end function**

3.3.2 Random Over-sampling

The Random over-sampling (BRANCO; TORGO; RIBEIRO, 2019) strategy, presented in Algorithm 5, works by first selecting the examples that are above the relevance threshold t_R (line 2) as candidates to be duplicated, $Bins_R$. Then, for each bin B belonging to the rare examples $Bins_R$, the number of replicas $tgtNr$ generated is defined according to its cardinality $|B|$ and the oversampling percentage o (Line 4). The $|B|$ represents the number of elements (data points or examples) contained within that specific bin B. This oversampling percentage is a hyperparameter defined by the user. Random sampling is performed on line 5, and the duplicated cases are added to the new dataset ($newD$) on line 6. When performing this algorithm, no special treatment is required to generate the target values. As the examples generated are identical to the existing rare cases, the duplicated ones have exactly the same target value.

Algorithm 5 Random over-sampling

Input:
D - original dataset
t_R - relevance threshold
o - percentage of over-sampling
Output:
$newD$ - a new modified dataset

1: $newD \leftarrow D$
2: $Bins_R \leftarrow \{D_i \in D : \forall (x,y) \in D_i, \phi(y) \geq t_R\}$
3: **for each** $B \in Bins_R$ **do**
4: $\quad tgtNr \leftarrow o \times |B|$ ▷ number of replicas to be added
5: $\quad selCases \leftarrow$ sample randomly $tgtNr$ elements from B
6: $\quad newD \leftarrow newD \bigcup selCases$ ▷ add the replicas to the new data
7: **end for**
8: **return** $newD$

Algorithm 6 Random under-sampling

Input:
D - original dataset
t_R - relevance threshold
u - percentage of under-sampling
Output:
$newD$ - a new modified dataset

1: $Bins_R \leftarrow \{D_i \in D : \forall (x,y) \in D_i, \phi(y) \geq t_R\}$
2: $Bins_N \leftarrow D \backslash Bins_R$
3: $newD \leftarrow Bins_R$
4: **for each** $B \in Bins_N$ **do**
5: $\quad tgtNr \leftarrow u \times |B|$ ▷ number of replicas to be removed
6: $\quad NormCases \leftarrow$ randomly under-sample $tgtNr$ elements from B ▷ remove the examples from B
7: $\quad newD \leftarrow newD \bigcup NormCases$
8: **end for**
9: **return** $newD$

3.3.3 Random Under-sampling

The Random Under-Sampling strategy (Algorithm 6) was proposed by Torgo et al. (2013). In this approach, the under-sampling is performed by first using the relevance function (Section 3.2.1) and a relevance threshold t_R to define the rare cases in the dataset (line 1). The examples below t_R are considered normal, being candidates to be removed from the final dataset (BRANCO; RIBEIRO; TORGO, 2016) (line 2), while rare cases are kept. The removal of

the normal examples is thus performed according to an under-sampling rate provided by the user u, which defines the percentage of under-sampling applied in the dataset. For each bin B belonging to the set of normal examples $Bins_N$, the number of examples removed from it is computed based on its cardinality and the percentage of undersampling u (Line 5). Line 6 performs the under-sampling in B by randomly selecting data points to be removed, resulting in a reduced set that is used to compose the final dataset $newD$.

3.3.4 Introduction of Gaussian Noise

Generating synthetic examples through Gaussian noise (Introduction of Gaussian Noise - GN) constitutes an adaptation of the method proposed in Lee (1999), Lee (2000) for classification tasks to the regression context. Algorithm 7 presents the GN technique. It starts by dividing the dataset into normal cases $Bins_N$ and rare cases $Bins_R$ according to the relevance function $\phi(y)$ and the relevance threshold t_R (Lines 1 and 2). Examples belonging to $Bins_N$ (i.e., normal examples) are reduced in size, using the Random under-sampling technique (lines 4 to 6). The amount of reduction is controlled by the percentage of the under-sampling hyperparameter u defined by the user.

From lines 8 to 20, the over-sampling procedure is performed using the samples in $Bins_R$. For each seed case selected and used in the generation process, a total of ng new artificial generated examples are added to the dataset. ng is computed based on the percentage of the overs-sampling hyperparameter o and the number of examples in the corresponding set $B \in Bins_R$ (Line 9). The artificial cases are generated by introducing a small perturbation on both the attributes and the target variable value of the seed case. If the attributes are nominal (line 13), the generation is performed with probability proportional to the frequency of the values found in the category (lines 14 and 15). Otherwise, for the numeric attributes, a random perturbation from a normal distribution is added, as indicated on lines 17 and 18, where δ is the perturbation amplitude defined by the user and $sd(a)$ is the standard deviation of the attribute a estimated using the examples in the category. The normal perturbation is also applied to the seed target value in order to generate the target value of the newly generated example.

Algorithm 7 Introduction of Gaussian Noise

Input:
D - original dataset
t_R - relevance threshold
u - percentage of under-sampling
o - percentage of over-sampling
δ - perturbation amplitude
Output:
$newD$ - a new modified dataset

1: $Bins_N \leftarrow \{D_i \in D : \forall (x,y) \in D_i, \phi(y) < t_R\}$
2: $Bins_R \leftarrow \{D_i \in D : \forall (x,y) \in D_i, \phi(y) \geq t_R\}$
3: $newD \leftarrow Bins_R$
4: **for each** $B \in Bins_N$ **do**
5: $selNormCases \leftarrow$ random sample of $u \times |B|$ elements from B
6: $newD \leftarrow newD \bigcup selNormCases$
7: **end for**
8: **for each** $B \in Bins_R$ **do** ▷ over-sampling procedure
9: $ng \leftarrow o \times |B|$ ▷ number of synthetic examples for each case in *B*
10: **for each** $case \in B$ **do** ▷ generate synthetic examples
11: **for** $i \leftarrow 1$ **to** ng **do**
12: **for each** $a \in Attrs \bigcup Y$ **do**
13: **if** a *is nominal* **then**
14: $probs \leftarrow$ frequency of possible values of a
15: $new[a] \leftarrow$ sample a value from the values of a with weights $= probs$
16: **else**
17: $new[a] \leftarrow$ multiply $case[a]$ with a random sample from
18: $N(0, \delta \cdot sd(a))$
19: **end if**
20: **end for**
21: $newD \leftarrow newD \bigcup \{new\}$ ▷ add synthetic case to newD
22: **end for**
23: **end for**
24: **end for**
25: **return** $newD$

3.3.5 SmoteR with Gaussian Noise

The SmoteR with Gaussian Noise (SMOGN - SG) (BRANCO; TORGO; RIBEIRO, 2017) (Algorithm 8) combines the Random under-sampling strategy (lines 6 to 9) with two over-sampling strategies: SmoteR and Introduction of Gaussian Noise. The goal is to limit the potential risks to the SmoteR of generating bad examples when the seed and its selected neighbor are not close enough by using the more conservative strategy of just introducing Gaussian noise to generate new cases. These bad examples may not represent of the underlying data distribution

and can introduce several issues like noise, bias, or inconsistencies into the dataset. Moreover, the technique aims to allow for an increase in diversity when generating examples, which is not feasible by using only the Introduction of Gaussian Noise method (BRANCO; TORGO; RIBEIRO, 2017). Increasing diversity means producing a comprehensive range of examples covering different data distribution aspects. The generated examples should not be overly similar or redundant. Instead, they should capture different patterns, variations, or scenarios present in the data to represent the data distribution comprehensively. Thus, SMOGN addresses the main drawbacks of SmoteR and the introduction of Gaussian noise techniques.

Line 11 determines the number of synthetic cases ng that will be generated according to the percentage of the over-sampling hyperparameter o and the number of existing cases in the corresponding bin B. Then, for each seed case in B, its K-Nearest Neighbors and the maximum allowed distance to generate new cases with SmoteR are computed (lines 13 to 15). When the seed case and the selected neighbor are “sufficiently near" (i.e., distance below the computed threshold $maxD$), the SMOGN generates new synthetic examples with the SmoteR (lines 17 and 18) technique. Otherwise, it uses the Introduction of Gaussian Noise method when the distance between the two examples is higher than the estimated threshold (lines 20 and 21). The generated data points are then added to the new dataset, $newD$.

Algorithm 8 SMOGN

Input:
D - original dataset
t_R - relevance threshold
u - percentage of under-sampling
o - percentage of over-sampling
k - number of nearest neighbors
$dist$ - distance metric
Output:
$newD$ - a new modified dataset

$OrdD \leftarrow D$ order D by ascending value of Y
$\phi() \leftarrow$ relevance function
$Bins_N \leftarrow$ partitions of consecutive examples $\langle x_i, y_i \rangle \in OrdD$, such that $\phi(y_i) < t_R$
$Bins_R \leftarrow$ partitions of consecutive examples $\langle x_i, y_i \rangle \in OrdD$, such that $\phi(y_i) \geq t_R$
$newD \leftarrow Bins_R$
for each $B \in Bins_N$ **do** ▷ under-sampling procedure
    $selNormCases \leftarrow$ randomly sample $u \times |B|$ cases from B
    $newD \leftarrow newD \bigcup selNormCases$
end for
for each $B \in Bins_R$ **do** ▷ over-sampling procedure
    $ng \leftarrow o \times |B|$ ▷ number of synthetic examples for each case in B
    for each $case \in B$ **do** ▷ generate synthetic examples
        $nns \leftarrow kNN(k, case, dist)$ ▷ K-Nearest Neighbors of case
        $DistM \leftarrow$ distances between the case and the examples in B
        $maxD \leftarrow median(DistM)/2$
        for $i \leftarrow 1$ **to** ng **do** $x \leftarrow$ randomly choose one of the nns
            if $DistM(x) < maxD$ **then** ▷ safe kNN selected
                $new \leftarrow$ use SmoteR to interpolate x and $case$
            else ▷ non-safe kNN selected
                $pert \leftarrow min(maxD, 0.02)$
                $new \leftarrow$ introduce Gaussian Noise in $case$ with a perturbation $pert$
            end if
            $newD \leftarrow newD \bigcup new$ ▷ add synthetic case to newD
        end for
    end for
end for
return $newD$

3.3.6 WEighted Relevance based Combination Strategy

The WEighted Relevance-based Combination Strategy (WERCS) strategy (BRANCO; TORGO; RIBEIRO, 2019) combines biased versions of the under- and over-sampling strategies which depend exclusively on the relevance function provided to the dataset without requiring establishing a relevance threshold. Under the WERCS, the relevance function and a modification

of the relevance are used to attribute weights that are used as inclusion and removal criteria for the examples. Algorithm 9 details this resampling strategy. The over-sampling and under-sampling on lines 4 and 7, respectively, are performed considering weights obtained on lines 3 and 6. These weights are calculated based on the relevance function. The weights associated with over-sampling $WOver$ are proportional to the relevance function (line 3). Therefore, the higher the relevance of a case, the higher its probability of being selected for generating new cases. Conversely, the weights associated with under-sampling $WUnd$ are inversely proportional to the relevance value (line 6). Thus, normal examples, which are usually associated with lower relevance values, have a higher probability of being removed rather than used in the generation process. The number of generated and removed samples is defined based on the percentage of over-sampling o and under-sampling u, respectively.

Therefore, the main advantage of this technique is that as a relevance threshold is not set a priori, each example can participate in both processes. Thus, both under-sampling and over-sampling strategies are applied over the entire dataset. Also, the technique eliminates the dependency on the relevance threshold t_R that was a key component necessary for applying all other resampling strategies reviewed in this work.

Algorithm 9 WEighted Relevance-based Combination Strategy (WERCS)

Input:
D - original dataset
u - percentage of under-sampling
o - percentage of over-sampling
Output:
$newD$ - a new modified dataset

1: $\phi() \leftarrow$ relevance function
2: $newD \leftarrow D$
3: $WOver \leftarrow \{\phi(y_i) \mid y_i \in Y\}$
4: $Over \leftarrow$ sample $o \times |D|$ cases from D with $WOver$ weights ▷ over-sampling procedure
5: $newD \leftarrow newD \bigcup Over$
6: $WUnd \leftarrow \{1 - \phi(y_i) \mid y_i \in Y\}$
7: $Und \leftarrow$ sample $u \times |D|$ cases from D with $WUnd$ weights
8: $newD \leftarrow newD \backslash Und$ ▷ under-sampling procedure
9: **return** $newD$

3.3.7 Advantages and Disadvantages of Strategies

The strategies to resample data can have both advantages and disadvantages. Therefore, it is crucial to understand the behavior of each strategy. While these strategies can potentially

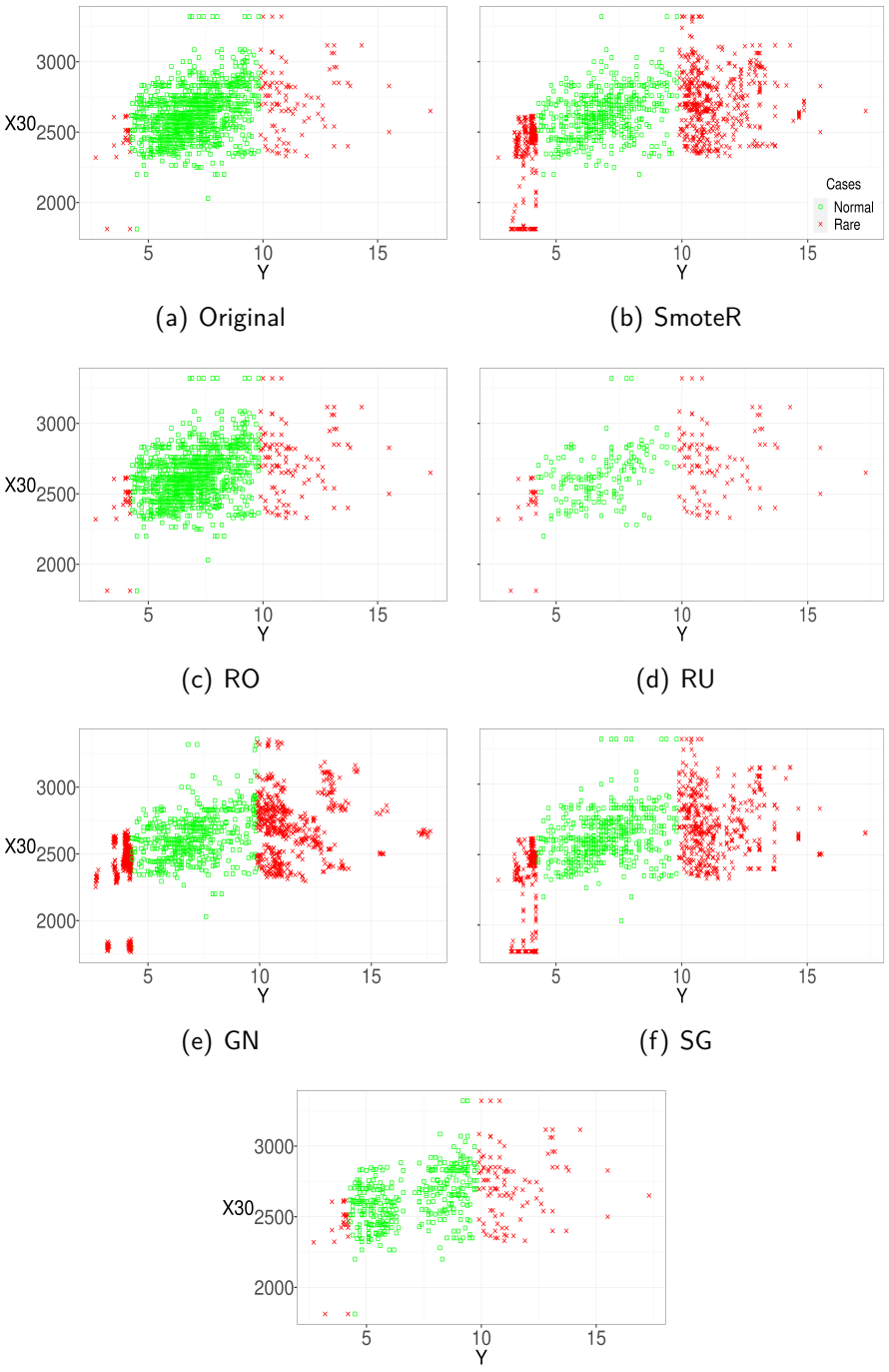
enhance learning, they can also impede the learning process of the models. Figure 17 introduces the result of applying the resampling strategies to the *FuelCons* dataset. The following values were attributed to the algorithm's parameter: $u/o =$ balance and $t_R = 0.8$ (except for the WERCS, since it does not require establishing the threshold). The standard values were adopted for the remaining parameters. For the visualization, the target values (Y) and the attribute (X30) were considered.

Despite selecting the nearest examples to generate new cases, SmoteR still involves the risk of the example being too far and of generating an example that does not correspond to the seed very well. This phenomenon is shown in the lower left side of Figure 18(a), where the generated examples are far from the original examples. In the RO strategy, high percentages of over-sampling may cause an overfitting (BRANCO; TORGO; RIBEIRO, 2019) problem. Even though the technique increases the representation of rare cases considerably, the generated dataset does not present a high points diversity. The generation process consists in just duplicating existing samples without covering the feature space well.

Figure 18(b) shows the rare data points in darker shade, given that the RO only makes copies of the examples. This can therefore lead to learning algorithms overfitting such rare examples. In addition, if the replication rate is too high, many duplicate data points are added to the dataset which can significantly increase the training time. In contrast to the RO, in the RU strategy some meaningful information may be lost due to the removal of training data (Figure 18(e)), which may hamper the learning of the model. Figure 18(c) shows the result after using the GN strategy, which promotes over-sampling by adding normally distributed noise. Once again, in contrast to the RO strategy, examples different from the originals ones are generated, and this diversity can help to mitigate overfitting. For the SG strategy, even though one of its goals is to reduce the risks seen in SmoteR by creating different examples from the original, Figure 18(d) shows that there is still a similarity with the SmoteR distribution. However, when compared to GN, it is evident that the diversity of generated examples is higher in SG. In the WERCS strategy (Figure 18(f)), it can be seen that the green data points are divided into two groups after the under-sampling, and this result can complicate the learning process. The WERCS over-sampling strategy performs similarly to RO, where the generated data are copies of the originals; such as, no new information is added to the training set.

The advantages and disadvantages of each resampling strategy are quite evident, as is the fact that there is no perfect strategy. We hypothesize that other variables, such as the regression model and the dataset under investigation, are required to determine the best data

Figure 6 – Distribution of the examples of the FuelCons dataset after applying the resampling strategies, considering t_R=0.8.



(a) Original (b) SmoteR

(c) RO (d) RU

(e) GN (f) SG

(g) WERCS

Source: Prepared by the author.

resampling strategy. Thus, our research allows to understand the behaviors of these strategies with different regression models and problems, which in turn allows to establish directions for combinations of the three variables, namely, the resampling strategy, the regression model, and the dataset.

3.4 RESEARCH METHODOLOGY

3.4.1 Datasets

Experiments were performed using 30 imbalanced regression datasets chosen to match the frequency generally used in studies looking at imbalanced regression. The levels of imbalance in these datasets are defined from the relevance function (Section 3.2.1). A study conducted by Branco, Torgo e Ribeiro (2019) involved varying the relevance threshold from 0.5 to 1. Nevertheless, the findings showed a complex relationship between the number of rare cases, the learning algorithm, and the applied pre-processing strategy. Therefore, our experiments considered a commonly used threshold (t_R) of 0.8, as used in Branco, Torgo e Ribeiro (2017), Branco, Torgo e Ribeiro (2019) and Branco, Torgo e Ribeiro (2018). Thus, we obtained datasets with different percentages of rare cases (imbalanced levels), varying between 5.1% and 23.4%. The main features of these sets are presented in Table 4. Datasets are presented in descending order in terms of the percentage of rare cases (%Rare). It is important to clarify that counting rare cases is conducted across the entire dataset, as commonly practiced in the literature. Counting rare cases on the entire dataset is crucial for comprehensively understanding their rarity within the data context. This approach allows us to analyze the model's behavior within the original context of the dataset. However, resampling strategies are applied only to the training set to prevent data leakage during cross-validation. The nominal attributes were codified, transforming the vector of categories into whole values between 0 and the number of categories-1. As for the ordinal attributes, a pre-defined order was established (e.g., *small*: 1, *medium*: 2, *large*: 3).

For each dataset, the results were calculated by applying two 10-fold cross-validation repetitions (i.e., 2×10 cross-validation) in order to obtain the mean and standard deviation of the results. Nested cross-validation with 2-fold was employed to optimize the hyperparameters of the resampling strategies, specifically utilizing the SERA metric for optimization. The SERA metric was chosen to optimize the hyperparameters because it was specifically created

for imbalanced regression. This metric evaluates models' performance in predicting extreme values, penalizing model biases without requiring a threshold, and conducting a global assessment (RIBEIRO; MONIZ, 2020). Unlike the F1-score, which conducts a local assessment by considering only rare examples, SERA evaluates all examples.

3.4.2 Algorithms

The experiments were performed with the following learning algorithms: Bagging (BG), Regression Tree (RT), Multilayer Perceptron (MLP), Random Forest (RF), Support Vector Regressor (SVR), and XGBoost (XG). Default hyperparameters were applied for these models. For details and descriptions of default hyperparameters and used packages, refer to Appendix A.

As resample techniques, we considered the following strategies: SmoteR (SMT), Randon Over-sampling (RO), Randon Under-sampling (RU), Introduction of Gaussian Noise (GN), SMOGN (SG), and WEighted Relevance-based Combination Strategy (WERCS). Details about hyperparameters and packages can be found in Table 5.

3.4.3 Model Evaluation

In imbalanced tasks, choosing the appropriate metrics for model evaluation is essential. This work uses the F1-score and SERA metrics to evaluate regression models, allowing the evaluation of different perspectives of the model performance. While the F1-score metric is based on the concept of utility-based evaluation and performs a local assessment according to the definition of a relevance threshold, the SERA metric evaluates the effectiveness of models in predicting extreme values while penalizing several model biases without the need for a threshold, and performing a global assessment (RIBEIRO; MONIZ, 2020). The results for the RMSE and MAE metrics can be consulted in the supplementary material (Appendix B) for benchmarking purposes.

Table 4 – Characteristics of the 30 datasets used in the experiments. N: number of cases; p.total: number of attributes; p.nom: number of nominal attributes; p.num: number of numeric attributes; nRare: number of rare cases; Imbalance ratio (IR): $\frac{|D_R|}{|D_N|}$; %Rare: $100 \times nRare/N$. Datasets are arranged in descending order regarding the percentage of rare cases (%Rare).

Datasets	N	p.total	p.nom	p.num	nRare	IR	%Rare
wine-quality	6497	11	0	11	1523	0.306	23.4
analcat-apnea3	450	11	0	11	103	0.297	22.9
meta	528	65	0	65	108	0.257	20.5
cocomo-numeric	60	56	0	56	10	0.200	16.7
Abalone	4177	8	1	7	679	0.194	16.3
a3	198	11	3	8	32	0.193	16.2
forestFires	517	12	0	12	79	0.180	15.3
a1	198	11	3	8	28	0.165	14.1
a7	198	11	3	8	27	0.158	13.6
boston	506	13	0	13	65	0.147	12.8
pdgfr	79	320	0	320	10	0.145	12.7
sensory	576	11	0	11	69	0.136	12.0
a2	198	11	3	8	22	0.125	11.1
kdd-coil-1	316	18	0	18	34	0.121	10.8
triazines	186	60	0	60	20	0.120	10.8
airfoild	1503	5	0	5	161	0.120	10.7
treasury	1049	15	0	15	109	0.116	10.4
mortgage	1049	15	0	15	106	0.112	10.1
debutanizer	2394	7	0	7	240	0.111	10.0
fuelCons	1764	37	12	25	164	0.103	9.3
heat	7400	11	3	8	664	0.099	9.0
california	20640	8	0	8	1821	0.097	8.8
AvailPwr	1802	15	7	8	157	0.095	8.7
compactiv	8192	21	0	21	713	0.095	8.7
cpuSm	8192	12	0	12	713	0.095	8.7
maxTorq	1802	32	13	19	129	0.077	7.2
lungcancer-shedden	442	24	0	24	25	0.060	5.7
space-ga	3107	6	0	6	173	0.059	5.6
ConcrStr	1030	8	0	8	55	0.056	5.3
Accel	1732	14	3	11	89	0.054	5.1

Source: Prepared by the author.

Table 5 – Resampling strategies, hyperparameters, and packages used

Algorithms	**Hyperparameters**	**Packages**
SMT	u/o = {balance, extreme}, k = {3, 5, 7}	ImbalancedLearningRegression
RO	o = {balance, extreme}	ImbalancedLearningRegression
RU	u = {balance, extreme}	ImbalancedLearningRegression
GN	u/o = {balance, extreme}, δ = {0.00,0.05,0.10,...,0.95,1.00}	ImbalancedLearningRegression
SG	u/o = {balance, extreme}, k = {3, 5, 7}, δ = {0.00,0.05,0.10,...,0.95,1.00}	smogn
WERCS	u, o = {0.3, 0.5, 0.7, 0.9}	resreg

Source: Prepared by the author.

<https://pypi.org/project/ImbalancedLearningRegression/> - Version 0.0.1
<https://pypi.org/project/smogn/> - Version 0.1.2
<https://pypi.org/project/resreg/> - Version 0.2

3.5 RESULTS

The experiments aimed at answering the following research questions:

1. Is it worth using resampling strategies?
2. Which resampling strategies influence the predictive performance the most?
3. Does the choice of best strategy depend on the problem, the learning model, and the metrics used?
4. Does the number of training examples resulting from each strategy influence the results?
5. Do the features of the data (percentage of rare cases, number of rare cases, dataset size, number of attribues and imbalance ratio) impact the predictive performance of the models?

Tables 6 and 7 show how many times each algorithm obtains the highest value for the F1-score and SERA metrics, respectively. Where there is a tie, each of the n tied strategies receives $1/n$ point. Each row in this table must add up to 30, the number of datasets assessed. For both metrics used, we found that the larger number of wins occurs when using some of the resampling strategies, which points to an advantage of using such strategies, and highlighting the RO and GN considering the F1-score, and the GN and WERCS, according to SERA. Another point observed is that the choice of best strategy possibly depends on the regression model used. As for the metrics, both agree regarding the GN strategy. By observing the score by rows, also in Tables 6 and 7, it is clear that there is no general agreement between the datasets for a resampling strategy since each point is a dataset, and all of them are distributed in different strategies. The results per learning algorithm, including mean and standard deviation, can be accessed in the supplementary material. –Appendix B..

To identify the best way to preprocess each dataset, Tables 8 and 9 introduce the best and worst results for the F1-score and SERA metrics, respectively. The results show that most datasets have distinct preferences in terms of combining the best learning model and the resampling strategy. This distinction is also found for the metrics used. As for the worst results, the SVR and MLP, without preprocessing, are the worst combinations for both metrics. Thus, balancing the dataset before applying these models is crucial to reaching more promising results. It is also crucial to note the significant difference between the best and worst results

Table 6 – Number of times each algorithm and resampling strategy achieved the best result according to the F1-score metric.

	None	SMT	RO	RU	GN	SG	WERCS
BG	3	3	13	2	6	3	0
RT	6	5	8	3	6	2	0
MLP	3	4	12	1	3	6	1
RF	2	4	9	6	3	3	3
SVR	1	5	13	1	2	6	2
XG	4.7	1	7	5	5.2	3.2	4
Total	19.7	22	**62**	18	**25.2**	23.2	10

Source: Prepared by the author.

Table 7 – Number of times each algorithm and resampling strategy reached the best result according to the SERA metric.

	None	SMT	RO	RU	GN	SG	WERCS
BG	4	3	3	5	4	1	10
RT	4	2	7	6	5	2	4
MLP	2	4	9	1	4	7	3
RF	2	1	4	7	6	2	8
SVR	0	5	1	3	12	5	4
XG	3	1	2	5	8.5	3.5	7
Total	15	16	26	27	**39.5**	20.5	**36**

Source: Prepared by the author.

per problem. So, obtaining good results depends on the correct choice of resampling strategy and learning model. Unfortunately, the SG strategy failed to perform on the california, heat, and wine-quality datasets. These are large datasets, highlighting the potential challenges in optimizing hyperparameters, rendering the use of this model impractical.

We applied the Friedman test to better measure the advantage of using resampling strategies ($p-value < 0.05$). The Friedman statistical test was chosen since it can compare multiple techniques over several datasets (DEMŠAR, 2006). For this measurement, ranking sequences are compared. Tables 10 and 11 present the mean ranking of the means of the algorithms with a combination of each resampling strategy, considering the F1-score and SERA metrics. The lower the ranking, the better the algorithm performance. The algorithms used present significant differences. In general, the best average rank of each algorithm was obtained by using some of the resampling strategies evaluated in this work.

Table 8 – Best and worst results for each dataset based on the F1-score metric

Datasets	**Best result**		**Worst result**	
wine-quality	0.738	RF.RO	0.000	SG*
analcat-apnea3	0.233	MLP.NONE	2.00e-5	SVR.NONE
meta	0.435	BG.GN	2.00e-5	SVR.NONE
cocomo-numeric	0.371	RF.RU	1.60e-5	SVR.NONE
Abalone	0.702	RF.RU	1.79e-1	SVR.NONE
a3	0.506	RF.RU	1.95e-5	SVR.NONE
forestFires	0.403	SVR.SMT	2.00e-5	SVR.NONE
a1	0.723	RF.SG	2.00e-5	SVR.NONE
a7	0.411	SVR.RO	2.00e-5	SVR.NONE
boston	0.893	RF.RO	2.00e-5	SVR.NONE
pdgfr	0.229	RF.GN	1.60e-5	RF.SG
sensory	0.672	XG.GN	2.00e-5	SVR.NONE
a2	0.580	RF.RU	2.00e-5	SVR.NONE
kdd-coil-1	0.677	RF.RU	1.90e-5	SVR.NONE
triazines	0.226	RF.WERCS	0.028	RF.NONE
airfoild	0.951	BG.RO	2.00e-5	SVR.WERCS
treasury	0.980	RF.GN	0.777	SVR.NONE
mortgage	0.985	RF.RO	0.834	SVR.NONE
debutanizer	0.901	RF.SMT	0.646	MLP.GN
fuelCons	0.942	XG.GN	0.094	MLP.RU
heat	0.989	XG.RO	0.000	SG*
california	0.902	XG.NONE	0.000	SG*
AvailPwr	0.977	XG.NONE	0.725	SVR.NONE
compactiv	0.528	RF.SG	0.109	MLP.RO
cpuSm	0.526	RF.SMT	0.105	MLP.RO
maxTorq	0.988	XG.WERCS	2.00e-5	SVR.NONE
lungcancer-shedden	0.665	MLP.SG	2.00e-5	SVR.NONE
space-ga	0.802	XG.GN	2.00e-5	SVR.NONE
ConcrStr	0.966	XG.GN	2.00e-5	SVR.RU
Accel	0.961	XG.RO	2.00e-5	SVR.GN

* Not completed in a reasonable timeframe

Source: Prepared by the author.

To verify which approaches are statistically different, we applied the Nemenyi post-hoc test. Figure 7 (a-f) illustrates the critical difference diagrams (DEMŠAR, 2006) for each of the learning models, considering the F1-score metric. The horizontal line demonstrates the significance of the difference between the models. Models that are not connected present a

Table 9 – Best and worst results for each dataset based on the SERA metric

Datasets	Best result		Worst result	
wine-quality	1.43e+2	RF.WERCS	0.000	SG*
analcat-apnea3	3.93e+7	RF.GN	4.80e+8	SVR.NONE
meta	2.79e+7	MLP.GN	3.83e+7	RT.SG
cocomo-numeric	4.16e+5	XG.SMT	2.91e+6	SVR.NONE
Abalone	1.17e+3	RF.WERCS	2.59e+3	SVR.NONE
a3	3.30e+2	SVR.GN	2.48e+7	MLP.RU
forestFires	1.88e+5	MLP.SG	3.02e+5	RT.GN
a1	1.63e+3	SVR.SG	1.76e+6	MLP.NONE
a7	2.58e+2	XG.RU	2.62e+7	MLP.RU
boston	2.30e+2	BG.WERCS	2.51e+8	MLP.RU
pdgfr	3.86e-2	RF.GN	2.40e-1	RT.WERCS
sensory	1.39e+1	RF.RU	1.16e+2	MLP.SMT
a2	6.91e+2	SVR.GN	2.28e+7	MLP.RU
kdd-coil-1	2.43e+3	SVR.RU	9.09e+3	SVR.NONE
triazines	6.99e-2	RF.RU	3.54e-1	MLP.NONE
airfoild	6.78e+7	XG.RO	2.05e+11	SVR.NONE
treasury	3.08e+0	RF.GN	1.78e+3	MLP.RU
mortgage	1.30e+0	XG.RO	1.70e+2	RT.SG
debutanizer	3.23e-1	RF.SG	2.52e+0	MLP.NONE
fuelCons	1.52e+1	XG.GN	1.32e+7	MLP.RU
heat	6.42e+2	XG.GN	0.000	SG*
california	1.87e+12	RF.RU	0.000	SG*
AvailPwr	4.72e+3	XG.NONE	1.77e+5	MLP.RU
compactiv	1.61e+3	RF.WERCS	6.26e+7	MLP.NONE
cpuSm	2.11e+3	RF.WERCS	5.25e+7	MLP.RO
maxTorq	4.63e+3	XG.GN	1.17e+6	SVR.NONE
lungcancer-shedden	5.75e+1	SVR.GN	2.16e+2	SVR.NONE
space-ga	1.78e+0	XG.WERCS	4.16e+10	MLP.RO
ConcrStr	5.64e+2	XG.WERCS	9.27e+3	SVR.GN
Accel	2.34e+1	XG.GN	7.41e+5	MLP.RU

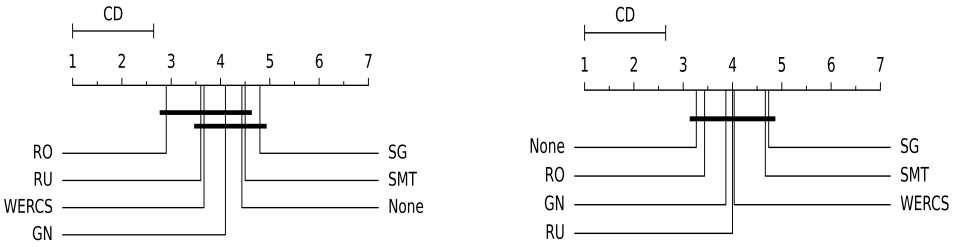
* Not completed in a reasonable timeframe

Source: Prepared by the author.

significant difference ($p-value < 0.05$) in relation to the others. This test once again confirms that, globally, resampling strategies can significantly improve the regressors' performance. The Nemenyi test reveals that the RO obtained the best results and the most significant differences in relation to None (data without any preprocessing) for the metric F1-score. In most cases,

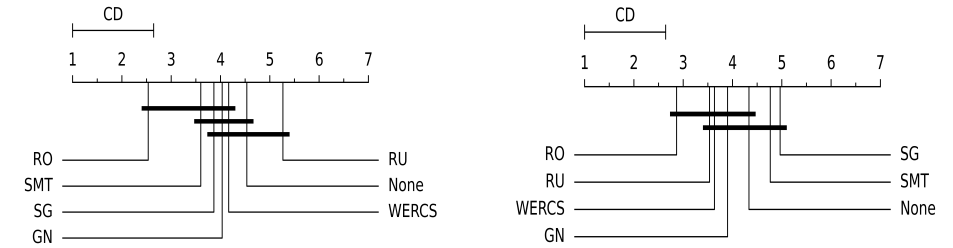
the SMT, SG, RU techniques achieve the worst results. Figure 8 (a-f) considers the SERA metric; in such a scenario, most of the best results are obtained using the GN strategy, followed by WERCS, given the number of times where the best results were achieved in the critical difference chart.

Figure 7 – Critical difference diagrams for each learning algorithm considering the F1-score metric.



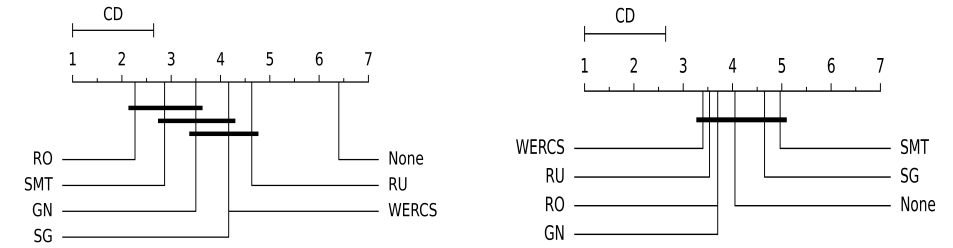
(a) Results of the F1-score metric for the BG algorithm

(b) Results of the F1-score metric for the RT algorithm



(c) Results of the F1-score metric for the MLP algorithm

(d) Results of the F1-score metric for the RF algorithm



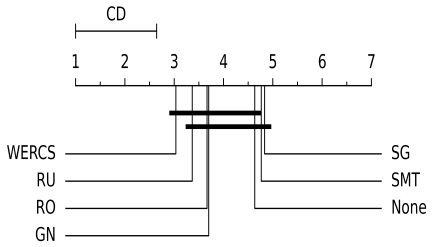
(e) Results of the F1-score metric for the SVR algorithm

(f) Results of the F1-score metric for the XG algorithm

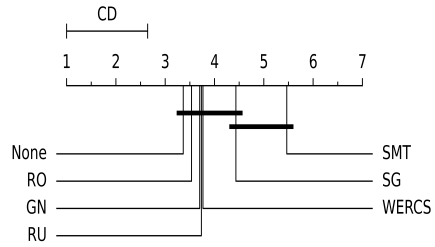
Source: Prepared by the author.

Another interesting fact is how different learning algorithms perform when no resampling strategy is applied. In both metrics, the RT model achieved better results with the original data sets. Additionally, an interesting aspect involves the ensemble models, Random Forest (RF) and XGBoost (XG) obtained better results than single models, corroborating the analysis conducted in Moniz, Branco e Torgo (2017), which says that ensemble methods provide a better result than single models. Conversely, the SVR and MLP algorithms obtained the worst results, especially when no preprocessing techniques were employed. Thus, it can be concluded

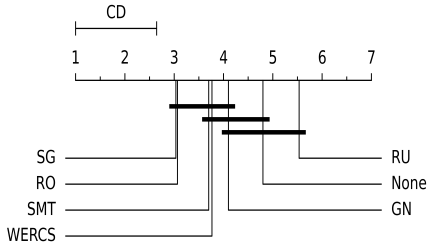
Figure 8 – Critical difference diagrams for each learning algorithm considering the SERA metric.



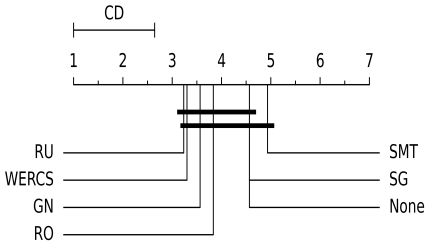
(a) Results of the SERA metric for the BG algorithm



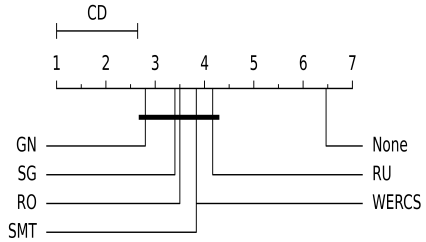
(b) Results of the SERA metric for the RT algorithm



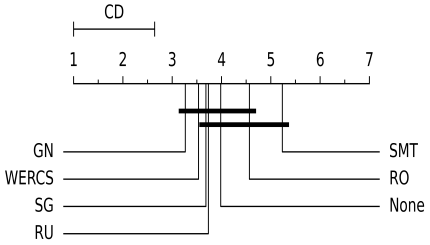
(c) Results of the SERA metric for the MLP algorithm



(d) Results of the SERA metric for the RF algorithm



(e) Results of the SERA metric for the SVR algorithm



(f) Results of the SERA metric for the XG algorithm

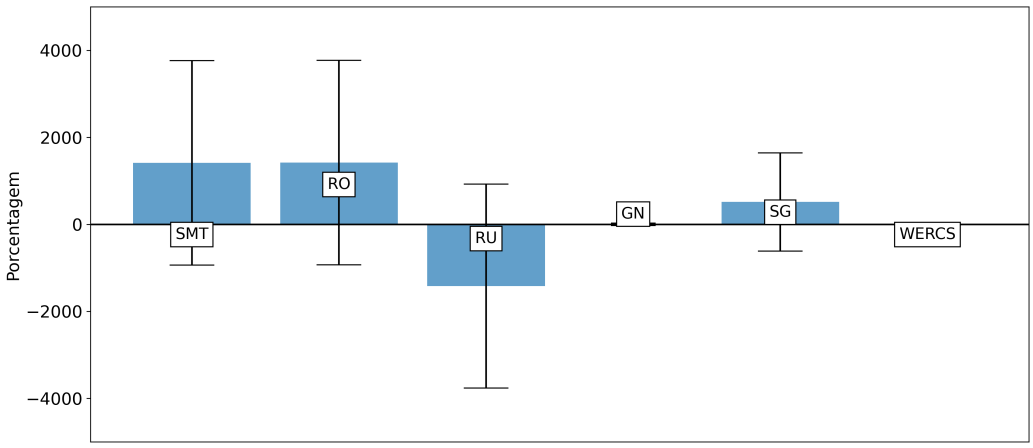
Source: Prepared by the author.

that these algorithms are the most affected by having an imbalanced target distribution and require special attention when applied in the imbalanced regression context.

As described in Section 3.3, each resampling algorithm uses different heuristics to balance the dataset. Figure 9 illustrates the percentage of increase/decrease in the training examples for each strategy. It was previously concluded that the best results were achieved using the RO, GN, and WERCS strategies. The GN and WERCS strategies present a small percentage of 1.28% and 2.83%, respectively. Conversely, the RO presents an increased percentage of 1421.1%. Therefore, the influence of the number of examples on the results is unclear since the strategies with different percentages of increase/decrease obtained good results. Nonetheless, it may be disadvantageous (from a training time point of view) to use a strategy, such as the RO, that considerably increases the training set. Other strategies also deliver satisfactory results

without excessively increasing the training set. More details about the number of instances after the application of the resampling strategies can be found in the supplementary material – Appendix C.

Figure 9 – Percentage of increase/decrease in the training set for each resampling strategy.



Source: Prepared by the author.

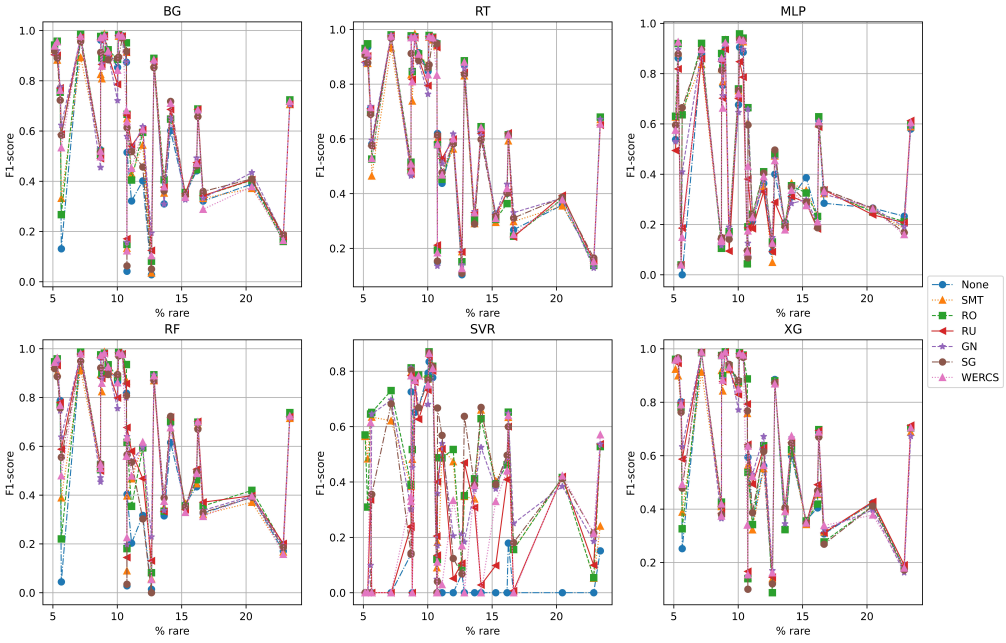
Figures 10, 11, 12, 13 and 14 present the F1-score results arranged according to some dataset characteristics in a bid to assess their impact on the performance of the models. The following characteristics were assessed: percentage of rare cases, number of rare cases, dataset size, number of attributes, and imbalance ratio. The imbalance ratio is calculated as the ratio between the number of rare cases (D_R) and the number of normal cases (D_N), i.e., $\frac{|D_R|}{|D_N|}$. Each box represents a regression model (BG, RT, MLP, RF, SVR and XG), and each point represents a specific set of data, and each line represents a resampling strategy (None, SMT, RO, RU, GN, SG and WERCS).

The results presented in Figure 10 correspond to the same ordering provided in Table 4, where the datasets are arranged in decreasing order of the percentage of rare cases. In such conditions, it is not possible to find any pattern. Therefore, it is unclear how this aspect relates to the model's performance. Figures 11 and 12 are arranged according to the number of rare cases and the dataset size, respectively. These circumstances reveal that the smaller datasets with a lower number of rare cases represent the hardest tasks, as observed in Branco, Torgo e Ribeiro (2019). Figure 13 illustrates the evolution of F1 considering the number of attributes in each dataset. In some instances, it is noticeable that datasets with fewer features exhibit superior performance. Finally, in Figure 14, the datasets are sorted according

to their respective imbalance ratios. The regression models with all resampling strategies face more significant challenges when dealing with datasets exhibiting higher imbalance ratios. This difficulty arises because higher imbalance ratios mean the rare cases are significantly underrepresented compared to the normal cases. As a result, the model may struggle to learn the underlying patterns and become biased toward the normal cases.

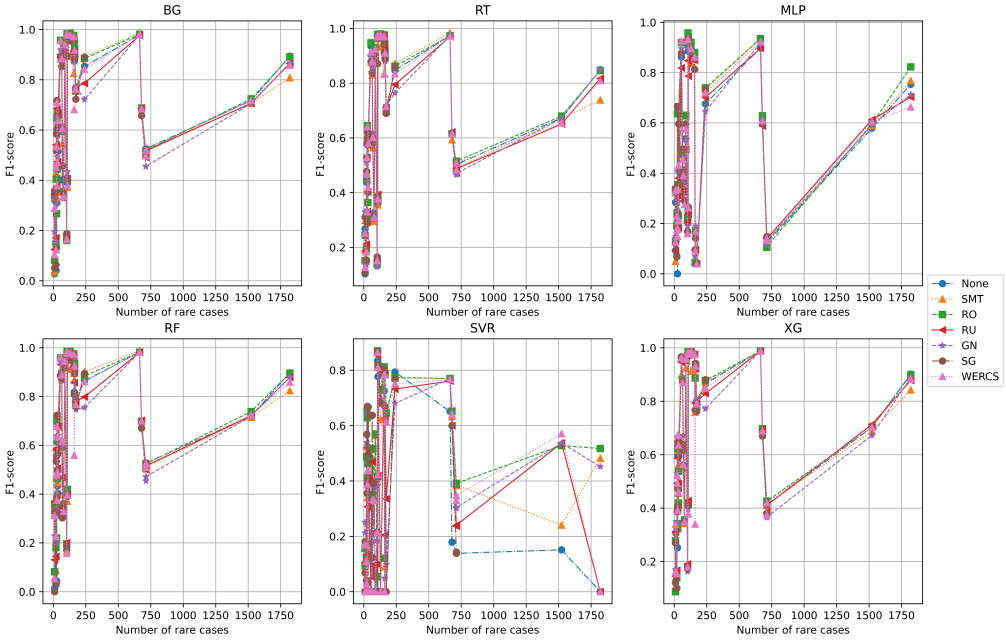
For all the evaluated dataset characteristics, the behavior of the resampling strategies is quite similar, resulting in an overlap of the graph's line. For better clarity, another analysis is conducted considering the best F1-score for each dataset. The figures in Appendix D present the best F1-score for each dataset, considering the dataset characteristics. With this, we can visualize how the data characteristics affect the performance of the top models. The percentage of rare cases does not exhibit a clear pattern. Thus, concluding whether this characteristic affects the model's performance is challenging. Regarding the number of rare cases and the dataset size, models achieve better performance when there are more rare cases and a larger dataset. When we consider the number of attributes, we observe that a higher number leads to better model performance. As for the imbalance ratio, the higher the imbalance ratio, the worse the model's performance.

Figure 10 – Evolution of the *F1-score* with datasets sorted by percentage of rare cases.



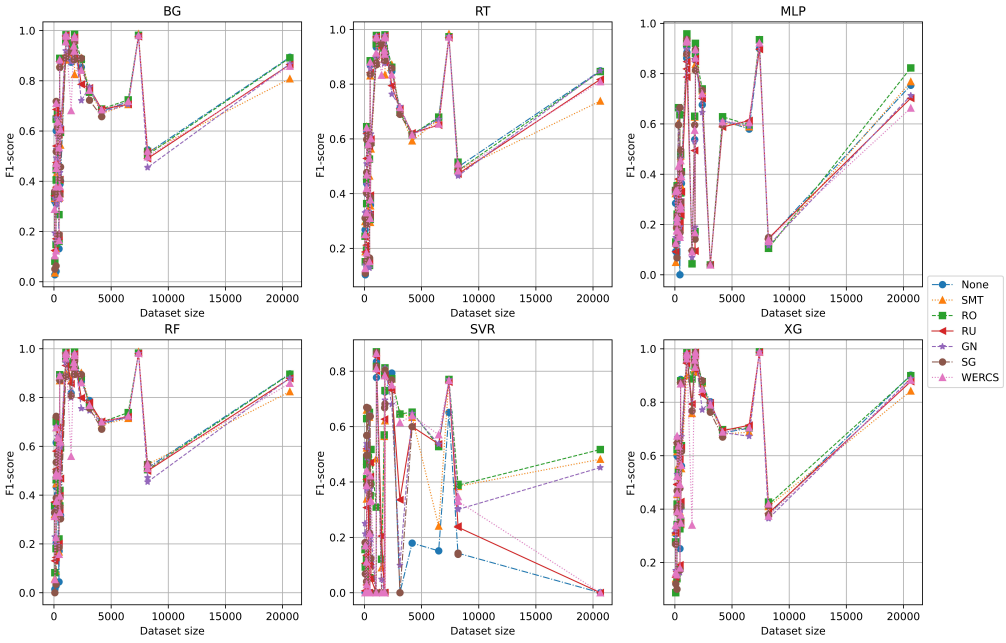
Source: Prepared by the author.

Figure 11 – Evolution of the *F1-score* with datasets sorted by number of rare cases.



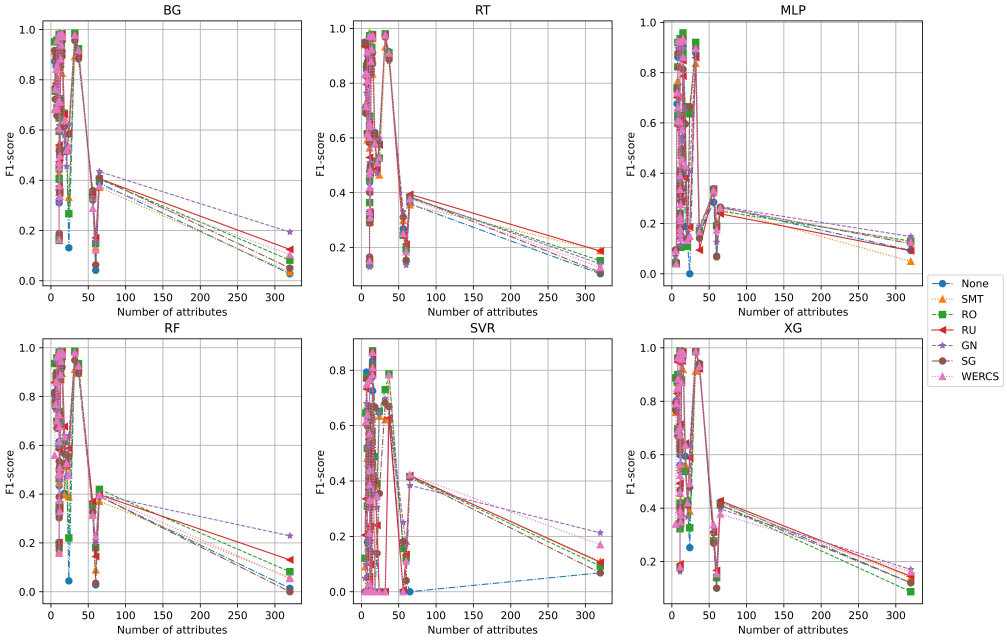
Source: Prepared by the author.

Figure 12 – Evolution of the *F1-score* with datasets sorted by size.



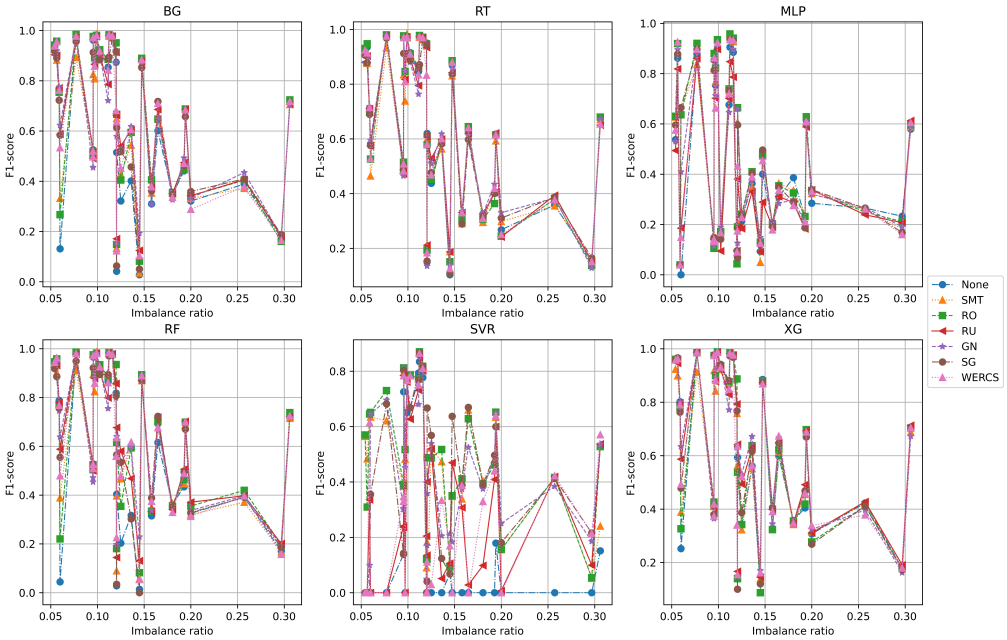
Source: Prepared by the author.

Figure 13 – Evolution of the *F1-score* with datasets sorted by number of attributes.



Source: Prepared by the author.

Figure 14 – Evolution of the *F1-score* with datasets sorted by imbalance ratio.



Source: Prepared by the author.

Table 10 – Average ranking (F1-score).

Algorithm	Average
RF.RO	10.0
RF.GN	11.8
RF.RU	11.9
XG.RU	12.0
BG.RO	12.2
RF.WERCS	12.4
XG.WERCS	12.8
XG.GN	13.3
BG.RU	13.6
XG.RO	14.6
BG.WERCS	15.5
XG.NONE	15.6
BG.GN	16.0
RF.NONE	17.3
RF.SMT	17.8
XG.SMT	18.4
XG.SG	18.5
BG.NONE	18.8
BG.SMT	18.8
RT.RO	19.3
RF.SG	20.3
RT.NONE	20.4
BG.SG	20.5
RT.RU	20.9
RT.GN	21.0
RT.WERCS	21.1
RT.SMT	22.9
RT.SG	25.1
SVR.RO	26.5
SVR.SMT	26.7
MLP.RO	26.8
SVR.GN	27.6
SVR.SG	28.4
MLP.SMT	29.8
MLP.GN	30.8
SVR.WERCS	31.3
MLP.WERCS	31.7
MLP.SG	31.7
MLP.NONE	32.5
MLP.RU	33.3
SVR.RU	33.8
SVR.NONE	39.6

Source: Prepared by the author.

Table 11 – Average ranking (SERA).

Algorithm	Average
RF.RU	8.7
RF.GN	10.0
RF.WERCS	10.9
RF.RO	11.3
XG.GN	11.5
XG.WERCS	12.4
XG.RU	12.5
BG.WERCS	13.4
RF.NONE	13.6
XG.SG	13.6
XG.NONE	13.9
BG.RU	14.2
RF.SG	14.3
BG.GN	14.7
XG.RO	14.8
BG.RO	15.4
RF.SMT	15.8
XG.SMT	17.8
BG.NONE	18.3
BG.SG	18.7
BG.SMT	19.6
SVR.SG	23.9
SVR.GN	23.9
RT.GN	26.8
RT.NONE	26.9
SVR.RO	27.1
RT.RU	27.2
RT.RO	27.3
RT.SG	27.4
SVR.RU	27.5
MLP.SG	27.8
SVR.SMT	28.0
RT.WERCS	28.6
MLP.RO	28.7
SVR.WERCS	28.9
MLP.GN	29.4
MLP.SMT	30.4
MLP.WERCS	30.7
RT.SMT	31.4
MLP.NONE	33.9
MLP.RU	35.1
SVR.NONE	36.6

Source: Prepared by the author.

3.6 LESSONS LEARNED

Different approaches have been proposed in a bid to solve the imbalanced problem in the context of regression, including resampling strategies. Our research introduced a review and an experimental study of the main resampling strategies for dealing with imbalanced regression problems. In this section, the research questions are revisited and answered succinctly.

1. **Is it worth using resampling strategies?**

   We answer this question by accounting for the number of times that each strategy won (Tables 6 and 7). For both metrics, four of the resampling strategies used won more times than when no resampling strategy was used. Furthermore, the Nemenyi post-hoc statistical tests performed (Figures 7 and 8) demonstrate that many resampling strategies are statistically better as compared to the absence of a strategy. Therefore, it is advantageous to use (some) resampling strategies.

2. **Which resampling strategies influence the predictive performance the most?**

   Considering the F1-score metric, the RO and GN strategies positively influenced the results of the learning algorithms. As for the SERA metric, the GN and WERCS techniques are the best strategies. Statistically, in general, the GN, RO, and WERCS strategies held the best results (Figures 7 and 8). Conversely, in terms of predictive performance, the SMT, SG, RU techniques achieve the worst results.

3. **Does the choice of best strategy depend on the dataset, the learning model, and the metrics used?**

   Most of the datasets used have distinct preferences regarding the combination of the best regression model and resampling strategy (Tables 8 and 9). For the regression models, different resampling strategies can reach better results. As for the metrics, both agree that the GN is a good resampling strategy. Nonetheless, there are cases of disagreement between them.

4. **Does the number of training examples resulting from each strategy influence the results?**

Given that the best results were obtained using the GN, RO, and WERCS strategies, which have different percentages (1.28%, 1421.1%, 2.83%, respectively) of increase/decrease in the training examples (Figure 9), the influence of the number of examples on the results is not clear. Nonetheless, it may not be advantageous (from a training time point of view) to use a strategy like the RO, which considerably increases the training set, as other strategies also deliver equivalent results without this excessive increase.

5. **Do the features of the data (percentage of rare cases, number of rare cases, dataset size, number of attribues and imbalance ratio) impact the predictive performance of the models?**

   In the studies performed, the percentage of rare cases did not have a clear impact on the results. On the other hand, considering the dataset size and number of rare cases, it could be seen that the smaller datasets with fewer rare cases correspond to the most difficult tasks. Models demonstrate superior performance in datasets with fewer features. Lastly, concerning the imbalance ratio, regression models encounter more significant challenges with a higher imbalance ratio. The results for this question are shown in Figures 10, 11, 12, 13 and 14. Appendix D presents the evolution of the best F1-score for each dataset characteristic, providing a clearer view of the impact of these dataset characteristics on model performance.

3.7 CONCLUSION

This work reviews and performs a comparative study of data resampling strategies for handling imbalanced regression problems. We reviewed six state-of-the-art resampling strategies for regression based on three approaches: i) under-sampling, ii) oversampling, and iii) a mix of undersampling and oversampling, while discussing the advantages and drawbacks of each existing technique.

Then, we performed an extensive experimental analysis comprised of 6 regression algorithms and 7 scenarios (6 resampling strategies and not using resampling) that can guide the development of new strategies to solve the imbalanced regression problem. Our experimental results demonstrate that it is important to use a resampling technique for most models as resampling techniques lead to statistically better results. The experimental study also shows

that no resampling technique outperforms all others. Furthermore, choosing the best resampling technique depends on three main factors: the learning algorithm, the dataset, and the performance metric used to assess the model's performance.

Further studies should address the recommendation of combining resampling strategies with a regression model for each specific dataset. Another element worth addressing is the dataset characteristics, which should be investigated through data complexity measures (LORENA et al., 2018) in order to assess the adverse effects of these features on prediction performance. Moreover, an essential point to address involves proposing a new relevance function since currently, only one definition exists. This proposal aims to conduct studies and comparisons regarding the definition of an imbalanced regression dataset.

4 IMBALANCED REGRESSION PIPELINE RECOMMENDATION

AUTHORS

Juscimara G. Avelino *Universidade Federal de Pernambuco, Recife, Brazil*
George D. C. Cavalcanti *Universidade Federal de Pernambuco, Recife, Brazil*
Rafael M. O. Cruz *École de technologie supérieure, Montreal, Canada*

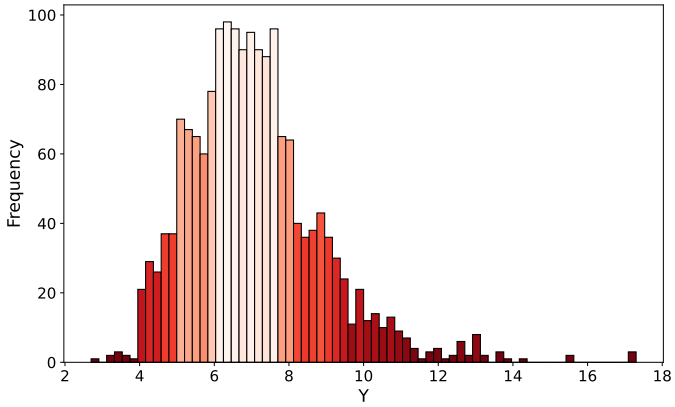
ABSTRACT

Imbalanced problems are prevalent in various real-world scenarios and are extensively explored in classification tasks. However, they also present challenges for regression tasks due to the rarity of certain target values. A common alternative is to employ balancing algorithms in pre-processing to address dataset imbalance. However, due to the variety of resampling methods and learning models, determining the optimal solution requires testing many combinations. Furthermore, the learning model, dataset, and evaluation metric affect the best strategies. This work proposes the Meta-learning for Imbalanced Regression (Meta-IR) framework, which diverges from existing literature by training meta-classifiers to recommend the best pipeline composed of the resampling strategy and learning model per task in a zero-shot fashion. The meta-classifiers are trained using a set of meta-features to learn how to map the meta-features to the classes indicating the best pipeline. We propose two formulations: Independent and Chained. Independent trains the meta-classifiers to separately indicate the best learning algorithm and resampling strategy. Chained involves a sequential procedure where the output of one meta-classifier is used as input for another to model intrinsic relationship factors. The Chained scenario showed superior performance, suggesting a relationship between the learning algorithm and the resampling strategy per task. Compared with AutoML frameworks, Meta-IR obtained better results. Moreover, compared with baselines of six learning algorithms and six resampling algorithms plus no resampling, totaling 42 (6×7) configurations, Meta-IR outperformed all of them. The code, data, and further information of the experiments can be found on GitHub: <https://github.com/JusciAvelino/Meta-IR>.

4.1 INTRODUCTION

Imbalanced problems are characterized by the low representation of some target values. In classification tasks, an imbalanced dataset is determined by the presence of a class with low representation (minority class) compared to another (majority class) (GANGANWAR, 2012; HAIXIANG et al., 2017). However, in regression problems, the target value is continuous, representing a complex definition. In this context, Ribeiro (2011) proposes the concept of relevance function that determines the relevance of continuous target values to define which examples are rare and normal. Such classification allows verifying the imbalance between the instances called rare and normal. Figure 15 represents the distribution and frequency of examples from an imbalanced dataset (FuelCons). The values on the edges of the graph have low frequency and are considered rare examples.

Figure 15 – Distribution and frequency of the target value Y from the FuelCons dataset. Stronger colors indicate less represented examples (rare examples), while lighter colors indicate more represented examples (normal examples).



Source: Prepared by the author.

In many cases, low-represented values hold significant importance, both to users and prediction processes. For instance, in software engineering, prediction mistakes in large projects are associated with higher development costs (RATHORE; KUMAR, 2017a; RATHORE; KUMAR, 2017b). Similarly, forecasting errors become significantly more costly in meteorological applications when dealing with extreme conditions such as extremely high temperatures (RIBEIRO; MONIZ, 2020). Learning algorithms often struggle with these scenarios, prioritizing frequent target ranges while neglecting rare cases. This limitation can lead to suboptimal predictions for these specific instances.

Few studies explore solutions for handling imbalanced regression problems. Commonly, such

solutions employ resampling strategies such as the Synthetic Minority Oversampling TEchnique for Regression (SmoteR) (TORGO et al., 2013) and WEighted Relevance-based Combination Strategy (WERCS) (BRANCO; TORGO; RIBEIRO, 2019) to balance the training data distributions prior to the training process. Although there is limited research exploring solutions for handling imbalanced regression problems, there are various resampling strategies available to choose from (WU; KUNZ; BRANCO, 2022). Moreover, in a study conducted in Avelino, Cavalcanti e Cruz (2024), it was observed that the selection of resampling strategies can significantly impact regression performance, and the choice of the best resampling strategy depends on the specific imbalanced regression task distribution as well as the regression algorithm used for prediction.

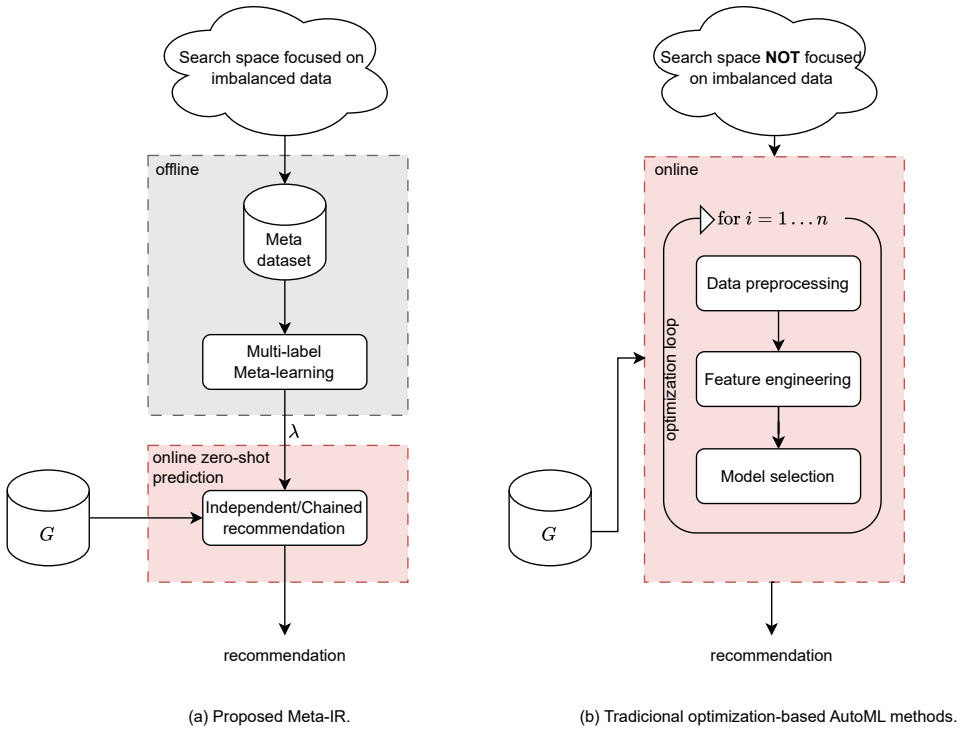
Analyzing this study, we can see that the best learning model changes in nearly 50% of the datasets after applying the resampling algorithm. Thus, an approach that initially selects and optimizes only the best model per dataset, not considering pre-processing algorithms like Naive AutoML (MOHR; WEVER, 2023), may perform poorly. It may overlook the significant impact of resampling techniques on model performance, missing potentially more effective models during the initial selection. Thus, identifying the optimal pipeline is crucial for accurate imbalanced regression results. In a landscape with numerous resampling strategies and regression algorithms, identifying the most fitting pipeline requires potentially evaluating all or a significant subset, thereby leading to a resource-intensive endeavor.

To enhance optimal solution selection, meta-learning (MtL) (VANSCHOREN, 2019; KHAN et al., 2020) models prove effective. These models are based on meta-features, i.e., problem characteristics extracted from data, to learn new tasks more quickly. These meta-features — such as number of examples, number of attributes, number of rare cases, percentage of rare cases and data complexity measures (LORENA et al., 2019) — enable a meta-classifier to correlate them with algorithm performance (target attribute). Once trained, the meta-learning model works in a zero-shot fashion, suggesting algorithms and techniques that best fit the new task's meta-features.

In light of this, we propose a meta-learning-based model — Meta-learning for Imbalanced Regression (Meta-IR)— as an alternative solution for imbalanced regression problems. Meta-IR, illustrated in Figure 16(a), is designed to suggest optimal learning models and resampling strategies based on problem meta-features. The meta-classifier recommends resampling strategy and learning algorithm — a **pipeline** — customized to the new problem's meta-features. For example, it could recommend the Random Over-sampling strategy with the Decision Tree learning model for a given dataset. Alternatively, for a different dataset, it might be more

appropriate to use the WEighted Relevance-based Combination strategy with the Random Forest regression model, demonstrating the need to adjust the strategy and model according to the specific characteristics of the dataset. We propose two ways to train the meta-models: Independent and Chained. The Independent approach involves training the meta-classifiers to indicate the best learning model and resample strategy separately. In contrast, the Chained model is proposed based on the hypothesis that a dependency exists between the resampling algorithm and the regressor. In this way, Chained performs a sequential procedure where the output of one meta-classifier is used as input for another. In other words, the information of the best resampling algorithm is fed into the meta-classifier that predicts the best regressor, and vice versa.

Figure 16 – Comparison between our proposed meta-learning-based model Meta-IR and traditional AutoML.



Source: Prepared by the author.

On the other hand, traditional methods lack specific approaches to handle imbalanced regression problems in the search space. In contrast, our method addresses imbalanced regression problems within the search space. When dealing with a new dataset, traditional methods conduct an online search for the optimal pipeline, resulting in delayed system response. In contrast, our method can recommend the pipeline in a zero-shot prediction. Thus, our method is suitable for imbalanced regression problems and has a significantly lower computational cost.

The experimental analysis, including 218 datasets, six learning models, six resampling strategies, and no resampling, resulting in a total of 42 possible configurations, showed that Meta-IR outperforms the Random recommendation and Majority (always recommends the technique that appears most frequently in the meta-dataset) at the meta-level. At the base-level, comparing the performance achieved by the recommended learning models and resampling strategies, Meta-IR recommendations were better than Random, Majority, and AutoML frameworks — such as Auto-sklearn (FEURER et al., 2015), H2O (LEDELL; POIRIER, 2020), and TPOT (OLSON et al., 2016) — for F1-score for Regression (F1-scoreR). Furthermore, statistical analysis revealed that the proposed method produces statistically better significant results at the base-level. In conclusion, the Meta-IR approach demonstrated an advantage in terms of time efficiency, as it was approximately 50 times faster than the AutoML frameworks. In the analysis of meta-feature importance, it was observed that certain characteristics significantly influenced the recommendation of resampling strategies and learning models. We conducted an interpretation and identified the most important features for these meta-learning tasks, providing insights into which factors are most influential in the recommendation process.

The main contributions of this work are:

- We propose a meta-learning method for recommending pipelines for imbalanced regression problems.
- We introduce two meta-learning recommendation approaches: Independent and Chained. The Independent approach trains meta-classifiers separately to recommend the resampling strategy and regression model, while the Chained approach uses the output of one meta-classifier as input for the next, aiming to enhance overall performance.
- Extensive experiments on 218 imbalanced regression datasets show Meta-IR outperforms state-of-the-art AutoML approaches, established baselines such as Majority and Random selection, and all combinations of evaluated algorithms. Additionally, Meta-IR is significantly faster than other model selection approaches.
- An analysis of meta-feature importance reveals key meta-features crucial for the meta-model's predictive ability in recommending learning models and resampling strategies for imbalanced regression problems.

This work is organized as follows: Section 4.2 presents an overview of the areas of imbalanced regression, automated machine learning and meta-learning. The proposed model is

presented in detail in Section 4.3. Section 4.4 presents the experimental methodology, describing the metadata, meta-features, meta-targets, meta-classifier, and evaluation methodology. In Sections 4.5 and 4.6, we discuss the results and conclude this research.

4.2 BACKGROUND

Imbalanced Regression and AutoML are the two areas of research that have been combined in the proposed method in this work. Therefore, this section provides a global overview of these two areas of research.

4.2.1 Imbalanced Regression

4.2.1.1 Relevance Function

The simultaneous occurrence of two factors characterizes the imbalance problem: i) the disproportionate user preference in the target variable domain; and ii) the insufficient representation, in the available data, of the most relevant cases for the user (BRANCO; TORGO; RIBEIRO, 2017). There is a greater difficulty when the problem occurs in regression tasks, compared to classification tasks, since the continuous target value can have an infinite number of values and the determination of rare cases is not trivial (BRANCO; TORGO; RIBEIRO, 2017). To address this, Ribeiro (2011) proposed the notion of a relevance function ($\phi : Y \rightarrow [0, 1]$). For each dataset, the relevance function maps the target value (Y) into a [0,1] scale of relevance, where 0 and 1 represent the minimum and maximum relevance, respectively.

The relevance function automatically set the significance of data points using the Piecewise Cubic Hermite Interpolating Polynomials (pchip) (DOUGHERTY; EDELMAN; HYMAN, 1989) over a set of control points. These control points can be defined either through domain knowledge or provided by an automated method. Defining control points using an automatic function is preferable since knowledge can often be unavailable (RIBEIRO; MONIZ, 2020). In Ribeiro (2011) the control points are based on Tukey's bloxpot (TUKEY, 1970). The interval proposed by Tukey is based on the adjacent limits [$adj_L = Q1 - 1.5 \cdot IQR$ and $adj_H = Q3 + 1.5 \cdot IQR$] where $Q1$ and $Q3$ are the first and third quartile, respectively, and $IQR = Q3 - Q1$. In turn, the control points are defined by the adjacent limits (adj_L and adj_H) and the median value ($\tilde{Y}$).

Given a dataset D, two sets containing rare (D_R) and normal (D_N) instances are defined considering the relevance threshold (t_R) defined by the user as follows: $D_R = \{\langle \mathbf{x}, y\rangle \in D : \phi(y) \geq t_R\}$ and $D_N = \{\langle \mathbf{x}, y\rangle \in D : \phi(y) < t_R\}$. Full details about the relevant functions can be found in Ribeiro (2011), Avelino, Cavalcanti e Cruz (2024).

4.2.1.2 Resampling Strategies

Strategies for Imbalanced Regression can be categorized into three main groups (AVELINO; CAVALCANTI; CRUZ, 2024): i) Regression models, ii) Learning Process Modification, and iii) Evaluate Metrics. The first group consists of regression models, including both single models and ensembles. The second group highlights additional strategies aimed at refining the learning process to better handle imbalanced target, such as resampling strategies. Finally, the third group focuses on evaluation metrics proposed specific for imbalanced regression. Among these strategies, the most common way to deal with imbalanced data sets is to use resampling strategies, changing the data distribution to balance the targets (MONIZ; BRANCO; TORGO, 2017).

Resampling strategies or balancing algorithms, precede the learning process by altering the distribution of examples through resampling the training data. This method removes samples of normal cases (i.e., under-sampling) or generates synthetic samples for rare cases (i.e., over-sampling). Different resampling strategies have been proposed to deal with imbalanced regression problems. Most of these techniques are based on existing resampling strategies proposed for classification problems (AVELINO; CAVALCANTI; CRUZ, 2024).

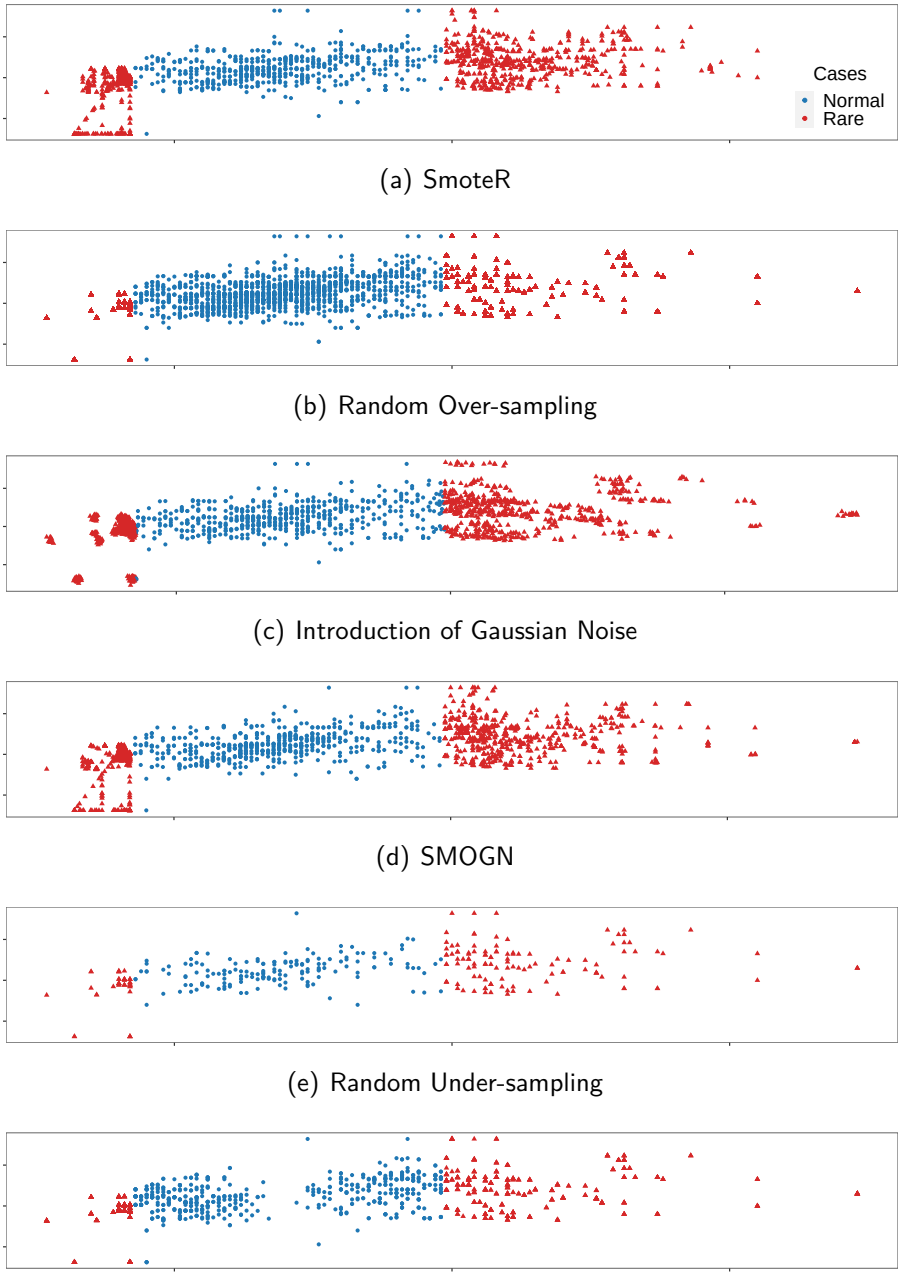
The SmoteR algorithm, which is a variant of Smote (CHAWLA et al., 2002), in which adaptations were made to fit the regression problem, the main ones being: i) definition of rare cases; ii) creation of synthetic examples; and iii) definition of target values for new examples. Also on the basis of the Smote algorithm, Camacho, Douzas e Bacao (2022) proposed Geometric SMOTE, which generates synthetic data points along the line connecting two existing data points. Other strategies adapted from the imbalanced classification are: Random Under-sampling (TORGO et al., 2013) which was based on the idea in Kubat, Matwin et al. (1997), Random Over-sampling (BRANCO; TORGO; RIBEIRO, 2019) proposed for classification in Batista, Prati e Monard (2004), and Introduction of Gaussian Noise (BRANCO; RIBEIRO; TORGO, 2016) adapted from Lee (1999), Lee (2000). In turn, the SmoteR with Gaussian Noise (SMOGN) (BRANCO; TORGO; RIBEIRO, 2017) and WEighted Relevance-based Combina-

tion Strategy (WERCS) (BRANCO; TORGO; RIBEIRO, 2019) strategies were originally proposed for the imbalanced regression problem. Furthermore, Song, Dao e Branco (2022) introduced a distributed version of the SMOGN called DistSMOGN. The method uses a weighted sampling technique to generate synthetic samples for rare cases, in addition to considering the data distribution in each node of the distributed system. For the imbalanced data streams in regression models context, Aminian, Ribeiro e Gama (2021) introduced two sampling strategies (ChebyUS, ChebyOS) based on the Chebyshev inequality to improve the performance of existing regression methods on imbalanced data streams. The approaches use a weighted learning strategy that assigns higher weights to rare cases in order to balance the training process.

Each strategy resamples the data differently. Figure 17 shows this difference, applying the resampling strategies SmoteR, Random Over-sampling, Random Under-sampling, Gaussian Noise Introduction, SMOGN and WERCS to the *FuelCons* dataset. The t_R parameter value was set to 0.8, except for the WERCS, as it operates without a threshold. Standard values were utilized for the other parameters. In the visualization, the Y-axis represented the target values, and the X-axis represented the attribute. In contrast to SmoteR 18(a), which generates synthetic data through oversampling, the Random Over-sampling strategy 18(b) increases data by simply replicating rare cases (data points in darker shade). Another way of over-sampling is through the Introduction of Gaussian Noise, as shown in Figure 18(c), where the generated data differs from the originals, creating more diversity than Random Over-sampling. Similarly, the SMOGN strategy 18(d) combines the Introduction of Gaussian Noise and SmoteR, maintaining a similarity with the SMOTER distribution. However, compared to the Introduction of Gaussian Noise, it becomes evident that the diversity of generated examples is higher in SMOGN. On the other hand, Random Under-sampling 18(e) removes normal data randomly. Furthermore, the WERCS strategy 18(f) combines Random Over-sampling and Random Under-sampling and is the only strategy that does not require a threshold.

Despite the difference, resampling strategies are based on the same principles: decrease normal examples and/or increase rare examples. Under-sampling, which reduces normal examples, is the basis of the Random Under-sampling strategy. On the contrary, over-sampling, which increases rare examples, can be performed simply as in Random Over-sampling or by generating synthetic cases as in the SmoteR Algorithm and Introduction of Gaussian Noise. Other strategies are based on the models already described, such as the SmoteR with Gaussian Noise (SMOGN), which combines the Random Under-sampling strategy and the over-sampling strategies SmoteR and Introduction of Gaussian Noise. And the WEighted Relevance-based

Figure 17 – Distribution of the examples of the FuelCons dataset after applying the resampling strategies, considering t_R=0.8.



(a) SmoteR

(b) Random Over-sampling

(c) Introduction of Gaussian Noise

(d) SMOGN

(e) Random Under-sampling

(f) WERCS

Source: Prepared by the author.

Combination Strategy (WERCS) which combines the Random Under-sampling and Random Over-sampling strategies, using weights for resampling. More details about these strategies can be found in Branco, Torgo e Ribeiro (2019), Torgo et al. (2013), Avelino, Cavalcanti e Cruz (2024).

Resampling strategies have the advantage of allowing the use of any learning algorithm without affecting the model's explainability (BRANCO; TORGO; RIBEIRO, 2019). However, apply-

ing and validating these strategies require effort due to the vast space of solutions. Furthermore, selecting the optimal strategy depends on the dataset, the regression model employed, and the metrics used to assess system performance (AVELINO; CAVALCANTI; CRUZ, 2024). Therefore, given the variety of problems and the increasing interest in this field, there is a need to advance the studies and address the problem of imbalanced regression from another perspective. One way to address the problem is through a recommendation model based on meta-learning. Thus, we propose a model to automate the choice of solutions, i.e., to recommend a pipeline for the problem, with decisions based on data meta-features. It is important to highlight that we are the first to approach this problem from this perspective.

4.2.2 Meta-Learning

The concept of meta-learning (MtL) can be defined as the process of acquiring knowledge through experiences (GIRAUD-CARRIER; VILALTA; BRAZDIL, 2004). Unlike traditional machine learning tasks, in MtL, multiple datasets are used to accumulate experience. In meta-learning, there are three main components (KHAN et al., 2020): meta-features, which represent data characteristics; the meta-learner, which is tasked with inducing knowledge; and the meta-target, which is the learning target.

Meta-features are characteristics extracted from data to describe its properties (RIVOLLI et al., 2022). For each dataset, meta-features are extracted, becoming a meta-example of the meta-dataset. The meta-features are used as input for meta-learner algorithms to select the most appropriate model. There are different ways to extract features from the data (BRAZDIL et al., 2008). Simple and statistical, complexity-based, model-based and landmarkers are commonly used meta-features.

- *Simple and Statistical:* These characteristics are directly extracted from datasets (REIF et al., 2014) and are similar to those used in classification problems, except for those involving the target variable, which needs to be adapted to consider the continuous target value. Simple meta-features include measures such as the number of examples, the number of attributes, the proportion of discrete attributes, the proportion of missing values, and the proportion of outliers. Statistical measures may include, for example, calculations of kurtosis, correlation, and covariance of instances.

- *Data Complexity:* Various studies have used data complexity measures as meta-features. For classification problems, this category is investigated in Cavalcanti, Ren e Vale (2012), Leyva, González e Perez (2014), Garcia, Carvalho e Lorena (2016), Morán-Fernández, Bolón-Canedo e Alonso-Betanzos (2017), Garcia et al. (2018), and for regression in Lorena et al. (2018). Complexity measures have also been adapted for imbalanced classification problems (BARELLA et al., 2018; BARELLA; GARCIA; CARVALHO, 2020; BARELLA et al., 2021). In Lorena et al. (2018), the complexity measures are divided into Feature correlation measures, Linearity measures, Smoothness measures and Geometry, topology and density measures.
- *Model-based:* This measure provides information obtained from learning models. Measures such as mean absolute value and residual variation of a linear regressor were described in Lorena et al. (2018) as complexity-based measures, but because they are derived from models, they can also be considered model-based measures.
- *Landmarking:* These measures are derived from the performance of baseline models on specific datasets (PFAHRINGER; BENSUSAN; GIRAUD-CARRIER, 2000). These measures offer valuable insights into problem difficulty and data characteristics. Their simplicity and computational efficiency make them a quick and informative tool for assessing dataset challenges before engaging in more complex tasks like model selection or hyperparameter optimization. However, while landmarking measures provide a rapid overview, they may not fully capture dataset complexity or accurately evaluate the performance of more advanced models.

Meta-learner is the main component of the meta-learning framework. Meta-learner is the algorithm that models the relationship between dataset characteristics (meta-features) and candidate algorithms (meta-target). The meta-learner receives meta-features as input and recommends appropriate algorithms (KHAN et al., 2020). The most common classifiers used as meta-models are instance-based (BRAZDIL; SOARES; COSTA, 2003; BRAZDIL; SOARES, 2000; GAROUANI et al., 2023) and decision tree-based models (BRAZDIL; SOARES, 2000; AGUIAR et al., 2022; AMORIM; CAVALCANTI; CRUZ, 2024; SOUSA et al., 2016). Instance-based models offer flexibility in adapting and are extensible for new data without need for re-learning (BRAZDIL; SOARES; COSTA, 2003). On the other hand, decision tree-based models provide interpretability, implicitly select important features.

Meta-target there are three meta-target types: Best Algorithm, Ranked List, and Multiple Algorithms (KHAN et al., 2020). 1) Best Algorithm: aims to determine the most appropriate algorithm that achieves the highest performance on a specific task or dataset based on a given metric (e.g., F1-measure); 2) Ranked List: involves evaluating and ranking multiple algorithms based on their performance, permitting practitioners to choose from a list of top-performing options. This method presents a more comprehensive view by considering multiple candidates rather than a singular winner; 3) Multiple Algorithms: provide a set of algorithms that indicate equivalent predicted performance in the given task, indicating no significant difference in their performance. In such a scenario, the user can choose any recommended algorithm.

As far as we know, no meta-learning-based models have been applied to pipeline recommendations for imbalanced regression problems. Therefore, we explored related works that use meta-learning for pipeline recommendation to set our method apart regarding the meta-learning level. Table 12 summarizes these methods. In Moniz e Cerqueira (2021) meta-learning was used for imbalanced classification problems, where a single meta-model is induced to predict the performance of workflow configurations. Their method deals with a particular case of the full model selection formulation (ESCALANTE; MONTES; SUCAR, 2009) and a variant of the popular CASH problem (THORNTON et al., 2013), which involves both workflow selection and hyperparameter optimization combined. In Zagatti et al. (2021), the MetaPrep MtL system is introduced to recommend data preprocessing pipelines, with a focus on four key tasks: imputation, transformation of categorical data to numerical, scaling, and class balancing.

Meta-learning has also been used to suggest algorithms for data that change over time (ROSSI et al., 2014; ROSSI et al., 2021), using one meta-model for each set of examples and one meta-model for each dataset, respectively. Additionally, meta-learning is employed in Amorim, Cavalcanti e Cruz (2024) to build meta-models to automatically select the best Scaling Techniques for a given dataset and classification algorithm. In Aguiar et al. (2022), meta-learning is used in multi-target regression problems to recommend transformation methods and regression models, addressing three types of recommendations: i) both algorithms independently, ii) first the learning model then the transformation method, and iii) first the transformation method then the regression model.

While meta-learning techniques have been explored in various domains, our method – Meta-learning for Imbalance Regression (Meta-IR) – stands out as the only approach specifically

Table 12 – Meta-learning related works

Method	Recommends	Meta-model type
MetaStream (ROSSI et al., 2014)	Learning algorithm	One for each set of examples
ATOMIC (MONIZ; CERQUEIRA, 2021)	Learning algorithm and Resampling strategy	Single
Micro-MetaStream (ROSSI et al., 2021)	Learning algorithm	One for each example
MetaPrep (ZAGATTI et al., 2021)	Imputation, scaling, categorical-numerical transformation and class balancing	Multiple, depends on dataset constraints
Multi-target (AGUIAR et al., 2022)	Multi-target method and base-learner	Two models, one for each label
Meta-Scaler (AMORIM; CAVALCANTI; CRUZ, 2024)	Scaling technique	Multiple, one for each base model
Meta-IR	**Learning algorithm and Resampling strategy**	**Two models, one for each label**

Source: Prepared by the author.

tailored to address the challenge of imbalanced regression tasks by incorporating specialized mechanisms to handle imbalanced data distributions. Meta-IR employs meta-learning to build two models to select the best learning model and resampling strategy automatically in a zero-shot fashion. Regarding the meta-learning level, (AGUIAR et al., 2022) is similar to our proposal. In the same way, we induce two meta-models addressing two recommendation types that we named Independent and Chained.

4.2.3 Automated Machine Learning

Automated Machine Learning (AutoML) is a paradigm that automates the process of constructing and selecting the best machine learning models for a given task, aiming to reduce the effort required to build machine learning models, while also being applicable for parameter optimization (YAO et al., 2018; ZÖLLER; HUBER, 2021; HE; ZHAO; CHU, 2021). There are several search methods strategies employed by the AutoML models, such as Bayesian optimization (HUTTER; HOOS; LEYTON-BROWN, 2011; SNOEK; LAROCHELLE; ADAMS, 2012; GARNETT, 2023), Evolutionary algorithms (BÄCK; FOGEL; MICHALEWICZ, 1997; SIMON, 2013), Gradient-based optimization (BENGIO, 2000), Random search (BERGSTRA; BENGIO, 2012), and Meta-Learning (BRAZDIL et al., 2022b). Table 13 describes some AutoML frameworks that use different forms of optimization.

Table 13 – Related works.

Framework	**Imb**	**MtL**	**Task**	**Search Space**	**Regression Metrics**
Auto-sklearn	×	✓	Both	DP, MS, FE, HO	MedAE, MAE, MSE, R^2
TPOT	×	×	Both	DP, MS, FE, HO	MedAE, MAE, MSE, R^2
LightAutoML	×	×	Both	DP, MS, HO	MSE, RMSE, MAE, R^2
H2O	×	×	Both	MS, HO	MSE, RMSE, RMSLE, MAE, R^2
FLAML	×	×	Both	MS, HO	MAE, MSE, R^2, MAPE
Naive AutoML	×	×	Both	DP, MS, FE, HO	×
ATOMIC	✓	✓	Classification	MS, RS, HO	×
Meta-IR	✓	✓	**Regression**	**MS, RS**	**SERA, F1-scoreR**

Source: Prepared by the author.

Auto-sklearn (FEURER et al., 2015) facilitates the exploration of potential solutions by leveraging meta-learning to warm start the Bayesian optimization. The Bayesian optimization determine the optimal pipeline configuration. TPOT (Tree-based Pipeline Optimization Tool) (OLSON et al., 2016) utilizes genetic programming to optimize machine learning pipelines. Each pipeline represents a sequence of pre-processing steps, feature transformations, and machine learning models. However, the method's primary limitation is the substantial computational resources needed for the optimization process, making it impractical for large datasets or limited computing environments. H2O AutoML (LEDELL; POIRIER, 2020) use a blend of quick random search and stacked ensembles to produce highly competitive outcomes.

The Fast and Lightweight AutoML Library (FLAML) (WANG et al., 2021) uses Estimated cost for improvement (ECI) to learner choices and adopted a randomized direct search method (WU; WANG; HUANG, 2021) to perform cost-effective optimization for cost-related hyperparameters. In LightAutoML framework (VAKHRUSHEV et al., 2021) Tree-structured Parzen Estimators (BERGSTRA et al., 2011) are employed for model fine-tuning. Additionally, warm-starting and early stopping techniques are utilized for optimizing linear models through grid search.

Naive AutoML (MOHR; WEVER, 2023) aims to offer a basic method that acts as a starting point for comparing with more complicated ones. Unlike other approaches that view process steps as interconnected, Naive AutoML simplifies the optimization of each step through random search. It then selects the best algorithm to construct the final process for each step. This method reduces the amount of searching needed, making it faster and more efficient. In terms of execution time, when compared to Meta-IR, Naive AutoML follows a more iterative approach, exploring different components and hyperparameters for each pipeline step. This

process can be time-consuming, necessitating multiple iterations to optimize the model. For example, with 6 learning models, this method first searches for the best model among the 6 before optimizing. Conversely, Meta-IR takes a zero-shot method, recommending both a learning model and a resampling strategy based on the problem's characteristics.

Finally, the Automated Imbalanced Classification method (ATOMIC) (MONIZ; CERQUEIRA, 2021) is the work closest to our proposal, the method uses meta-learning to anticipating the score of the loss criteria, but deal only with imbalance classification tasks. ATOMIC combine the concepts of Complete Model Selection (ESCALANTE; MONTES; SUCAR, 2009) and Combined Algorithm Selection and Hyperparameter optimization (CASH) (THORNTON et al., 2013). In contrast, the Meta-IR utilizes only meta-learning as a recommendation strategy, performing recommendation in zero-shot and thus requiring less computational power, while also featuring an adaptive recommendation phase.

Regarding the search space, Auto-sklearn, TPOT, Naive AutoML address the entire pipeline, recommending Data pre-processing (DP), Model Selection (MS), Feature Engineering (FE), and Hyperparameter Optimization (HO). The LightAutoML recommends nearly the entire pipeline except for Feature Engineering. In contrast, H2O and FLAML specifically concentrate on Model Selection and Hyperparameter Optimization. In the context of imbalanced classification, ATOMIC beyond Model Selection and Hyperparameter Optimization also recommends the Resampling strategy (RS). It is worth noting that none of these methods include resampling strategies for regression in their search space. Conversely, Meta-IR recommends the learning model and resampling strategy for imbalanced regression problems.

Another critical aspect to consider is that the approaches described focus on defining pipelines without considering the imbalance problem, except ATOMIC, which was proposed for the imbalanced classification scenario. As a result, these frameworks do not incorporate specific metrics for imbalanced regression. They employ standard metrics like MedAE, MSE, MAE and R^2 as evaluation metrics. These metrics have limitations when used for evaluation in imbalanced regression problems, as they can deceive the user when the focus is on the accuracy of rare values (MONIZ; TORGO; RODRIGUES, 2014), as they do not consider the relevance of each example. Thus, we hypothesize that employing specialized metrics and methods for imbalanced regression is crucial for improving the performance and reliability of machine learning models in these scenarios. By addressing the relevance of each example and focusing on the accuracy of predictions for rare values, it is possible to achieve a more accurate and fair assessment of model performance.

In this way, our proposal – Meta-IR – differs from previous work in multiple perspectives: (1) As the first framework to deal with imbalanced regression, we recommend learning models and resampling strategies; (2) It is different in how to evaluate the pipelines, using specific metrics for imbalanced regression; (3) Our method is a zero-shot method based on meta-learning, enabling an efficient and adaptive recommendation phase; (4) We employ two different ways in training phase (Independent and Chained).

4.3 PROPOSED METHOD

This section presents a recommendation system that suggests a pipeline (learning models and resampling strategies) for imbalanced regression using meta-learning. Section 4.3.1 introduces the problem and Section 4.3.2 describes all phases of the proposed method.

4.3.1 Problem Definition

The performance of learning models in imbalanced problems can be improved by using strategies that balance the datasets. Therefore, the problem of defining a pipeline is addressed in this work through a meta-learning perspective. The meta-features of the data are used as a criterion for making recommendations. Given a set of datasets $\mathbb{D} = \{\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \ldots, \mathbf{D}_m\}$, a set of learning algorithms $\mathbb{L} = \{l_1, l_2, l_3, \ldots, l_s\}$ and a set of resampling strategies $\mathbb{R} = \{r_1, r_2, r_3, \ldots, r_n\}$, the task is defined as follows:

- **Meta-problem**: the task is to predict the best learning algorithm ($l \in \mathbb{L}$) and the best resampling strategy ($r \in \mathbb{R}$) for an imbalanced regression dataset $\mathbf{D}_i$, $i = \{1, 2, \ldots, m\}$, based on its meta-features ($\mathbf{f}_{\mathbf{D}_i}$).
- **Meta-feature**: each dataset ($\mathbf{D}_i$) is represented by a meta-feature vector $\mathbf{f}_{\mathbf{D}_i} = \{f_1^i, f_2^i, \ldots, f_k^i\}$ composed of k features.
- **Meta-targets**: the meta-targets per dataset $\mathbf{D}_i$ are $[l_i, r_i]$, where $l_i \in \mathbb{L}$ is the learning algorithm and $r_i \in \mathbb{R}$ is the resampling algorithm that are the best fit for $\mathbf{D}_i$.
- **Meta-dataset**: the meta-dataset $\mathbf{M}$ stores the meta-features and the meta-targets of each dataset $\mathbf{D}_i$. $\mathbf{M}$ has m tuples $(\mathbf{f}_{\mathbf{D}_i}, l_i, r_i)$, where $\mathbf{f}_{\mathbf{D}_i}$ is the meta-feature vector, and l_i and r_i are the meta-targets that represent the learning and the resampling algorithms respectively. Thus, $\mathbf{M}$ has m rows and $(k + 2)$ columns; m datasets, k features, and

two meta-targets.

- **Meta-classifiers**: two meta-classifier (λ_L and λ_R) are trained using the meta-dataset $\mathbf{M}$. The meta-classifier λ_L predicts the best learning algorithm $l \in \mathbb{L}$, while the meta-classifier λ_R predicts the best resampling algorithm $r \in \mathbb{R}$ per dataset.

4.3.2 Meta-learning for Imbalanced Regression (Meta-IR)

The Meta-IR method is a meta-learning-based solution to deal with imbalanced regression problems. The main objective of Meta-IR is to facilitate the choice of the learning model and pre-processing strategy for the datasets, given that the conventional workflow for making this choice is extensive.

Meta-IR is divided into three phases: (I) Meta-dataset construction, (II) Meta-classifiers training, and (III) Recommendation, as shown in Figure 18. In the first phase (Meta-dataset construction), the meta-dataset $\mathbf{M}$ is constructed by extracting meta-features and defining the meta-target through pipeline evaluation. Then, the meta-models λ_L and λ_R are trained in the Meta-classifiers training phase to recommend the learning model and resampling strategy, respectively. Finally, in the Recommendation Phase, given an imbalanced regression dataset ($\mathbf{G}$), its meta-features are extracted, and the meta-classifiers generated in the previous phase perform the prediction, recommending a learning model and resampling strategy that are supposedly the best for that dataset. The following sections detail each phase.

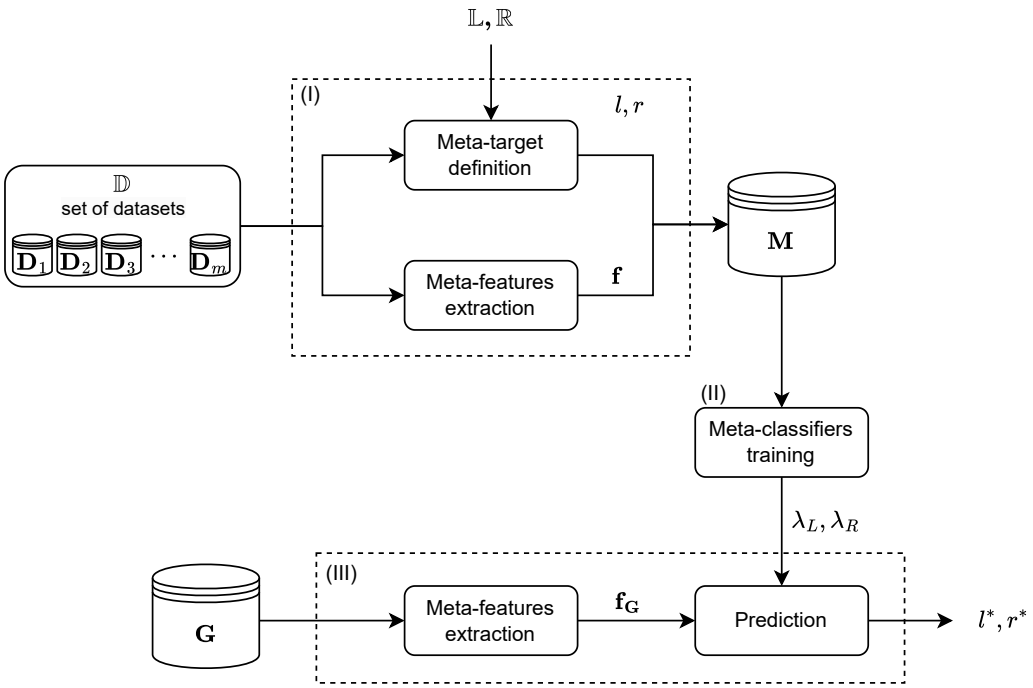
4.3.2.1 Meta-dataset construction phase

This phase has two modules: Meta-feature extraction and meta-target definition. In the Meta-features extraction, each $\mathbf{D}_i \in \mathbb{D}$ is represented by a vector $\mathbf{f}_{\mathbf{D}i} = \{f_1^i, f_2^i, \ldots, f_k^i\}$ with k meta-features. We employ simple and complexity-based meta-features (refer to Appendix B, Table 21, for details), aiming to capture different aspects of the datasets.

In parallel, the Meta-target definition module selects the best learning algorithms ($l_i \in \mathbb{L}$) and sampling techniques ($r_i \in \mathbb{R}$) per $\mathbf{D}_i$, given a performance metric. A total of $|\mathbb{L}| \times |\mathbb{R}|$ pairs of learning algorithms and sampling techniques are evaluated, and the $\mathbf{D}_i$ meta-targets are defined as the pair (l_i, r_i).

The outputs of these two modules are concatenated to compose the meta-dataset $\mathbf{M}$. Thus, $\mathbf{M}$ is composed of m tuples $(\mathbf{f}_{\mathbf{D}i}, l_i, r_i)$, one for each $\mathbf{D}_i$.

Figure 18 – The proposed Meta-IR framework. (I) **Meta-dataset construction:** the meta-features ($\mathbf{f}$) are extracted, and the pipelines are evaluated to construct the meta-dataset $\mathbf{M}$. (II) **Meta-classifiers training:** the two meta-classifiers (λ_L and λ_R) are trained. (III) **Recommendation:** Given the meta-features ($\mathbf{f_G}$) of an imbalanced regression dataset $\mathbf{G}$, this phase recommends a learning model ($l^* \in \mathbb{L}$) and a resampling strategy ($r^* \in \mathbb{R}$).



Source: Prepared by the author.

4.3.2.2 Meta-classifiers training phase

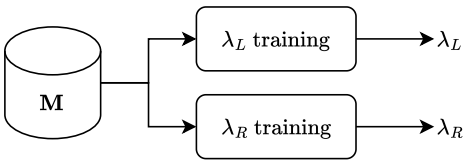
In this phase, two meta-classifiers are trained: λ_L that recommends a learning model ($l_i \in \mathbb{L}$) and λ_R to indicate the resampling strategy ($r_i \in \mathbb{R}$) for an imbalanced regression dataset ($\mathbf{D}_i$). We proposed two training strategies: Independent and Chained.

Independent. Let λ_L and λ_R be learning models that recommend which learning algorithm and resampling strategy should be used for $\mathbf{D}_i$. The λ_L and λ_R models are trained using $\mathbf{M}$ to predict $\mathbf{D}_i$'s best learning algorithm and best resampling strategy. Thus, the final recommendation combines the learning algorithm recommended by λ_L and the resampling strategy recommended by λ_R. The assumption is that the choice of the resampling strategy and learning model is independent and can be performed separately; the best resampling strategy can be chosen independently of the chosen learning model and vice versa. Figure 19(a) illustrates this training strategy.

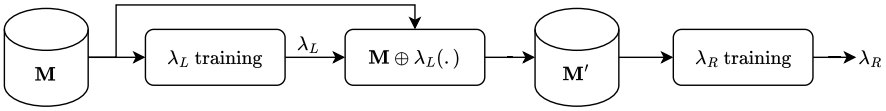
Chained. There are two alternatives for the chaining order: Model first (Figure 19(b)),

and Strategy first (Figure 19(c)). In the Model First approach, λ_L is trained using $\mathbf{M}$, and the model's recommendation $l \in \mathbb{L}$ is added to the meta-dataset as a meta-feature, generating $\mathbf{M}'$. After, the model λ_R is trained using $\mathbf{M}'$ to recommend the resampling strategy. The assumption is that the choice of the learning model helps to select the resampling strategy. In the Strategy First approach, λ_R is trained using $\mathbf{M}$, and the model's recommendation $r \in \mathbb{R}$ is added to the meta-dataset as a meta-feature, generating $\mathbf{M}'$. The λ_L model is then trained to recommend the learning model using $\mathbf{M}'$. The assumption is that the choice of resampling strategy helps to select the learning model.

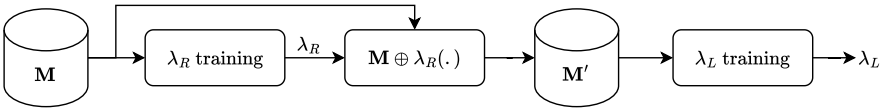
Figure 19 – Training of the meta-classifiers λ_L and λ_R.



(a) Independet



(b) Model first



(c) Strategy first

Source: Prepared by the author.

4.3.2.3 Recommendation Phase

Given an imbalanced regression dataset $\mathbf{G}$, this phase recommends a learning model ($l^* \in \mathbb{L}$) and a resampling strategy ($r^* \in \mathbb{R}$). The "Meta-features extraction" module (Phase III in Figure 18) represents $\mathbf{G}$ as a vector of meta-features $\mathbf{f_G}$. After, $\mathbf{f_G}$ is used as input to the meta-classifiers λ_L and λ_R that return the learning model l^* and the resampling strategy r^*. The type of recommendation is the best algorithm, where only the model and strategy with

the highest performance are recommended.

4.4 EXPERIMENTAL METHODOLOGY

4.4.1 Datasets

The datasets were taken from the OpenML repository (VANSCHOREN et al., 2014). After searching only for regression problems on OpenML, a total of 660 datasets was returned. Subsequently, the datasets having less than 2.0% of rare cases were removed, resulting in 200 datasets. Additionally, we included 18 datasets from Moniz, Branco e Torgo (2017).

Thus, the evaluation study is performed on $|\mathbb{D}| = 218$ datasets. An important aspect of these datasets is their diversity in terms of number of examples varying between 27 and 20.640, the number of features ranging from 5 to 1024, and percentage of rare examples varying from 2.0% to 39.6%. This diversity is crucial in the meta-learning process, as it allows for the generalization of the model, making it more flexible and capable of dealing with a greater variety of datasets (refer to Appendix A, Table 5.1, for more details).

4.4.2 Meta-features

Each dataset $\mathbf{D}_i$ in the meta-dataset is represented by a vector of meta-features $\mathbf{f}_{\mathbf{D}i}$. Forty-three meta-features were considered, with measures from different categories: simple information about the dataset and complexity measures proposed in Lorena et al. (2018). The simple category includes the number of examples, the number of attributes, the number of rare cases, and the percentage of rare cases. Regarding complexity measures, data set distribution, correlation between attributes and targets, performance metrics related to linear regression, and data smoothness are considered (refer to Appendix B for details). The ECoL package (LORENA et al., 2018; LORENA et al., 2019) was used to extract complexity measures.

4.4.3 Learning Algorithms

The learning algorithms set $\mathbb{L}$ is composed of six models from different families (single and ensemble): Bagging (BG), Decision Tree (DT), Multilayer Perceptron (MLP), Random Forest (RF), Support Vector Machine (SVM), and XGBoost (XG). These models were previ-

ously employed in Avelino, Cavalcanti e Cruz (2024), where their sensitivity to the selection of resampling strategies was demonstrated.

4.4.4 Resampling strategies

The resample strategies set $\mathbb{R}$ comprises six strategies plus no resampling (NONE). The strategies, previously described in Section 4.2.1.2, are SmoteR (SMT), Random Over-sampling (RO), Random Under-sampling (RU), Introduction of Gaussian Noise (GN), SMOGN (SG) and WEighted Relevance-based Combination Strategy (WERCS).

4.4.5 Meta-Models

We select classification algorithms for the meta-models, focusing on models based on Decision Trees. These algorithms are renowned for their interpretability, widespread adoption in classification tasks, and consistently strong performance across diverse problem domains (HASTIE et al., 2009). Besides, Decision Trees requires no scaling data transformation (de Amorim; CAVALCANTI; CRUZ, 2023). Our chosen models include Random Forest (RF)[1] (BREIMAN, 2001), Extra trees (ET)[1] (GEURTS; ERNST; WEHENKEL, 2006), Bagging (BG)[1] (BREIMAN, 1996), k-Nearest Oracles-Eliminate (Knora-E)[2] (KO; SABOURIN; JR, 2008), k-Nearest Oracles-Union (Knora-U)[2] (KO; SABOURIN; JR, 2008), XGBoost (XG)[3] (CHEN; GUESTRIN, 2016) e DES-MI[2] (GARCÍA et al., 2018).

These classification models have also been used in other meta-learning tasks (AGUIAR et al., 2019; AMORIM; CAVALCANTI; CRUZ, 2024; SOUSA et al., 2016). We employed the default configurations for all models. After comparing these models, we found that Random Forest (RF) outperformed the others, achieving superior results (refer to Appendix C, Tables 22 and 23, for detailed comparisons). As a result, we selected this model for the Meta-IR analyses.

4.4.6 Evaluation Methodology

The Meta-IR is evaluated using the leave-one-dataset-out method. This cross-validation technique involves using each dataset as the test set in an iteration, and the other datasets

[1] `scikit-learn` - V. 1.2.2
[2] `deslib` - V. 0.3.7
[3] `xgboost` - V. 1.0.2

are used to compose the meta-dataset (VANSCHOREN, 2019). Since we have 218 datasets, for each iteration, the meta-dataset is constructed using $|\mathbb{D}| = 217$ datasets, while the test $\mathbf{G}$ has only one dataset.

The proposal is evaluated at the meta-level and base-level. The meta-level focuses on evaluating the meta-classifiers for recommending the resampling strategy and regressor, with the two proposed meta-classifier strategies, Independent and Chained, also being evaluated. The evaluation at this level utilizes the F1-macro (classification) metric. At the base-level, performance is evaluated on the recommendations of the meta-classifiers from the meta-level, using the F1-scoreR (TORGO; RIBEIRO, 2009) and Squared error-relevance area (SERA) metric (RIBEIRO; MONIZ, 2020). These metrics were chosen because they aim to effectively evaluate model performance for extreme value predictions while being robust to model bias. Furthermore, we also analyze the proposed method's execution time and compare it to the state-of-the-art AutoML methods.

The proposed method is compared at the meta-level with two baseline methods: Random and Majority. The Random model randomly selects a class label from the available labels. Let $\mathbb{L}$ and $\mathbb{R}$, where $\mathbb{L}$ contains the learning models and $\mathbb{R}$ the resampling algorithms. For each instance of the meta-dataset, the random recommendation model randomly chooses one instance from $\mathbb{L}$ and one from $\mathbb{R}$. The Majority model selects the class that appears most frequently. In other words, this model chooses the class label most frequently in $\mathbb{L}$ and $\mathbb{R}$. For the F1-scoreR metric, the resampling strategy most represented is WERCS, and the learning model is DT. For SERA is WERCS and RF (refer to Appendix D, Figure 27, for details). These baselines are commonly employed to emphasize the necessity of a recommendation system (BRAZDIL et al., 2008).

At the meta-level, Meta-IR is evaluated from three perspectives: i) Comparison with each pipeline; ii) Comparison with AutoML frameworks such as Auto-sklearn, H2O, TPOT, FLAML, LightAutoML, NaiveAutoML and the baselines Random and Majority; iii) Meta-IR as a pre-processing step for AutoML frameworks.

The comparison of each pipeline demonstrates that no single combination is capable of achieving more wins than Meta-IR. This indicates that there is no one-size-fits-all solution for all datasets. This highlights the importance of our method, which determines the pipeline according to the characteristics of the problem. On the other hand, the comparison with AutoML frameworks aims to demonstrate that Meta-IR achieves better results than these methods in imbalanced regression scenarios. Finally, by using Meta-IR output as a pre-processing step

for AutoML frameworks, we highlight the importance of incorporating resampling strategy recommendation into AutoML methods to address imbalanced problems.

4.5 RESULTS AND DISCUSSION

The evaluation of Meta-IR is divided into three perspectives: Meta-Level Analysis, Base-level Analysis and Execution time analysis. In the Meta-Level Analysis (Section 4.5.1), the meta-models performance is evaluated and compared with two baseline approaches: Random and Majority. In Section 4.5.2, we present the Base-level Analysis, where we evaluate the recommended learning models and resampling strategies. In this analysis, at the base-level, we compare the recommendation of Meta-IR with Random, Majority, and the AutoML frameworks Auto-sklearn, H2O, TPOT, FLAML, LightAutoML and NaiveAutoML. Furthermore, in this section, we evaluate the Meta-IR as a pre-processing step for AutoML frameworks. The third analysis presented in Section 4.5.3, where we consider the execution time of Meta-IR and each AutoML framework. Finally, in Section 4.5.4, an analysis of the meta-features is conducted.

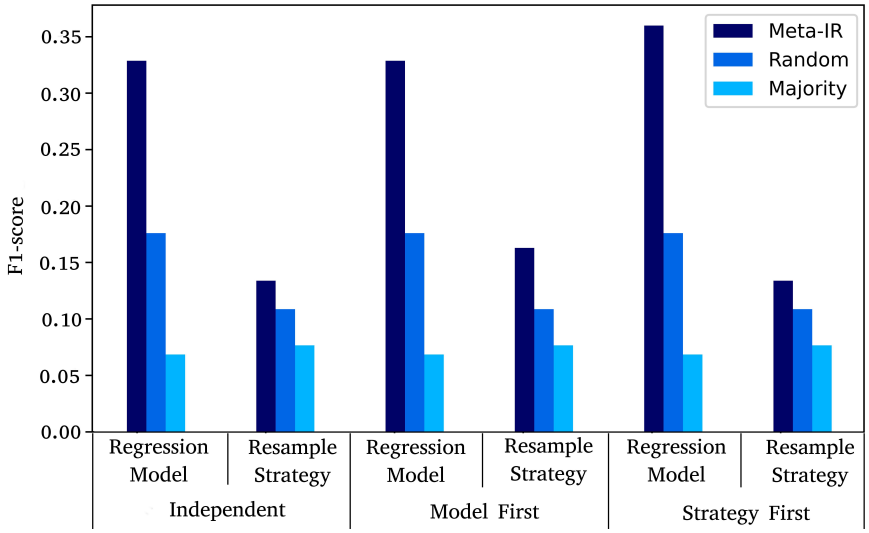
4.5.1 Meta-level analysis

The meta-level refers to evaluating the meta-model performance, that is, the effectiveness exhibited by the meta-model in predicting the meta-target for the datasets. The Meta-IR performance is compared with the performance of Random and Majority models, considering the F1-macro metric. These results are illustrated in Figure 20 (a-b), considering F1-scoreR and SERA metrics as optimization functions. The optimization function refers to a specific metric used to guide the optimization process. It defines the objective that the model aims to maximize (F1-scoreR) or minimize (SERA) during the learning process. The F1-scoreR and SERA metrics were chosen as optimization functions because they are well-suited for capturing the performance of models in the imbalanced regression problem.

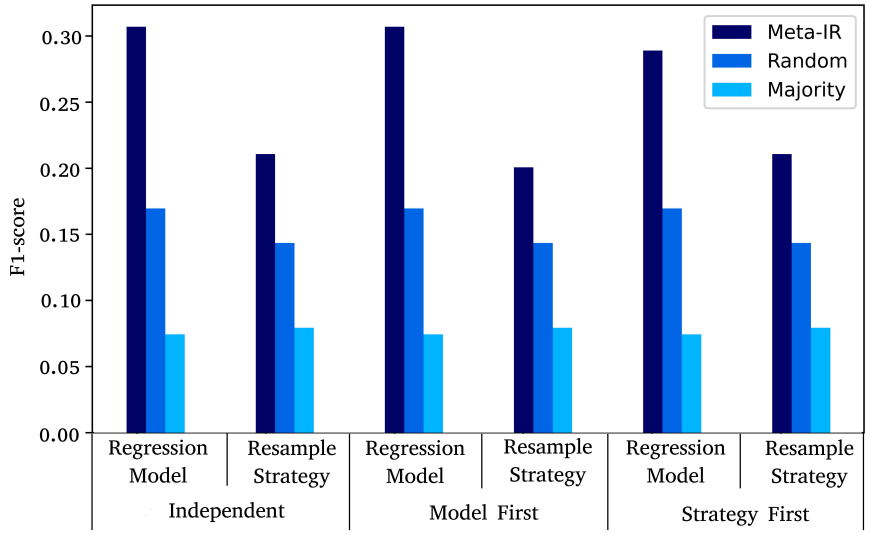
The results are presented for each recommendation procedure. Meta-IR obtained better results recommending the learning model and resampling strategy than Random and Majority for all types of training in both optimization functions.

Another analysis we can extract from Figure 20 is about the training strategy. It is observed that the order in which the model and resampling strategy are recommended affects the recommendation. For the F1-scoreR, the Independent model achieved a lower F1-macro

Figure 20 – Meta-IR performance for each type of training (Independent, Model First, and Strategy First) compared with the baselines Random and Majority.



(a) F1-scoreR



(b) SERA

Source: Prepared by the author.

score compared to the Chained-trained model. Specifically, the Independent model attained an F1-macro score of 0.33 for recommending the learning model, while the Chained model (Strategy first) achieved 0.36, indicating that recommending the resampling strategy first led to an increase in the F1-macro score for recommending the learning model. Regarding the recommendation of the resampling strategy, the Independent model scored 0.14, while the Chained model (Model first) scored 0.16, showing that recommending the learning model first led to an increase in the F1-macro score for recommending the resampling strategy.

On the other hand, when we evaluate considering the SERA metric, the behavior is different. The use of the Chained model for training has a negative impact on the recommendation of the learning model and resampling strategy. The Independent model attained an F1-macro score of 0.31 for recommending the learning model, while the Chained model (Strategy first) achieved 0.21. Regarding the recommendation of the resampling strategy, the Independent model scored 0.21, while the Chained model (Model first) scored 0.20.

In conclusion, the analysis emphasizes the importance of the training strategy on the performance of the Meta-IR meta-model, especially in how the order of recommendations affects the outcomes. The Chained approach (Strategy first and Model first) improved the F1-macro score for recommending the learning model and the resampling strategy, but it was less effective when evaluated using the SERA metric. These findings highlight the importance of selecting an appropriate training strategy based on the evaluation metric used, as it can significantly impact the effectiveness of model and resampling strategy recommendations for addressing imbalanced regression problems.

4.5.2 Base-level analysis

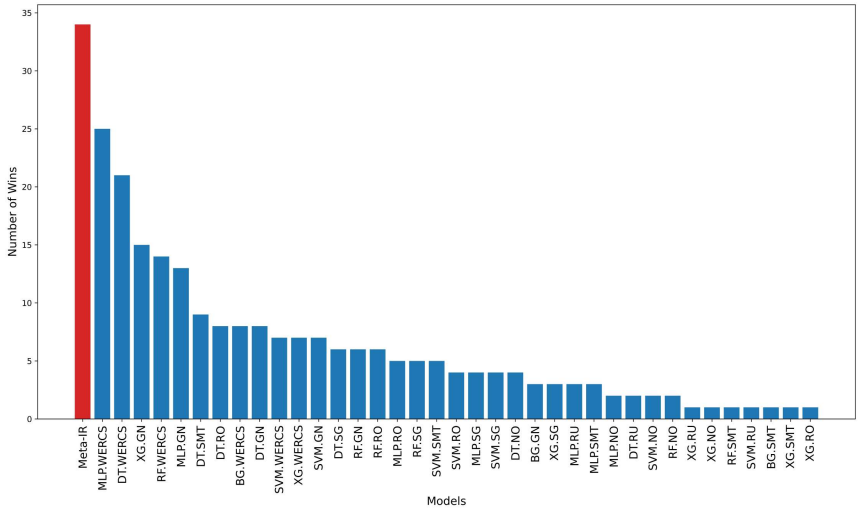
The Base-level Analysis assesses the learning models and resampling strategies recommended by Meta-IR. We conducted the following analysis: (1) Meta-IR compared to each combination of sampling strategy and learning model (Section 4.5.2.1); 2) Meta-IR compared to Random, Majority and the AutoML frameworks: Auto-sklearn, H2O, TPOT, FLAML, LightAutoML and NaiveAutoML (Section 4.5.2.2).

4.5.2.1 Comparison with each pipeline

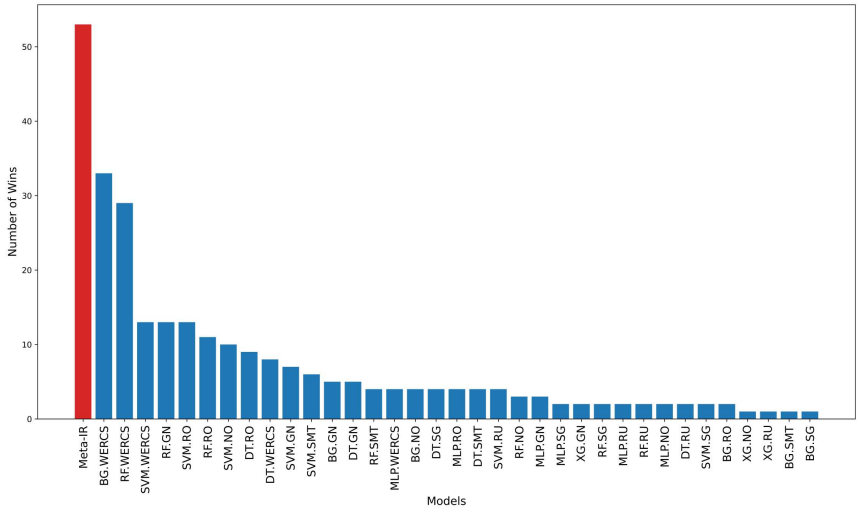
Figure 21 shows how many times each model achieved the best result among all evaluated combinations of resampling strategies and learning algorithms. These results represent the oracle, showcasing the best possible outcomes for each dataset. The number of wins summation per figure is equal to 218, which is the number of datasets in the used corpus. The results indicate that Meta-IR outperformed all other strategies for both metrics, indicating that no single combination consistently achieves a significant number of wins compared to Meta-IR. Therefore, these outcomes highlight the potential of the Meta-IR model and the importance of integrating Meta-IR into the pipeline recommendation process for imbalanced regression

problems.

Figure 21 – Number of wins over all datasets: Meta-IR *versus* each pipeline (resampling strategy and regressor).



(a) F1-scoreR



(b) SERA

Source: Prepared by the author.

Tables 14 and 15 show the p-values of the Wilcoxon signed rank test, illustrating the significance of performance discrepancies between Meta-IR and each specific combination. There are 42 combinations in total (6 learning algorithms $\times$ 7 resampling strategies). For the F1-scoreR, Meta-IR achieved statistically significantly better results, except for DT.GN and DT.RO. It is crucial to highlight that the Decision Tree Regressor (DT) was the model that appears more frequently as the best one, and it was defined as the Majority model as described in Section 4.4.6. Meta-IR consistently performed significantly better regarding the SERA metric, with p-values < 0.001. These findings suggest that Meta-IR consistently

outperforms all combinations.

Table 14 – Pairwise comparison between the performances of the Meta-IR *versus* each pipeline (resampling strategy and regressor) over all datasets. The values represent the p-value of the Wilcoxon signed-rank test using the F1-scoreR metric. All results with $\alpha < 0.05$ are in bold.

Resampl Strat	NONE	GN	RO	RU	SG	SMT	WERCS
				p-value			
BG	**5.65e-24**	**7.06e-13**	**3.79e-13**	**5.08e-26**	**1.73e-18**	**4.98e-16**	**2.95e-11**
DT	**2.04e-5**	0.323	0.177	**9.65e-8**	**0.009**	**0.012**	**0.049**
MLP	**4.13e-17**	**3.18e-8**	**1.86e-11**	**5.73e-15**	**3.48e-10**	**9.44e-13**	**1.06e-8**
RF	**8.07e-18**	**9.26e-10**	**5.94e-14**	**3.54e-16**	**8.22e-16**	**8.18e-15**	**4.86e-8**
SVR	**1.53e-16**	**1.78e-6**	**3.32e-13**	**1.20e-17**	**9.85e-11**	**1.20e-7**	**3.35e-10**
XG	**7.99e-18**	**4.46e-11**	**2.34e-16**	**4.02e-16**	**2.91e-15**	**9.66e-17**	**5.15e-13**

Source: Prepared by the author.

Table 15 – Pairwise comparison between the performances of the Meta-IR *versus* each pipeline (resampling strategy and regressor) over all datasets. The values represent the p-value of the Wilcoxon signed-rank test using the SERA metric. All results with $\alpha < 0.05$ are in bold.

Resampl Strat	NONE	GN	RO	RU	SG	SMT	WERCS
				p-value			
BG	**3.81e-15**	**5.77e-15**	**6.50e-16**	**4.45e-19**	**1.52e-18**	**1.11e-16**	**2.24e-14**
DT	**9.81e-21**	**1.75e-15**	**3.24e-18**	**7.57e-23**	**1.02e-21**	**1.03e-20**	**4.32e-23**
MLP	**1.68e-25**	**1.01e-28**	**2.16e-25**	**1.16e-27**	**3.28e-29**	**1.35e-25**	**1.96e-23**
RF	**3.19e-7**	**4.70e-7**	**1.17e-3**	**2.81e-12**	**9.16e-10**	**4.84e-7**	**2.88e-6**
SVR	**5.60e-14**	**8.58e-13**	**2.67e-14**	**4.21e-18**	**6.38e-16**	**2.22e-16**	**6.14e-15**
XG	**1.11e-29**	**1.42e-26**	**1.07e-29**	**1.30e-28**	**1.02e-28**	**7.87e-30**	**4.44e-29**

Source: Prepared by the author.

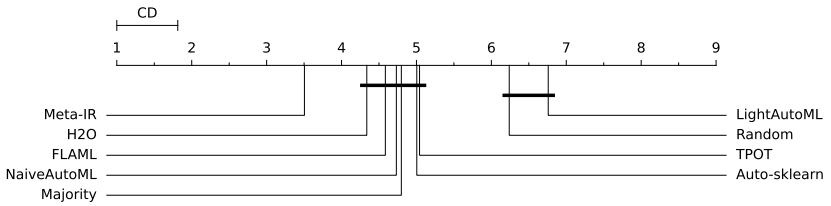
4.5.2.2 Comparison with AutoML frameworks

The comparison of Meta-IR and various AutoML frameworks such as Auto-sklearn, H2O, TPOT, FLAML, LightAutoML and NaiveAutoML, as well as the baselines Random and Majority based on two metrics: F1-scoreR and SERA, is presented in Figure 22(a-b). To verify which results are statistically different, the post-hoc Nemenyi test was applied. The presented Critical Difference diagrams (DEMŠAR, 2006) for each model consider both metrics. The horizontal line demonstrates the significance of the difference between the models. Models that are not connected present a significant difference (p-value < 0.05) compared to the other methods.

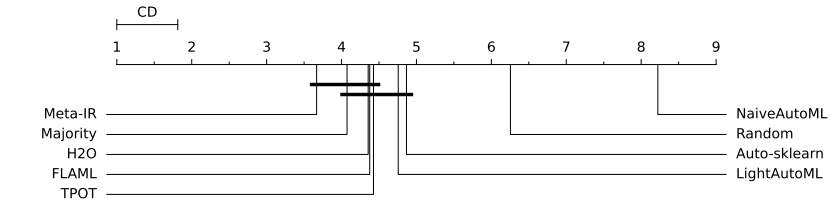
For both the F1-ScoreR and SERA metrics, Meta-IR demonstrates the best results. However, for the SERA metric, there is no significant difference when compared to the Majority,

H2O, FLAML and TPOT methods. Despite this, Meta-IR stands out when compared to other methods. Therefore, we can conclude that there are benefits to using an MtL-based model for recommending learning models and resampling strategies.

Figure 22 – Critical Difference diagrams obtained from the recommendations given by the models.



(a) F1-scoreR



(b) SERA

Source: Prepared by the author.

To better understand the behavior of the proposed model compared to the AutoML frameworks concerning the SERA metric, we evaluated it across different percentages of rare cases. By utilizing percentiles, we segmented the data into three categories of rare rare cases: low, medium, and high. The low level consists of 45 datasets with rare case percentages below 8.2, while the medium level consists of 78 datasets with percentages above 8.3 and below 15.2. Lastly, the high level consists of 95 datasets with percentages exceeding 15.2. We performed a pairwise comparison of the performance of the Meta-IR model and AutoML frameworks for each category (Table 16). At the high level of rare cases, Meta-IR outperformed all other AutoML frameworks, with p-values below the significance threshold ($\alpha < 0.05$). However, at the medium and low rare case levels, there were no statistically significant differences between the performance of Meta-IR and the AutoML frameworks, as indicated by p-values above 0.05. These findings suggest that while Meta-IR excels, particularly in scenarios with high rare cases, its performance is comparable to other AutoML frameworks in datasets with lower levels of rarity.

For the F1-scoreR metric, the Meta-IR consistently outperforms all other AutoML frameworks when using the F1-scoreR metric. This demonstrates the Meta-IR's effectiveness in handling rare instances. Moreover, when considering a smooth function that weights each ex-

Table 16 – Pairwise comparison between the performances of the Meta-IR *versus* AutoML frameworks at each level of percentage of rare cases (High, Medium and Low). The values represent the p-value of the Wilcoxon signed-rank test using the SERA metric. All results with $\alpha < 0.05$ are in bold.

	Auto-sklearn	H2O	TPOT	FLAML	LightAutoML	NaiveAutoML
High	**5.070e-6**	**1.829e-4**	**7.714e-5**	**0.001**	**1.502e-7**	**3.185e-16**
Medium	0.387	0.558	0.638	0.167	0.775	**3.879e-8**
Low	0.376	0.474	0.947	0.991	0.308	**6.655e-8**

Source: Prepared by the author.

ample based on a relevance function without explicitly distinguishing between rare and normal cases (SERA metric), the advantages of Meta-IR become more apparent, particularly in scenarios with higher percentages of rare cases. This suggests that the Meta-IR's ability to weigh the importance of rare cases effectively makes it advantageous in situations where rare instances significantly impact the overall performance. Therefore, Meta-IR is an effective solution for tasks where addressing rare cases is essential.

Another comparison between Meta-IR and AutoML frameworks is made in terms of wins, ties, and losses. The results, which are presented in Table 17, show that Meta-IR consistently outperformed the AutoML frameworks across various datasets, achieving a minimum victory percentage of 57.80% and a maximum of 75.69%. These percentages underscore Meta-IR's superiority over AutoML approaches for handling imbalanced regression, highlighting its effectiveness across diverse datasets and reinforcing its potential as a robust solution for imbalanced regression tasks.

Table 17 – Win/tie/loss of the Meta-IR versus AutoML frameworks.

	Auto-sklearn	**H2O**	**TPOT**	**FLAML**	**LightAutoML**	**NaiveAutoML**
F1-scoreR	143/0/75	143/0/75	147/2/69	145/0/73	165/4/49	132/4/82
SERA	135/0/83	128/0/90	129/0/89	126/0/92	142/0/76	199/0/19

Source: Prepared by the author.

Those findings are promising and showcase the potential of Meta-IR as a competitive alternative in automated machine learning, specifically in imbalanced regression problems. Furthermore, one notable disadvantage of AutoML frameworks is their lack of specific metrics for imbalanced regression as an optimization function. Traditional regression metrics may not adequately capture the model's performance. Unlike Meta-IR, which explicitly addresses imbalanced regression, AutoML frameworks have limited optimization functions, and the lack of resampling strategies can be a limiting aspect when dealing with imbalanced data.

4.5.2.3 Meta-IR as a pre-processing step for AutoML frameworks

This section explores the advantages of using the Meta-IR method in conjunction with AutoML frameworks for imbalanced regression tasks. While previous sections have demonstrated the effectiveness of Meta-IR, here we focus on its role as a pre-processing step to enhance the performance of AutoML frameworks. We used the recommendations from the resampling strategies provided by the Independent Meta-IR model and applied the recommendation as pre-processing before the AutoML models.

Table 18 compares different AutoML frameworks based on their F1-scoreR and SERA mean performance before and after the Meta-IR pre-processing step. The column Δ shows the percentage improvement from using Meta-IR. Regarding the F1-scoreR, all AutoML frameworks improve with Meta-IR. LightAutoML shows the highest improvement of 16.50%, followed by FLAML with 8.64%. However, the SERA metric results vary. Auto-sklearn and TPOT have significantly increased SERA values (1008.41% and 42.84%, respectively), while H2O, NaiveAutoML, and LightAutoML have substantially decreased SERA values, indicating improved performance, with reductions of -82.87%, -28.37% and -10.00%, respectively.

Table 18 – Meta-IR as a pre-processing step and Comparison with Baseline

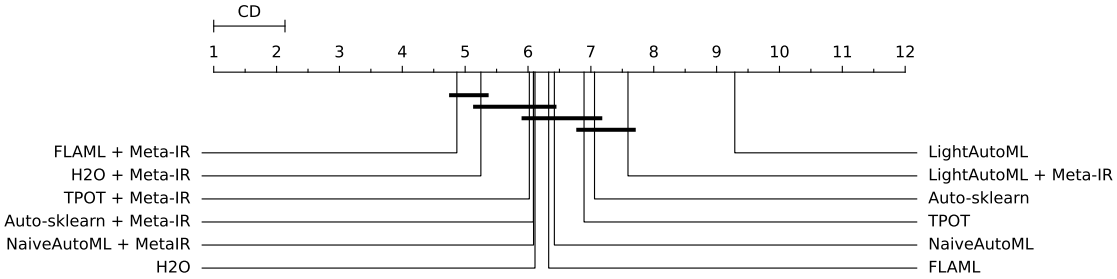
	Baseline	Meta-IR	Δ (%)	Baseline	Meta-IR	Δ (%)
	↑ **F1-scoreR**			↓ **SERA**		
Auto-sklearn	0.246	0.258	4.88	1.485e+10	1.645e+11	1008.41
H2O	0.248	0.259	4.44	2.219e+11	3.801e+10	-82.87
TPOT	0.244	0.258	5.74	2.116e+10	3.022e+10	42.84
FLAML	0.243	0.264	8.64	1.875e+10	2.355e+10	25.60
LightAutoML	0.206	0.240	16.50	3.132e+10	2.819e+10	-10.00
Naive AutoML	0.264	0.277	4.92	6.388e+10	4.575e+10	-28.37

Source: Prepared by the author.

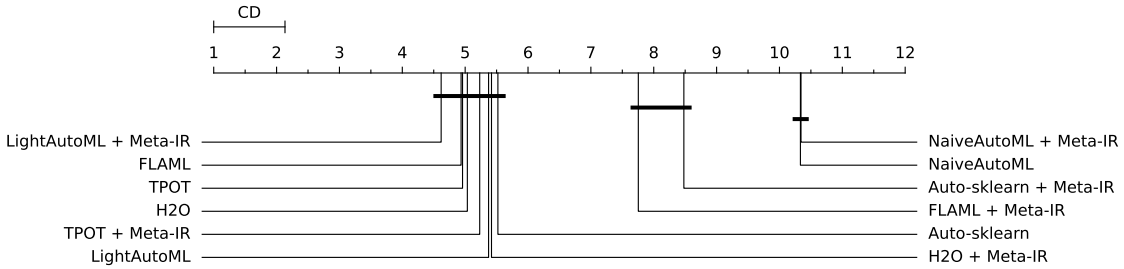
In Figure 23, critical difference diagrams illustrate the performance of the F1-scoreR and SERA metrics. Regarding F1-scoreR, the results reveal significant insights into the performance improvements the Meta-IR approach brings when combined with various AutoML methods. Notably, combinations like FLAML + Meta-IR, H2O + Meta-IR, and Auto-sklearn + Meta-IR consistently rank higher than their baseline. This suggests that the Meta-IR technique effectively enhances the predictive capabilities and overall F1-scoreR performance of these

AutoML frameworks. On the other hand, considering the SERA metric, the combination of LightAutoML with Meta-IR achieves the highest performance, significantly outperforming other methods, including its baseline.

Figure 23 – Critical Difference diagrams obtained from the recommendations given by the models.



(a) F1-scoreR



(b) SERA

Source: Prepared by the author.

In summary, the analysis of Meta-IR as a pre-processing step shows its potential to improve the performance of various AutoML frameworks significantly. There were noticeable enhancements in F1-scoreR for all tested methods, especially for LightAutoML and FLAML. These results suggest that Meta-IR effectively enhances the predictive capabilities of these frameworks. However, evaluating the SERA metric provides a different perspective, revealing substantial improvements with Meta-IR for LightAutoML, H2O and NaiveAutoML, while other methods yield mixed results. This indicates that the impact of Meta-IR on SERA varies by method, necessitating further exploration to understand the relationship. Overall, the positive improvements in F1-scoreR underscore the value of Meta-IR in enhancing model performance. Nonetheless, carefully considering its effects on different metrics is crucial to maximize its benefits in practical applications.

4.5.3 Execution time analysis

This section evaluates the execution time of Meta-IR against AutoML frameworks. The Meta-IR process involves extracting meta-features and recommending a pipeline and for all AutoML frameworks was given a maximum budget of 1 hour. However, it is important to emphasize that AutoML models can produce output before the 1-hour mark. A comparison was made using ten datasets (see Table 19) to determine whether Meta-IR offers a time-efficient option compared to AutoML frameworks. This assessment aims to highlight the scalability and potential time-saving benefits of the proposed approach.

Table 19 – Data sets used to calculate time.

Dataset	n.samples	n.attributes	n.rare	p.rare
cpu-act	209	36	33	15.8
a1	198	11	28	14.1
auto93	93	57	11	11.8
QSAR-TID-10929	154	1024	18	11.7
heat	7400	11	664	9.0
fri-c4-1000-10	1000	10	89	8.9
compactiv	8192	21	713	8.7
sleuth-case2002	147	6	12	8.2
space-ga	3107	6	173	5.6
wind	6574	14	283	4.3

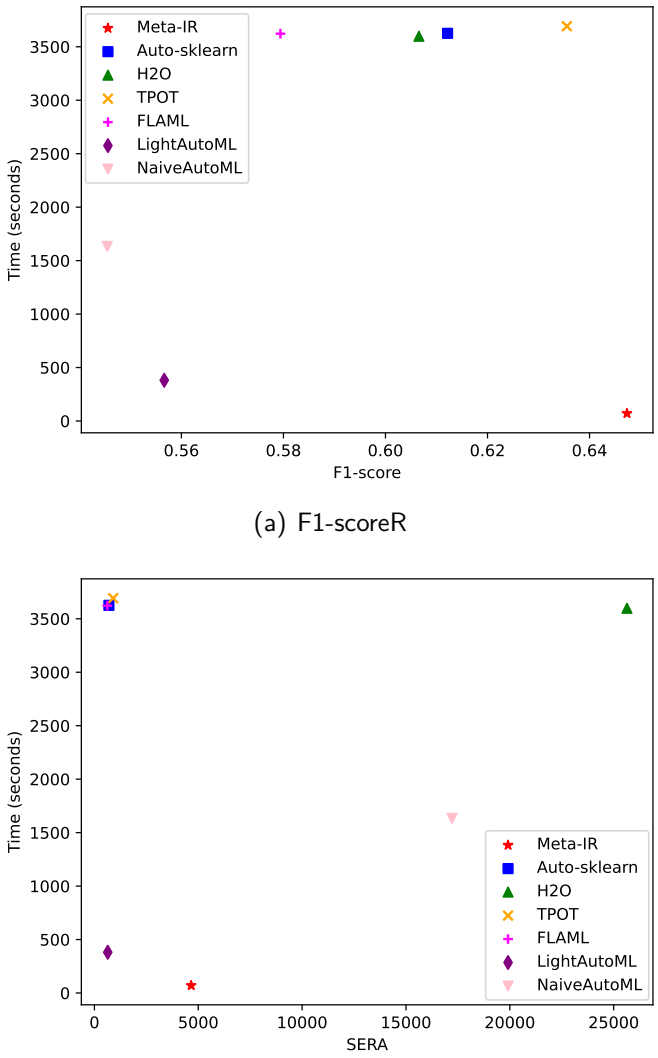
Source: Prepared by the author.

Figure 24 illustrates the relationship between time and performance for each of the models considered. Meta-IR outperforms all the AutoML frameworks in terms of processing speed. This finding suggests that the proposed method is efficient and scalable. However, it is essential to remember that execution times can vary considerably based on characteristics such as the input data size and the number of parameters evaluated for each model and strategy.

4.5.4 Meta-features analysis

This analysis explored the importance of meta-features for the Meta-IR meta-model. We determined their importance using the `feature_importance` attribute of the Random Forest model, which measures the relative influence of each meta-feature on the model's predictive ability. The feature importance is determined based on Gini impurity (BREIMAN, 2017), providing a measure of how much each meta-feature contributes to the overall performance of

Figure 24 – Time and performance of each model.



(a) F1-scoreR

(b) SERA

Source: Prepared by the author.

the model.

In Figures 25 and 26, we plot the feature importance (x-axis) of all meta-features (y-axis) for the meta-models trained for both metrics (F1-scoreR and SERA) as optimization function. The plot displays the features importance with a gradient color scheme, where lighter shades indicate lower importance and darker shades indicate higher importance.

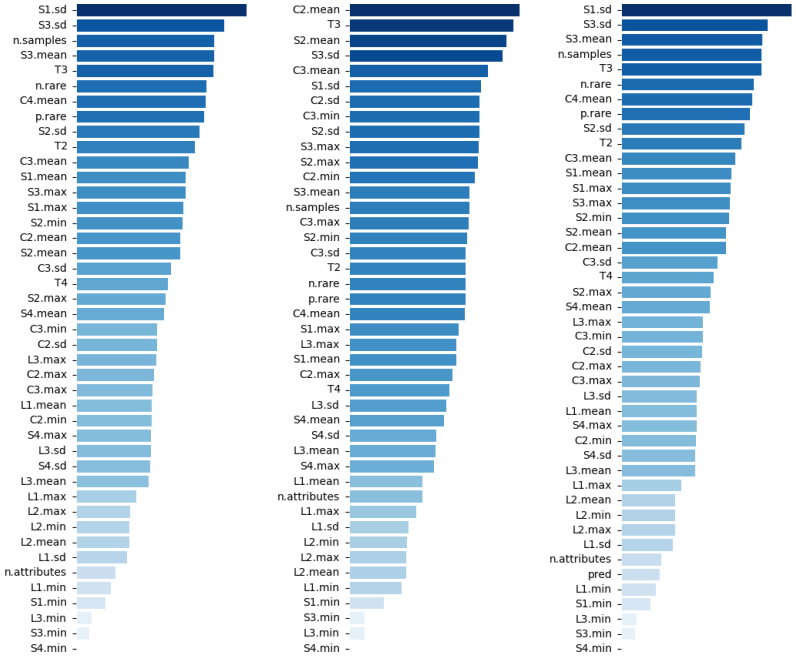
In analyzing both metrics, it became evident that the percentage of rare cases (p.rare) is a crucial factor in determining the recommended resampling strategy, due to its direct correlation with data imbalance. Additionally, the measures C3.min, C4.mean, and S1.mean also play a significant role in this determination. When considering model recommendation, S1.sd emerges as a key factor for the F1-scoreR, while T3 and C2.mean are notable for the SERA metric.

Generally, the minimum values of the measures (.min) have the least influence, indicating that, in the context of an imbalanced regression problem, these minimum values might not be significantly important in recommending learning models and resampling strategies. This suggests that focusing on other aspects or statistical properties, such as maximum values, average values, or variance, as well as simple statistics including the number of examples and the number of rare cases can be more beneficial in making informed decisions.

Upon analyzing the most important meta-features, we have observed that three (C2.mean, C3.min, and C4.mean) measure the relationship between features and the output. This indicates that the relationship between features and the target is crucial in representing imbalanced datasets. The other two meta-features (S1.mean and S1.sd) are related to the output distribution, a fundamental characteristic for addressing the imbalance. The target distribution may be directly linked to selecting the best resampling strategies and learning models. Understanding the target distribution is essential for choosing the pipeline.

It is essential to recognize a limitation related to the meta-features used in this study. Integrating additional high-quality features relevant to the problem could significantly enhance the model's effectiveness. This improvement would positively impact the outcomes of the meta-models and refine the base-level analysis. This recognition highlights the potential for future enhancements that could lead to more robust and accurate results, extending the applicability and effectiveness of the models proposed in this work.

Figure 25 – Feature Importance for recommending the Learning model (a-c) and the Resampling Strategy (d-f), considering F1-scoreR metric as optimization function.

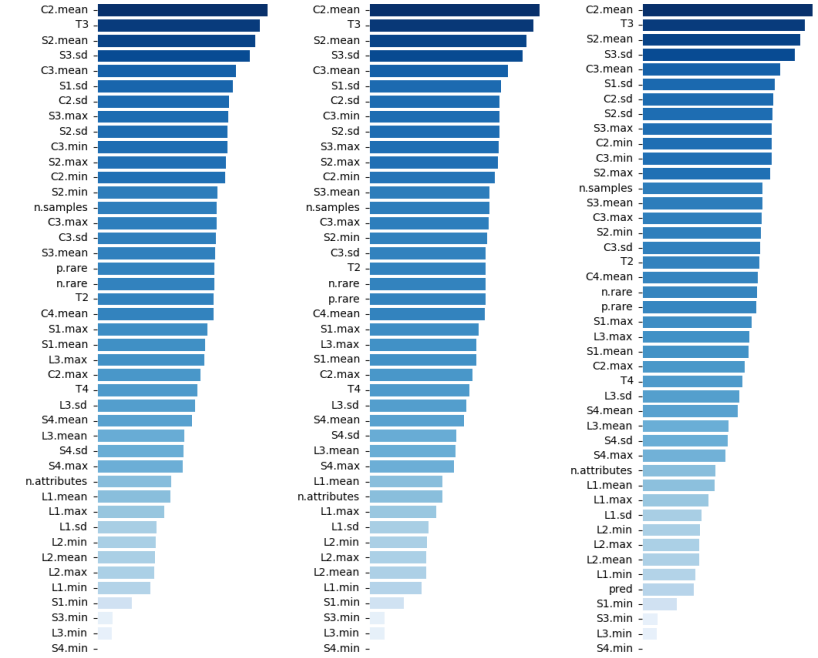


(a) Independent (b) ModelFirst (c) StrategyFirst

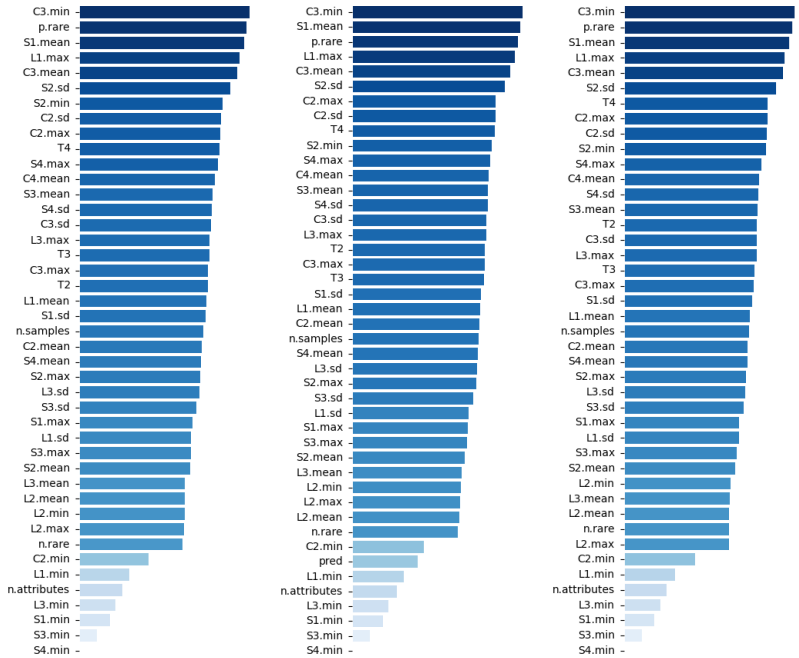
(d) Independent (e) ModelFirst (f) StrategyFirst

Source: Prepared by the author.

Figure 26 – Feature Importance for recommending the Learning model (a-c) and the Resampling Strategy (d-f), considering SERA metric as optimization function.



(a) Independent (b) ModelFirst (c) StrategyFirst



(d) Independent (e) ModelFirst (f) StrategyFirst

Source: Prepared by the author.

4.6 CONCLUSIONS

This work introduced Meta-IR, a meta-learning framework for recommending pipelines for imbalanced regression. Recommendations are made in two ways: Independent and Chained. We conducted an extensive experimental study consisting of 218 imbalanced regression datasets and taking into account 6 resampling techniques and 6 regression models. The meta-level analysis showed Meta-IR outperforms Majority and Random baselines in F1-scoreR and SERA metrics, highlighting the advantage of a meta-learning approach. The base-level assessment revealed that Meta-IR recommendations significantly surpassed Majority and Random. Furthermore, the results showed that Meta-IR consistently outperforms all AutoML frameworks when using the F1-scoreR metric, and when considering the SERA metric, Meta-IR has advantages, particularly in scenarios with a high percentage of rare cases. In terms of time-efficiency, Meta-IR is superior to AutoML frameworks.

The analysis also considered the impact of Meta-IR as a pre-processing step on the performance of various AutoML frameworks. It found that using Meta-IR significantly improved the F1-scoreR for all AutoML frameworks. However, its impact on the SERA metric varied depending on the AutoML method used. While it improved SERA for LightAutoML, H2O and NaiveAutoML, other methods showed deteriorated results. This indicates that the effect of Meta-IR on SERA depends on the AutoML method and needs more investigation.

One limitation of this work is the lack of hyperparameter optimization. We chose not to perform hyperparameter optimization in this study in order to manage computational complexity and to emphasize the primary contribution of the research. We have shown that the recommended models perform consistently well across various problems without requiring fine-tuning, thus underscoring the generality and applicability of our approach. While including hyperparameter optimization in future work could potentially further improve results, its absence does not compromise the validity of the main conclusions of this study. Our future work will also enhance the chained algorithm recommendations by assessing multilabel strategies and investigating another meta-feature set.

5 GENERAL CONCLUSION

The imbalanced regression problem has unique challenges because of the data distribution, where certain value ranges are underrepresented. One way to address these challenges is by using resampling strategies. Our thorough evaluation has revealed that the effectiveness of these strategies is highly contingent on several factors, including the problem, the learning model, and the performance metrics used. Despite the variety of resampling techniques available in the literature, there is no approach to recommend specific resampling strategies for every problem or to establish a relationship between the learning model and the resampling strategy. Therefore, choosing an appropriate resampling method is crucial for addressing this specific task.

To address this issue, we propose Meta-IR, a meta-learning-based model designed explicitly for imbalanced regression problems. Meta-IR aims to recommend optimal learning models and resampling strategies based on the meta-features of the problem. It customizes a pipeline by suggesting the most suitable resampling strategy and learning algorithm for the new problem's meta-features. It offers two approaches: Independent, which trains meta-classifiers separately for the best learning model and resampling strategy, and Chained, which integrates these decisions sequentially.

Overall, Meta-IR proves to be a promising approach for addressing imbalanced regression problems. Our extensive experimental study on 218 imbalanced datasets has demonstrated that Meta-IR significantly improves performance over Majority and Random baselines and consistently outperforms all AutoML frameworks when using the F1-scoreR metric. When considering the SERA metric, Meta-IR has advantages, particularly in scenarios with a high percentage of rare cases. In terms of time efficiency, Meta-IR is superior to AutoML frameworks.

The analysis also considered the impact of Meta-IR as a pre-processing step on the performance of various AutoML frameworks. It found that using Meta-IR significantly improved the F1-scoreR for all AutoML frameworks. However, its impact on the SERA metric varied depending on the AutoML method. While it improved SERA for LightAutoML, H2O, and NaiveAutoML, other methods showed deteriorated results. This indicates that the effect of Meta-IR on SERA depends on the AutoML method and needs more investigation.

In conclusion, by tailoring the resampling strategy and learning model to the specific characteristics of each dataset, Meta-IR achieves notable enhancements in predictive accuracy.

This approach meets the primary goals of our work, offering a solution for the challenges of imbalanced regression.

5.1 LIMITATIONS AND FUTURE WORK

While Meta-IR presents a promising approach for tackling imbalanced regression problems, there are limitations and areas for future improvement.

Firstly, hyperparameter optimization has not been explored in this framework. The current implementation of Meta-IR does not tune hyperparameters for the learning models and resampling strategies it recommends. Future work should focus on integrating comprehensive hyperparameter optimization to enhance the performance and robustness of the recommended pipelines. This could involve techniques such as grid search, random search, or more advanced methods like Bayesian optimization to explore the hyperparameter space and identify optimal settings systematically.

Secondly, the quality and relevance of the meta-features used by Meta-IR could also be improved. While the current meta-features provide a basis for recommending appropriate models and strategies, there is potential to enhance their effectiveness by incorporating additional or more sophisticated features. Future research could explore the inclusion of new meta-features that capture more profound insights into the data characteristics and problem specifics. This might involve developing new methods for feature extraction to create more informative meta-features.

Addressing these limitations by optimizing hyperparameters and refining meta-features will be essential for advancing the Meta-IR framework. These improvements are expected to enhance the performance of Meta-IR further and expand its applicability to a more expansive range of imbalanced regression scenarios.

REFERENCES

AGRAWAL, A.; PETERSEN, M. R. Detecting arsenic contamination using satellite imagery and machine learning. *Toxics*, Multidisciplinary Digital Publishing Institute, v. 9, n. 12, p. 333, 2021.

AGUIAR, G.; KRAWCZYK, B.; CANO, A. A survey on learning from imbalanced data streams: taxonomy, challenges, empirical study, and reproducible experimental framework. *arXiv preprint arXiv:2204.03719*, 2022.

AGUIAR, G. J.; MANTOVANI, R. G.; MASTELINI, S. M.; CARVALHO, A. C. de; CAMPOS, G. F.; JUNIOR, S. B. A meta-learning approach for selecting image segmentation algorithm. *Pattern Recognition Letters*, Elsevier, v. 128, p. 480–487, 2019.

AGUIAR, G. J.; SANTANA, E. J.; CARVALHO, A. C. de; JUNIOR, S. B. Using meta-learning for multi-target regression. *Information Sciences*, Elsevier, v. 584, p. 665–684, 2022.

ALI, H.; SALLEH, M. N. M.; HUSSAIN, K.; AHMAD, A.; ULLAH, A.; MUHAMMAD, A.; NASEEM, R.; KHAN, M. A review on data preprocessing methods for class imbalance problem. *International Journal of Engineering & Technology*, v. 8, p. 390–397, 2019.

AMINIAN, E.; RIBEIRO, R. P.; GAMA, J. Chebyshev approaches for imbalanced data streams regression models. *Data Mining and Knowledge Discovery*, Springer, v. 35, p. 2389–2466, 2021.

AMORIM, L. B. de; CAVALCANTI, G. D.; CRUZ, R. M. Meta-scaler: A meta-learning framework for the selection of scaling techniques. *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, 2024.

AVELINO, J. G.; CAVALCANTI, G. D.; CRUZ, R. M. Resampling strategies for imbalanced regression: a survey and empirical analysis. *Artificial Intelligence Review*, Springer, v. 57, n. 4, p. 82, 2024.

BÄCK, T.; FOGEL, D. B.; MICHALEWICZ, Z. Handbook of evolutionary computation. *Release*, v. 97, n. 1, p. B1, 1997.

BAHRI, M.; SALUTARI, F.; PUTINA, A.; SOZIO, M. Automl: state of the art with a focus on anomaly detection, challenges, and research directions. *International Journal of Data Science and Analytics*, Springer, v. 14, n. 2, p. 113–126, 2022.

BAL, P. R.; KUMAR, S. Cross project software defect prediction using extreme learning machine: An ensemble based study. In: *ICSOFT*. [S.l.: s.n.], 2018. p. 354–361.

BAL, P. R.; KUMAR, S. Wr-elm: Weighted regularization extreme learning machine for imbalance learning in software fault prediction. *IEEE Transactions on Reliability*, IEEE, v. 69, n. 4, p. 1355–1375, 2020.

BARATCHI, M.; WANG, C.; LIMMER, S.; RIJN, J. N. van; HOOS, H.; BÄCK, T.; OLHOFER, M. Automated machine learning: past, present and future. *Artificial Intelligence Review*, Springer, v. 57, n. 5, p. 1–88, 2024.

BARELLA, V. H.; GARCIA, L. P.; CARVALHO, A. C. de. Simulating complexity measures on imbalanced datasets. In: SPRINGER. *Brazilian Conference on Intelligent Systems*. [S.l.], 2020. p. 498–512.

BARELLA, V. H.; GARCIA, L. P.; SOUTO, M. C. de; LORENA, A. C.; CARVALHO, A. C. de. Assessing the data complexity of imbalanced datasets. *Information Sciences*, Elsevier, v. 553, p. 83–109, 2021.

BARELLA, V. H.; GARCIA, L. P.; SOUTO, M. P. de; LORENA, A. C.; CARVALHO, A. de. Data complexity measures for imbalanced classification tasks. In: IEEE. *2018 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2018. p. 1–8.

BATISTA, G. E.; PRATI, R. C.; MONARD, M. C. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, ACM New York, NY, USA, v. 6, n. 1, p. 20–29, 2004.

BENGIO, Y. Gradient-based optimization of hyperparameters. *Neural computation*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 12, n. 8, p. 1889–1900, 2000.

BERGSTRA, J.; BARDENET, R.; KÉGL, B.; BENGIO, Y. Implementations of algorithms for hyper-parameter optimization. In: *NIPS Workshop on Bayesian optimization*. [S.l.: s.n.], 2011. v. 29.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. *Journal of machine learning research*, v. 13, n. 2, 2012.

BRANCO, P.; RIBEIRO, R. P.; TORGO, L. Ubl: an r package for utility-based learning. *arXiv preprint arXiv:1604.08079*, 2016.

BRANCO, P.; TORGO, L.; RIBEIRO, R. P. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 49, n. 2, p. 1–50, 2016.

BRANCO, P.; TORGO, L.; RIBEIRO, R. P. Smogn: a pre-processing approach for imbalanced regression. In: *First International Workshop on Learning with Imbalanced Domains: Theory and Applications*. [S.l.: s.n.], 2017. v. 74, p. 36–50.

BRANCO, P.; TORGO, L.; RIBEIRO, R. P. Rebagg: Resampled bagging for imbalanced regression. In: *Second International Workshop on Learning with Imbalanced Domains: Theory and Applications*. [S.l.: s.n.], 2018. p. 67–81.

BRANCO, P.; TORGO, L.; RIBEIRO, R. P. Pre-processing approaches for imbalanced distributions in regression. *Neurocomputing*, Elsevier, v. 343, p. 76–99, 2019.

BRANCO, P. A. de O. Utility-based predictive analytics. 2018.

BRAZDIL, P.; CARRIER, C. G.; SOARES, C.; VILALTA, R. *Metalearning: Applications to data mining*. Heidelberg: Springer Science & Business Media, 2008.

BRAZDIL, P.; RIJN, J. N. van; SOARES, C.; VANSCHOREN, J. Dataset characteristics (metafeatures). In: *Metalearning*. [S.l.]: Springer, 2022. p. 53–75.

BRAZDIL, P.; RIJN, J. N. van; SOARES, C.; VANSCHOREN, J. *Metalearning: Applications to automated machine learning and data mining*. Cham: Springer Nature, 2022.

BRAZDIL, P. B.; SOARES, C. A comparison of ranking methods for classification algorithm selection. In: SPRINGER. *Machine Learning: ECML 2000: 11th European Conference on Machine Learning Barcelona, Catalonia, Spain, May 31–June 2, 2000 Proceedings 11*. [S.l.], 2000. p. 63–75.

BRAZDIL, P. B.; SOARES, C.; COSTA, J. P. D. Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results. *Machine Learning*, Springer, v. 50, p. 251–277, 2003.

BREIMAN, L. Bagging predictors. *Machine learning*, Springer, v. 24, n. 2, p. 123–140, 1996.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

BREIMAN, L. *Classification and regression trees*. [S.l.]: Routledge, 2017.

CAMACHO, L.; DOUZAS, G.; BACAO, F. Geometric smote for regression. *Expert Systems with Applications*, Elsevier, p. 116387, 2022.

CAVALCANTI, G. D.; REN, T. I.; VALE, B. A. Data complexity measures and nearest neighbor classifiers: a practical analysis for meta-learning. In: IEEE. *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*. [S.l.], 2012. v. 1, p. 1065–1069.

CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, v. 16, p. 321–357, 2002.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. [S.l.: s.n.], 2016. p. 785–794.

CRUZ, R. M.; SOUZA, M. A.; SABOURIN, R.; CAVALCANTI, G. D. Dynamic ensemble selection and data preprocessing for multi-class imbalance learning. *International Journal of Pattern Recognition and Artificial Intelligence*, World Scientific, v. 33, n. 11, p. 1940009, 2019.

de Amorim, L. B.; CAVALCANTI, G. D.; CRUZ, R. M. The choice of scaling technique matters for classification performance. *Applied Soft Computing*, v. 133, p. 109924, 2023. ISSN 1568-4946.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, v. 7, n. Jan, p. 1–30, 2006.

DÍEZ-PASTOR, J. F.; RODRÍGUEZ, J. J.; GARCÍA-OSORIO, C. I.; KUNCHEVA, L. I. Diversity techniques improve the performance of the best imbalance learning ensembles. *Information Sciences*, Elsevier, v. 325, p. 98–117, 2015.

DOUGHERTY, R. L.; EDELMAN, A. S.; HYMAN, J. M. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic hermite interpolation. *Mathematics of Computation*, v. 52, n. 186, p. 471–494, 1989.

ESCALANTE, H. J.; MONTES, M.; SUCAR, L. E. Particle swarm model selection. *Journal of Machine Learning Research*, v. 10, n. 2, 2009.

FEURER, M.; KLEIN, A.; EGGENSPERGER, K.; SPRINGENBERG, J.; BLUM, M.; HUTTER, F. Efficient and robust automated machine learning. *Advances in neural information processing systems*, v. 28, 2015.

FRITSCH, F. N.; CARLSON, R. E. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, SIAM, v. 17, n. 2, p. 238–246, 1980.

GADO, J. E.; BECKHAM, G. T.; PAYNE, C. M. Improving enzyme optimum temperature prediction with resampling strategies and ensemble learning. *Journal of Chemical Information and Modeling*, ACS Publications, v. 60, n. 8, p. 4098–4107, 2020.

GALAR, M.; FERNANDEZ, A.; BARRENECHEA, E.; BUSTINCE, H.; HERRERA, F. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, IEEE, v. 42, n. 4, p. 463–484, 2011.

GANGANWAR, V. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, v. 2, n. 4, p. 42–47, 2012.

GARCIA, L. P.; CARVALHO, A. C. de; LORENA, A. C. Noise detection in the meta-learning level. *Neurocomputing*, Elsevier, v. 176, p. 14–25, 2016.

GARCIA, L. P.; LORENA, A. C.; SOUTO, M. C. de; HO, T. K. Classifier recommendation using data complexity measures. In: IEEE. *2018 24th International Conference on Pattern Recognition (ICPR)*. [S.l.], 2018. p. 874–879.

GARCÍA, S.; ZHANG, Z.-L.; ALTALHI, A.; ALSHOMRANI, S.; HERRERA, F. Dynamic ensemble selection for multi-class imbalanced datasets. *Information Sciences*, Elsevier, v. 445, p. 22–37, 2018.

GARCÍA, V.; SÁNCHEZ, J. S.; MARQUÉS, A.; FLORENCIA, R.; RIVERA, G. Understanding the apparent superiority of over-sampling through an analysis of local information for class-imbalanced data. *Expert Systems with Applications*, Elsevier, v. 158, p. 113026, 2020.

GARNETT, R. *Bayesian Optimization*. Cambridge: Cambridge University Press, 2023.

GAROUANI, M.; AHMAD, A.; BOUNEFFA, M.; HAMLICH, M. Autoencoder-knn meta-model based data characterization approach for an automated selection of ai algorithms. *Journal of Big Data*, Springer, v. 10, n. 1, p. 14, 2023.

GEURTS, P.; ERNST, D.; WEHENKEL, L. Extremely randomized trees. *Machine learning*, Springer, v. 63, p. 3–42, 2006.

GHIMIRE, S.; GUÉGUEN, P.; GIFFARD-ROISIN, S.; SCHORLEMMER, D. Testing machine learning models for seismic damage prediction at a regional scale using building-damage dataset compiled after the 2015 gorkha nepal earthquake. *Earthquake Spectra*, SAGE Publications Sage UK: London, England, p. 87552930221106495, 2022.

GIRAUD-CARRIER, C.; VILALTA, R.; BRAZDIL, P. Introduction to the special issue on meta-learning. *Machine learning*, Springer, v. 54, n. 3, p. 187–193, 2004.

HAIXIANG, G.; YIJING, L.; SHANG, J.; MINGYUN, G.; YUANYUE, H.; BING, G. Learning from class-imbalanced data: Review of methods and applications. *Expert systems with applications*, Elsevier, v. 73, p. 220–239, 2017.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. H.; FRIEDMAN, J. H. *The elements of statistical learning: data mining, inference, and prediction*. [S.l.]: Springer, 2009. v. 2.

HE, A.; MUNASINGHE, T. Chronic respiratory disease: Risk modeling potential and limitations. In: IEEE. *2021 IEEE International Conference on Big Data (Big Data)*. [S.l.], 2021. p. 1045–1053.

HE, H.; GARCIA, E. A. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, Ieee, v. 21, n. 9, p. 1263–1284, 2009.

HE, X.; ZHAO, K.; CHU, X. Automl: A survey of the state-of-the-art. *Knowledge-based systems*, Elsevier, v. 212, p. 106622, 2021.

HUBERT, M.; VANDERVIEREN, E. An adjusted boxplot for skewed distributions. *Computational statistics & data analysis*, Elsevier, v. 52, n. 12, p. 5186–5201, 2008.

HUTTER, F.; HOOS, H. H.; LEYTON-BROWN, K. Sequential model-based optimization for general algorithm configuration. In: SPRINGER. *Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers 5*. [S.l.], 2011. p. 507–523.

HUTTER, F.; KOTTHOFF, L.; VANSCHOREN, J. *Automated Machine Learning: Methods, Systems, Challenges*. [S.l.]: Springer Nature, 2019.

JOHNSON, J. M.; KHOSHGOFTAAR, T. M. Survey on deep learning with class imbalance. *Journal of Big Data*, Springer, v. 6, n. 1, p. 1–54, 2019.

KHAN, I.; ZHANG, X.; REHMAN, M.; ALI, R. A literature survey and empirical study of meta-learning for classifier selection. *IEEE Access*, IEEE, v. 8, p. 10262–10281, 2020.

KO, A. H.; SABOURIN, R.; JR, A. S. B. From dynamic classifier selection to dynamic ensemble selection. *Pattern recognition*, Elsevier, v. 41, n. 5, p. 1718–1731, 2008.

KOVÁCS, G. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, Elsevier, v. 83, p. 105662, 2019.

KRAWCZYK, B. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, Springer, v. 5, n. 4, p. 221–232, 2016.

KUBAT, M.; MATWIN, S. et al. Addressing the curse of imbalanced training sets: one-sided selection. In: CITESEER. *Icml*. [S.l.], 1997. v. 97, n. 1, p. 179.

LEDELL, E.; POIRIER, S. H2O AutoML: Scalable automatic machine learning. *7th ICML Workshop on Automated Machine Learning (AutoML)*, July 2020. Disponível em: <https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf>.

LEE, S. S. Regularization in skewed binary classification. *Computational Statistics*, Springer, v. 14, n. 2, p. 277–292, 1999.

LEE, S. S. Noisy replication in skewed binary classification. *Computational statistics & data analysis*, Elsevier, v. 34, n. 2, p. 165–191, 2000.

LEYVA, E.; GONZÁLEZ, A.; PEREZ, R. A set of complexity measures designed for applying meta-learning to instance selection. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 27, n. 2, p. 354–367, 2014.

LÓPEZ, V.; FERNÁNDEZ, A.; GARCÍA, S.; PALADE, V.; HERRERA, F. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information sciences*, Elsevier, v. 250, p. 113–141, 2013.

LORENA, A. C.; GARCIA, L. P.; LEHMANN, J.; SOUTO, M. C.; HO, T. K. How complex is your classification problem? a survey on measuring classification complexity. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 52, n. 5, p. 1–34, 2019.

LORENA, A. C.; MACIEL, A. I.; MIRANDA, P. B. de; COSTA, I. G.; PRUDÊNCIO, R. B. Data complexity meta-features for regression problems. *Machine Learning*, Springer, v. 107, n. 1, p. 209–246, 2018.

MOHR, F.; WEVER, M. Naive automated machine learning. *Machine Learning*, Springer, v. 112, n. 4, p. 1131–1170, 2023.

MONIZ, N.; BRANCO, P.; TORGO, L. Resampling strategies for imbalanced time series forecasting. *International Journal of Data Science and Analytics*, Springer, v. 3, n. 3, p. 161–181, 2017.

MONIZ, N.; CERQUEIRA, V. Automated imbalanced classification via meta-learning. *Expert Systems with Applications*, Elsevier, v. 178, p. 115011, 2021.

MONIZ, N.; MONTEIRO, H. No free lunch in imbalanced learning. *Knowledge-Based Systems*, Elsevier, v. 227, p. 107222, 2021.

MONIZ, N.; RIBEIRO, R.; CERQUEIRA, V.; CHAWLA, N. Smoteboost for regression: Improving the prediction of extreme values. In: IEEE. *2018 IEEE 5th international conference on data science and advanced analytics (DSAA)*. [S.l.], 2018. p. 150–159.

MONIZ, N.; TORGO, L.; RODRIGUES, F. Resampling approaches to improve news importance prediction. In: SPRINGER. *International Symposium on Intelligent Data Analysis*. [S.l.], 2014. p. 215–226.

MONIZ, N. M.; BRANCO, P. O.; TORGO, L. Evaluation of ensemble methods in imbalanced regression tasks. In: *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*. [S.l.: s.n.], 2017. v. 74, p. 129–140.

MORÁN-FERNÁNDEZ, L.; BOLÓN-CANEDO, V.; ALONSO-BETANZOS, A. Can classification performance be predicted by complexity measures? a study using microarray data. *Knowledge and Information Systems*, Springer, v. 51, n. 3, p. 1067–1090, 2017.

OLSON, R. S.; BARTLEY, N.; URBANOWICZ, R. J.; MOORE, J. H. Evaluation of a tree-based pipeline optimization tool for automating data science. In: *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. New York, NY, USA: ACM, 2016. (GECCO '16), p. 485–492. ISBN 978-1-4503-4206-3.

PFAHRINGER, B.; BENSUSAN, H.; GIRAUD-CARRIER, C. G. Meta-learning by landmarking various learning algorithms. In: CITESEER. *ICML*. [S.l.], 2000. p. 743–750.

RATHORE, S. S.; KUMAR, S. Linear and non-linear heterogeneous ensemble methods to predict the number of faults in software systems. *Knowledge-Based Systems*, Elsevier, v. 119, p. 232–256, 2017.

RATHORE, S. S.; KUMAR, S. Towards an ensemble based system for predicting the number of software faults. *Expert Systems with Applications*, Elsevier, v. 82, p. 357–382, 2017.

REIF, M.; SHAFAIT, F.; GOLDSTEIN, M.; BREUEL, T.; DENGEL, A. Automatic classifier selection for non-experts. *Pattern Analysis and Applications*, Springer, v. 17, p. 83–96, 2014.

RIBEIRO, R. Utility-based regression. *Ph. D. dissertation*, Dep. Computer Science, Faculty of Sciences-University of Porto, 2011.

RIBEIRO, R. P.; MONIZ, N. Imbalanced regression and extreme value prediction. *Machine Learning*, Springer, v. 109, n. 9, p. 1803–1835, 2020.

RICE, J. R. The algorithm selection problem. In: *Advances in computers*. [S.l.]: Elsevier, 1976. v. 15, p. 65–118.

RIO, S. D.; BENÍTEZ, J. M.; HERRERA, F. Analysis of data preprocessing increasing the oversampling ratio for extremely imbalanced big data classification. In: IEEE. *2015 IEEE Trustcom/BigDataSE/ISPA*. [S.l.], 2015. v. 2, p. 180–185.

RIVOLLI, A.; GARCIA, L. P.; SOARES, C.; VANSCHOREN, J.; CARVALHO, A. C. de. Meta-features for meta-learning. *Knowledge-Based Systems*, Elsevier, v. 240, p. 108101, 2022.

ROSSI, A. L. D.; FERREIRA, A. C. P. de L.; SOARES, C.; SOUZA, B. F. D. et al. Metastream: A meta-learning based method for periodic algorithm selection in time-changing data. *Neurocomputing*, Elsevier, v. 127, p. 52–64, 2014.

ROSSI, A. L. D.; SOARES, C.; SOUZA, B. F. de; FERREIRA, A. C. P. de L. et al. Micro-metastream: algorithm selection for time-changing data. *Information Sciences*, Elsevier, v. 565, p. 262–277, 2021.

ROY, A.; CRUZ, R. M.; SABOURIN, R.; CAVALCANTI, G. D. A study on combining dynamic selection and data preprocessing for imbalance learning. *Neurocomputing*, Elsevier, v. 286, p. 179–192, 2018.

RUDD, D. H.; HUO, H.; XU, G. Predicting financial literacy via semi-supervised learning. In: SPRINGER. *Australasian Joint Conference on Artificial Intelligence*. [S.l.], 2022. p. 304–319.

SÁEZ, J. A.; KRAWCZYK, B.; WOŹNIAK, M. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition*, Elsevier, v. 57, p. 164–178, 2016.

SALCEDO-SANZ, S.; PÉREZ-ARACIL, J.; ASCENSO, G.; SER, J. D.; CASILLAS-PÉREZ, D.; KADOW, C.; FISTER, D.; BARRIOPEDRO, D.; GARCÍA-HERRERA, R.; RESTELLI, M. et al. Analysis, characterization, prediction and attribution of extreme atmospheric events with machine learning: a review. *arXiv preprint arXiv:2207.07580*, 2022.

SIMON, D. *Evolutionary optimization algorithms*. Hoboken: Wiley Online Library, 2013.

SNOEK, J.; LAROCHELLE, H.; ADAMS, R. P. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, v. 25, 2012.

SONG, X. Y.; DAO, N.; BRANCO, P. Distsmogn: Distributed smogn for imbalanced regression problems. In: PMLR. *Fourth International Workshop on Learning with Imbalanced Domains: Theory and Applications*. [S.l.], 2022. p. 38–52.

SOUSA, A. F.; PRUDÊNCIO, R. B.; LUDERMIR, T. B.; SOARES, C. Active learning and data manipulation techniques for generating training examples in meta-learning. *Neurocomputing*, Elsevier, v. 194, p. 45–55, 2016.

STEININGER, M.; KOBS, K.; DAVIDSON, P.; KRAUSE, A.; HOTHO, A. Density-based weighting for imbalanced regression. *Machine Learning*, Springer, v. 110, p. 2187–2211, 2021.

THORNTON, C.; HUTTER, F.; HOOS, H. H.; LEYTON-BROWN, K. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2013. p. 847–855.

TORGO, L.; RIBEIRO, R. Predicting outliers. In: SPRINGER. *European Conference on Principles of Data Mining and Knowledge Discovery*. [S.l.], 2003. p. 447–458.

TORGO, L.; RIBEIRO, R. Utility-based regression. In: SPRINGER. *European Conference on Principles of Data Mining and Knowledge Discovery*. [S.l.], 2007. p. 597–604.

TORGO, L.; RIBEIRO, R. Precision and recall for regression. In: SPRINGER. *International Conference on Discovery Science*. [S.l.], 2009. p. 332–346.

TORGO, L.; RIBEIRO, R. P.; PFAHRINGER, B.; BRANCO, P. Smote for regression. In: SPRINGER. *Portuguese conference on artificial intelligence*. [S.l.], 2013. p. 378–389.

TUKEY, J. *Exploratory Data Analysis, limited prelim. ed*. [S.l.]: Addison-Wesley, Reading, Mass, 1970.

VAKHRUSHEV, A.; RYZHKOV, A.; SAVCHENKO, M.; SIMAKOV, D.; DAMDINOV, R.; TUZHILIN, A. Lightautoml: Automl solution for a large financial services ecosystem. *arXiv preprint arXiv:2109.01528*, 2021.

VANSCHOREN, J. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

VANSCHOREN, J. Meta-learning. *Automated machine learning: methods, systems, challenges*, Springer International Publishing, p. 35–61, 2019.

VANSCHOREN, J.; RIJN, J. N. V.; BISCHL, B.; TORGO, L. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, ACM New York, NY, USA, v. 15, n. 2, p. 49–60, 2014.

VARMA, S.; SIMON, R. Bias and variance of cross-validation based estimators of gene selection error. *Bioinformatics*, Oxford University Press, v. 22, n. 14, p. 185–199, 2006.

WANG, C.; WU, Q.; WEIMER, M.; ZHU, E. Flaml: A fast and lightweight automl library. *Proceedings of Machine Learning and Systems*, v. 3, p. 434–447, 2021.

WOJCIECHOWSKI, S.; WILK, S. Difficulty factors and preprocessing in imbalanced data sets: an experimental study on artificial data. *Foundations of Computing and Decision Sciences*, v. 42, n. 2, p. 149–176, 2017.

WU, Q.; WANG, C.; HUANG, S. Frugal optimization for cost-related hyperparameters. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2021. v. 35, n. 12, p. 10347–10354.

WU, W.; KUNZ, N.; BRANCO, P. Imbalancedlearningregression-a python package to tackle the imbalanced regression problem. In: SPRINGER. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. [S.l.], 2022. p. 645–648.

YANG, Y.; ZHA, K.; CHEN, Y.; WANG, H.; KATABI, D. Delving into deep imbalanced regression. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2021. p. 11842–11851.

YAO, Q.; WANG, M.; CHEN, Y.; DAI, W.; LI, Y.-F.; TU, W.-W.; YANG, Q.; YU, Y. Taking human out of learning applications: A survey on automated machine learning. *arXiv preprint arXiv:1810.13306*, 2018.

ZAGATTI, F. R.; SILVA, L. C.; SILVA, L. N. D. S.; SETTE, B. S.; CASELI, H. de M.; LUCRÉDIO, D.; SILVA, D. F. Metaprep: Data preparation pipelines recommendation via meta-learning. In: IEEE. *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*. [S.l.], 2021. p. 1197–1202.

ZÖLLER, M.-A.; HUBER, M. F. Benchmark and survey of automated machine learning frameworks. *Journal of artificial intelligence research*, v. 70, p. 409–472, 2021.

ZYBLEWSKI, P.; SABOURIN, R.; WOŹNIAK, M. Data preprocessing and dynamic ensemble selection for imbalanced data stream classification. In: SPRINGER. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. [S.l.], 2019. p. 367–379.

APPENDIX A - DATASET DESCRIPTION

Table 20 – Datasets ordered by the percentage of rare cases. (n.examples: Number of examples; n.attributes: Number of attributes; n.rare: Number of rare cases; p.rare: $100 \times n.raro/n.exemplos$

Dataset	n.samples	n.attributes	n.rare	p.rare
QSAR-TID-20137	101	1024	40	39,6
QSAR-TID-11569	120	1024	44	36,7
QSAR-TID-11637	96	1024	34	35,4
QSAR-TID-10426	27	1024	9	33,3
QSAR-TID-12847	215	1024	68	31,6
QSAR-TID-101612	91	1024	28	30,8
QSAR-TID-101332	106	1024	32	30,2
QSAR-TID-12263	47	1024	13	27,7
QSAR-TID-101602	79	1024	21	26,6
QSAR-TID-12959	72	1024	19	26,4
QSAR-TID-10778	82	1024	21	25,6
rabe-265	51	6	13	25,5
QSAR-TID-30050	80	1024	20	25,0
QSAR-TID-30005	81	1024	20	24,7
QSAR-TID-103453	73	1024	18	24,7
QSAR-TID-101333	168	1024	41	24,4
QSAR-TID-101226	79	1024	19	24,1
QSAR-TID-11843	84	1024	20	23,8
QSAR-TID-101553	80	1024	19	23,8
QSAR-TID-10939	81	1024	19	23,5
wine-quality	6497	11	1523	23,4
analcat-apnea3	450	11	103	22,9

Dataset	n.samples	n.attributes	n.rare	p.rare
QSAR-TID-101231	79	1024	18	22,8
analcat-ncaa	120	19	27	22,5
QSAR-TID-19607	143	1024	32	22,4
QSAR-TID-101317	99	1024	22	22,2
QSAR-TID-10478	86	1024	19	22,1
analcat-apnea1	475	11	104	21,9
QSAR-TID-100833	83	1024	18	21,7
QSAR-TID-30024	84	1024	18	21,4
QSAR-TID-100975	75	1024	16	21,3
analcat-apnea2	475	11	101	21,3
QSAR-TID-101448	80	1024	17	21,3
QSAR-TID-101340	86	1024	18	20,9
analcat-supreme	4052	7	835	20,6
meta	528	65	108	20,5
QSAR-TID-100918	88	1024	18	20,5
QSAR-TID-101048	74	1024	15	20,3
QSAR-TID-101191	74	1024	15	20,3
QSAR-TID-100925	79	1024	16	20,3
QSAR-TID-10621	99	1024	20	20,2
QSAR-TID-10618	134	1024	27	20,1
kc1-numeric	145	94	29	20,0
QSAR-TID-101399	75	1024	15	20,0
QSAR-TID-101433	81	1024	16	19,8
QSAR-TID-30009	102	1024	20	19,6
QSAR-TID-100063	149	1024	29	19,5

Dataset	n.samples	n.attributes	n.rare	p.rare
QSAR-TID-30027	83	1024	16	19,3
QSAR-TID-20162	156	1024	30	19,2
QSAR-TID-101278	79	1024	15	19,0
QSAR-TID-101584	74	1024	14	18,9
QSAR-TID-11281	43	1024	8	18,6
QSAR-TID-100951	81	1024	15	18,5
QSAR-TID-11784	65	1024	12	18,5
QSAR-TID-11873	115	1024	21	18,3
analcat-chlamydia	100	17	18	18,0
QSAR-TID-30034	162	1024	29	17,9
socmob	1156	39	206	17,8
QSAR-TID-101130	79	1024	14	17,7
QSAR-TID-101239	80	1024	14	17,5
QSAR-TID-12780	121	1024	21	17,4
QSAR-TID-101055	81	1024	14	17,3
QSAR-TID-30022	81	1024	14	17,3
QSAR-TID-30041	81	1024	14	17,3
QSAR-TID-30015	116	1024	20	17,2
analcat-wildcat	163	5	28	17,2
QSAR-TID-11056	41	1024	7	17,1
QSAR-TID-30010	82	1024	14	17,1
QSAR-TID-10844	166	1024	28	16,9
QSAR-TID-30020	89	1024	15	16,9
a6	198	11	33	16,7
cocomo-numeric	60	56	10	16,7

Dataset	n.samples	n.attributes	n.rare	p.rare
QSAR-TID-30004	84	1024	14	16,7
kdd-coil-6	316	18	52	16,5
QSAR-TID-100906	79	1024	13	16,5
QSAR-TID-101324	79	1024	13	16,5
QSAR-TID-101309	73	1024	12	16,4
QSAR-TID-12718	134	1024	22	16,4
QSAR-TID-100416	122	1024	20	16,4
machineCPU	209	6	34	16,3
abalone	4177	8	679	16,3
a3	198	11	32	16,2
nasa-numeric	93	90	15	16,1
QSAR-TID-102669	180	1024	29	16,1
QSAR-TID-30042	81	1024	13	16,0
QSAR-TID-101204	75	1024	12	16,0
QSAR-TID-10983	119	1024	19	16,0
QSAR-TID-30032	107	1024	17	15,9
cpu-act	209	36	33	15,8
QSAR-TID-30000	83	1024	13	15,7
a4	198	11	31	15,7
QSAR-TID-30016	97	1024	15	15,5
QSAR-TID-17121	182	1024	28	15,4
QSAR-TID-11692	98	1024	15	15,3
forestFires	517	12	79	15,3
kdd-coil-3	316	18	48	15,2
QSAR-TID-101503	79	1024	12	15,2

Dataset	n.samples	n.attributes	n.rare	p.rare
QSAR-TID-11361	79	1024	12	15,2
QSAR-TID-101610	145	1024	22	15,2
analcat-olympic2000	66	11	10	15,2
QSAR-TID-12407	66	1024	10	15,2
sleuth-case1202	93	6	14	15,1
QSAR-TID-101312	80	1024	12	15,0
QSAR-TID-10782	116	1024	17	14,7
QSAR-TID-10143	62	1024	9	14,5
QSAR-TID-11639	201	1024	29	14,4
a1	198	11	28	14,1
QSAR-TID-11107	85	1024	12	14,1
QSAR-TID-10466	78	1024	11	14,1
fri-c2-100-5	100	5	14	14,0
fri-c4-250-10	250	10	35	14,0
kdd-coil-4	316	18	44	13,9
QSAR-TID-101506	79	1024	11	13,9
a7	198	11	27	13,6
QSAR-TID-100931	110	1024	15	13,6
kdd-coil-2	316	18	43	13,6
fri-c3-250-5	250	5	34	13,6
QSAR-TID-100835	125	1024	17	13,6
QSAR-TID-20025	89	1024	12	13,5
analcat-election2000	67	14	9	13,4
humans-numeric	75	14	10	13,3
QSAR-TID-103800	91	1024	12	13,2

Dataset	n.samples	n.attributes	n.rare	p.rare
kidney	76	10	10	13,2
veteran	137	13	18	13,1
kdd-coil-7	316	18	41	13,0
analcat-seropositive	132	5	17	12,9
boston	506	13	65	12,8
QSAR-TID-11113	94	1024	12	12,8
pdgfr	79	320	10	12,7
fri-c3-500-5	500	5	63	12,6
QSAR-TID-10187	64	1024	8	12,5
QSAR-TID-11094	192	1024	24	12,5
QSAR-TID-100424	97	1024	12	12,4
QSAR-TID-102667	106	1024	13	12,3
kdd-coil-5	316	18	38	12,0
sensory	576	11	69	12,0
auto93	93	57	11	11,8
QSAR-TID-11299	68	1024	8	11,8
QSAR-TID-10929	154	1024	18	11,7
delta-elevators	9517	6	1109	11,7
QSAR-TID-10012	224	1024	26	11,6
a2	198	11	22	11,1
QSAR-TID-101504	99	1024	11	11,1
fri-c3-100-50	100	50	11	11,0
QSAR-TID-20128	100	1024	11	11,0
QSAR-TID-17086	74	1024	8	10,8
kdd-coil-1	316	18	34	10,8

Dataset	n.samples	n.attributes	n.rare	p.rare
triazines	186	60	20	10,8
airfoild	1503	5	161	10,7
a5	198	11	21	10,6
QSAR-TID-101301	151	1024	16	10,6
treasury	1049	15	109	10,4
pharynx	195	10	20	10,3
mortgage	1049	15	106	10,1
debutanizer	2394	7	240	10,0
QSAR-TID-100790	170	1024	17	10,0
QSAR-TID-103071	150	1024	15	10,0
fri-c4-1000-25	1000	25	99	9,9
QSAR-TID-30046	81	1024	8	9,9
QSAR-TID-10407	176	1024	17	9,7
QSAR-TID-11209	95	1024	9	9,5
QSAR-TID-101033	75	1024	7	9,3
fuel-consumption-country	1764	37	164	9,3
QSAR-TID-12131	111	1024	10	9,0
fri-c2-500-25	500	25	45	9,0
heat	7400	11	664	9,0
QSAR-TID-100127	101	1024	9	8,9
QSAR-TID-10967	101	1024	9	8,9
fri-c4-1000-10	1000	10	89	8,9
california	20640	8	1821	8,8
available-power	1802	15	157	8,7
compactiv	8192	21	713	8,7

Dataset	n.samples	n.attributes	n.rare	p.rare
cpu	8192	21	713	8,7
fishcatch	158	7	13	8,2
fri-c3-1000-10	1000	10	82	8,2
fri-c3-1000-5	1000	5	82	8,2
autoPrice	159	15	13	8,2
sleuth-case2002	147	6	12	8,2
QSAR-TID-103062	74	1024	6	8,1
cps-85-wages	534	23	43	8,1
chscase-census2	400	7	32	8,0
QSAR-TID-11567	76	1024	6	7,9
chatfield-4	235	12	18	7,7
places	329	8	25	7,6
QSAR-TID-12536	202	1024	15	7,4
plasma-retinol	315	18	23	7,3
QSAR-TID-100155	138	1024	10	7,2
maximal	1802	32	129	7,2
QSAR-TID-12635	85	1024	6	7,1
QSAR-TID-10113	131	1024	9	6,9
QSAR-TID-12887	117	1024	8	6,8
QSAR-TID-102988	88	1024	6	6,8
chscase-census3	400	7	27	6,8
QSAR-TID-10856	121	1024	8	6,6
cloud	108	9	7	6,5
Moneyball	1232	53	79	6,4
fri-c0-250-10	250	10	16	6,4

Dataset	n.samples	n.attributes	n.rare	p.rare
bank8FM	8192	8	524	6,4
QSAR-TID-10728	172	1024	11	6,4
QSAR-TID-19689	157	1024	10	6,4
no2	500	7	31	6,2
QSAR-TID-12173	114	1024	7	6,1
QSAR-TID-12013	200	1024	12	6,0
QSAR-TID-12587	157	1024	9	5,7
lungcancer-shedden	442	24	25	5,7
cholesterol	303	13	17	5,6
space-ga	3107	6	173	5,6
liver-disorders	345	5	19	5,5
chscase-census6	400	6	22	5,5
concreteStrength	1030	8	55	5,3
acceleration	1732	14	89	5,1
pwLinear	200	10	10	5,0
wind	6574	14	283	4,3
pm10	500	7	21	4,2
chscase-census5	400	7	13	3,3
fri-c0-1000-50	1000	50	27	2,7
fri-c3-1000-50	1000	50	25	2,5
fri-c4-1000-50	1000	50	20	2,0

Source: Prepared by the author.

APPENDIX B - META-FEATURES DESCRIPTION

Maximum feature correlation to the output (C_1):

The absolute value of Spearman's correlation (ρ) is calculated between each attribute (x) and the outputs (y). C_1 is the maximum value obtained among all attributes, as represented in Equation 1. A higher value of this measure indicates a simpler problem.

$$C_1 = \max_{j=1,\ldots,d} |\rho(x^j, y)| \tag{1}$$

Average feature correlation to the output (C_2):

In this measure, the average of the correlations of all features with the output is calculated, as shown in Equation 2. Similarly to C_1, higher values indicate simpler problems.

$$C_2 = \sum_{j=1}^{d} \frac{|\rho(x^j, y)|}{d} \tag{2}$$

Individual feature efficiency (C_3):

For each attribute, the number of examples that need to be removed from the dataset to achieve a high correlation with the output is calculated. Then, the number of removed examples (n^j) is divided by the total number of examples (n) for each attribute, and the minimum of these values is returned, as shown in Equation 3.

$$C_3 = \min_{j=1}^{d} \frac{n^j}{n} \tag{3}$$

Collective feature efficiency (C_4):

Initially, the attribute with the highest correlation with the output is identified. Then, all examples with $|\epsilon_i| < 0.1$ are excluded. Subsequently, the most correlated attribute with the remaining points is found, and the process is repeated until all attributes have been analyzed. Finally, the proportion of examples where $|\epsilon_i| > 0.1$ is returned, as described in Equation 4.

$$C_4 = \frac{\#\{X_i||\epsilon_i| > 0.1\}T_l}{n} \tag{4}$$

Where n is the number of examples remaining in the dataset. T_l is the dataset from which this number is calculated, and l is the number of iterations performed by the algorithm.

Mean absolute error (L_1):

In L_1 (Equation 5), the average of the absolute values of the residuals from a multiple linear regressor is calculated. Lower values indicate simpler problems.

$$L_1 = \sum_{i=1}^{n} \frac{|\varepsilon_i|}{n} \tag{5}$$

Residual variance (L_2):

In L_2, the average of the squared residuals from a multiple linear regressor is calculated. Lower values indicate simpler problems, as described in Equation 6.

$$L_2 = \sum_{i=1}^{n} \frac{\varepsilon_i^2}{n} \tag{6}$$

Output distribution (S_1):

Initially, a Minimum Spanning Tree (MST) is generated from the input data. Each data item corresponds to a vertex in the graph, while edges are weighted according to the Euclidean distance between examples in the input space. The MST connects the closest examples to each other. Finally, S_1 monitors whether the examples joined in the MST have similar output values. Lower values indicate simpler problems. This measure is expressed in Equation 7.

$$S_1 = \frac{1}{n} \sum_{i:j \in MST} |y_i - y_j| \tag{7}$$

Where the sum is taken over all vertices i and j that are adjacent in the MST. S_1 calculates the average of the outputs of the points connected in the MST.

Input distribution (S_2):

In this measure, the Euclidean distance between pairs of neighboring examples is calculated. To achieve this, the data points are initially sorted according to their output values. S_2 is presented in Equation 8.

$$S_2 = \frac{1}{n}\sum_{i=1}^{n} ||x_i - x_{i-1}||_2 \tag{8}$$

Error of a nearest neighbor regressor (S_3):

In S_3, the closeness of examples is measured. For this purpose, a 1-NN regressor looks for the training example (x_i) most similar to the new example and assigns it the same output (y_i), as shown in Equation 9.

$$S_3 = \frac{1}{n}\sum_{i=1}^{n}(NN(x_i) - y_i)^2 \tag{9}$$

Non-linearity of a linear regressor (L_3):

Given a dataset, pairs of examples with similar outputs are initially selected, and a new test point is created by performing random interpolation. The original data is then used to train a linear regressor, and the new points are used to measure the mean squared error (MSE).

$$L_3 = \frac{1}{l}\sum_{i=1}^{l}(f(x'_i) - y'_i)^2 \tag{10}$$

Where l is the number of interpolated examples, x'_i are the generated points, and y'_i are their labels. Lower values indicate simpler problems.

Non-linearity of nearest neighbor regressor (S_4):

This measure employs the same steps described for L_3, but uses a nearest neighbor regressor for predictions. S_4 is defined in Equation 11.

$$S_4 = \frac{1}{l}\sum_{i=1}^{l}(NN(x'_i) - y'_i)^2 \tag{11}$$

Average number of examples per dimension (T_2):

It is defined as the average number of examples (n) per dimension (d), as presented in Equation 14. Lower values indicate more complex datasets.

$$T_2 = \frac{n}{d} \tag{12}$$

Average number of PCA dimensions per data point (T_3):

The metric T_3 uses Principal Component Analysis (PCA) to assess dataset characteristics. Unlike T_2, which relies on the raw dimensionality of the feature vector, T_3 employs the number of PCA components required to capture 95% of data variability (denoted as m') as the foundation for evaluating data sparsity.

$$T_3 = \frac{m'}{n} \tag{13}$$

Ratio of PCA dimensions to the original dimensions (T_4):

This measure estimates the proportion of relevant dimensions within the dataset. The concept of relevance is evaluated based on the PCA criterion, which aims to transform features into uncorrelated linear functions that effectively describe the majority of data variability.

$$T_4 = \frac{m'}{m} \tag{14}$$

Table 21 – Characteristic, acronym, aggregation functions and description of meta-features.

Characteristic	Acronym	Aggregation Functions	Description
Simple	n.examples	-	Number of examples
	n.attributes	-	Number of attributes
	n.rare	-	Number of rare cases ($\phi > 0.8$)
	p.rare	-	Percentage of rare cases
Dataset Distribution	T2	-	Average number of examples per dimension
	T3	-	Average intrinsic dimension per number of examples
	T4	-	Proportion of intrinsic dimensionality
Correlation between Attributes and Targets	C2	{avg, max, min, sd}	Average feature correlation to the output
	C3	{avg, max, min, sd}	Individual feature efficiency
	C4	{avg}	Collective feature efficiency
Linear Regression-Related Performance Metrics	L1	{avg, max, min, sd}	Mean absolute error
	L2	{avg, max, min}	Residual variance
	L3	{avg, max, min, sd}	Non-linearity of a linear regressor
Data Smoothness	S1	{avg, max, min, sd}	Output distribution
	S2	{avg, max, min, sd}	Input distribution
	S3	{avg, max, min, sd}	Error of a nearest neighbor regressor
	S4	{avg, max, min, sd}	Non-linearity of nearest neighbor regressor
Prediction **	Model	-	Predicted learning model
	Strategy	-	Predicted resampling strategy

** Used only in Model First and Strategy First recommendation procedures.

Source: Prepared by the author.

APPENDIX C - META-MODELS EVALUATION

Table 22 – Performances (Accuracy) achieved by the meta-models for each type of training. Best results are in bold.

Approach	Label	DESMI	KNORAU	KNORAE	ET	BG	RF	XGB
Independent	r	0.376	0.385	0.376	0.417	0.408	0.417	**0.431**
Independent	l	0.376	0.394	0.417	0.477	0.440	**0.500**	0.431
Model_first	r	0.353	0.390	0.289	0.417	0.408	**0.454**	0.436
Model_first	l	0.372	0.385	0.367	0.482	0.454	**0.491**	0.431
Strategy_first	r	0.339	0.427	0.335	0.390	0.394	0.417	**0.431**
Strategy_first	l	0.408	0.422	0.390	**0.468**	0.450	0.459	0.431

Source: Prepared by the author.

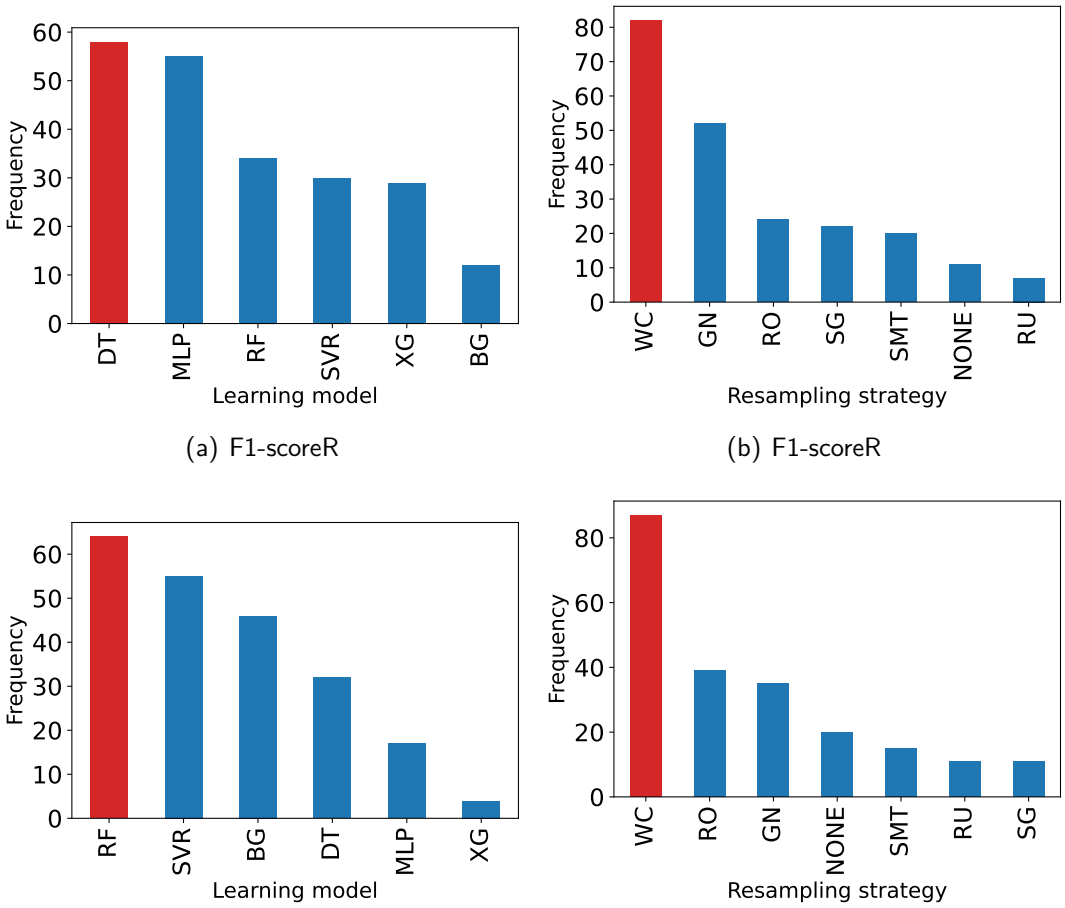
Table 23 – Performances (Accuracy) achieved by the meta-models for each type of training. Best results are in bold.

Approach	Label	DESMI	KNORAU	KNORAE	ET	BG	RF	XGB
Independent	r	0.330	0.339	0.303	0.353	0.317	**0.362**	0.335
Independent	l	0.367	0.404	0.427	0.440	0.440	**0.472**	0.394
Model_first	r	0.257	0.321	0.307	**0.376**	0.326	**0.376**	0.339
Model_first	l	0.404	0.381	0.394	**0.459**	0.417	0.454	0.394
Strategy_first	r	0.330	0.330	0.289	**0.362**	0.358	**0.362**	0.335
Strategy_first	l	0.376	0.413	0.344	0.436	0.427	**0.468**	0.408

Source: Prepared by the author.

APPENDIX D - META-TARGET DISTRIBUTION

Figure 27 – Frequency of resampling strategies and learning models used as meta-target.



(a) F1-scoreR (b) F1-scoreR

(c) SERA (d) SERA

Source: Prepared by the author.