U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

universidade de aveiro

Universidade do Minho

PhD

3rd CYCLE

FCUP
UA
UM
2025

U.PORTO

Learning from Imbalanced Regression Data Streams

Ehsan Aminian

FC

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

universidade de aveiro

Universidade do Minho

Learning from Imbalanced Regression Data Streams

Ehsan Aminian
Doctoral Program in Computer Science - MAP joint programme
Faculty of Sciences of the University of Porto, University of Aveiro and University of Minho
2025

PhD

Learning from Imbalanced Regression Data Streams

Ehsan Aminian
Thesis carried out as part of the Doctoral Program in Computer Science - MAP joint programme
Department of Computer Science
2025

Supervisor
João Gama, Full Professor, Faculty of Economics of the University of Porto

Co-supervisor
Rita P. Ribeiro, Assistant Professor, Faculty of Sciences of the University of Porto

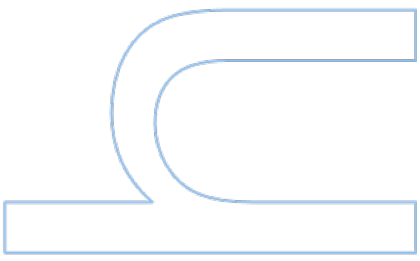
Note: This page should only be included **in the final version after defence and only if the jury approves the thesis with a recommendation for the student to correct errors, inaccuracies or formal inaccuracies** identified and expressly mentioned during the tests. **This page must be deleted from the version to be submitted as part of the thesis submission prior to the defence.**

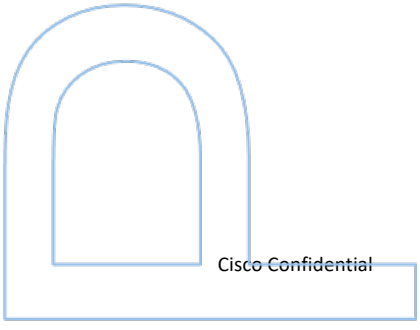
All the corrections determined by the jury, and only those, were made.

The Supervisor,

Porto, _____/_____/_____

Assinado por: **João Manuel Portela da Gama**
Num. de Identificação: 03155534
Data: 2025.09.17 00:23:21 +0100





Acknowledgements

I would like to begin by expressing my deepest gratitude to my family, whose unwavering love, support, and encouragement have been my greatest source of strength. To my wife, Mona, your patience, understanding, and constant belief in me have made this journey possible. To my parents, your lifelong guidance and care have shaped the person I am today. This work is a testament to your enduring support and love.

I am deeply grateful to my advisors, Professor João Gama and Professor Rita P. Ribeiro. Your expertise, guidance, and thoughtful feedback have been invaluable throughout this research. Your encouragement through every challenge and your belief in my abilities have been instrumental in shaping this work, and I am profoundly thankful for your mentorship.

Above all, I dedicate this work to the women of my homeland, Iran, whose resilience, courage, and unwavering determination continue to inspire generations. I especially honor Mahsa, Nika, Hadis, Sarina, Hananeh, Niloofar, and the thousands of other women who have sacrificed their lives and even their very sight in the pursuit of freedom and justice. Their courage and perseverance have been a constant source of inspiration, and it is my hope that this work, in some small way, honors their enduring spirit.

Resumo

O aumento da geração de dados contínuos provenientes de fontes como dispositivos IoT, plataformas de redes sociais e transações financeiras levou à necessidade de técnicas avançadas de mineração de fluxos de dados. Em particular, o tratamento de fluxos de dados de regressão desequilibradosonde valores raros e extremos são frequentemente mais significativos do que os frequentessurgiu como um desafio crítico. Enquanto grande parte da pesquisa existente se concentrou na classificação desequilibrada, esta tese aborda o domínio relativamente inexplorado da regressão desequilibrada em fluxos de dados.

Esta pesquisa introduz duas contribuições-chave visando melhorar a precisão das previsões para valores-alvo raros e extremos em fluxos de dados em tempo real. Primeiro, são propostas estratégias de subamostragem (`ChebyUS`) e superamostragem (`ChebyOS`) baseadas em Chebyshev, aproveitando a desigualdade de Chebyshev para diferenciar entre casos frequentes e raros com base na distância dos valores-alvo em relação à média. Estas estratégias permitem uma aprendizagem mais equilibrada, seja subamostrando casos frequentes ou superamostrando casos raros, melhorando o foco do modelo em instâncias críticas.

Reconhecendo as limitações de restringir casos raros aos valores extremos da distribuição-alvo, a tese introduz ainda estratégias de subamostragem (`HistUS`) e superamostragem (`HistOS`) baseadas em histograma. Estas abordagens ajustam-se dinamicamente à natureza evolutiva dos fluxos de dados, construindo um histograma online que orienta o processo de amostragem, garantindo que instâncias raras que ocorram em qualquer parte da distribuição sejam adequadamente representadas.

A eficácia destes métodos é avaliada integrando-os em quatro algoritmos de regressão amplamente utilizados. Experiências extensivas em conjuntos de dados sintéticos e do mundo real demonstram que as estratégias de amostragem propostas melhoram significativamente o desempenho dos modelos de regressão, particularmente na captura de valores raros e extremos. Os modelos treinados com estas estratégias superam consistentemente os modelos de base em várias métricas, sublinhando a importância de abordar o desequilíbrio em tarefas de regressão.

Esta tese avança o campo da mineração de fluxos de dados desequilibrados, oferecendo soluções práticas e escaláveis para aplicações em tempo real, como deteção de fraudes, diagnósticos médicos e monitorização ambiental, onde previsões precisas de eventos raros são cruciais.

Palavras-chave: Fluxos de Dados, Fluxos de Dados Desbalanceados, Regressão, Subamostragem e Sobreamostragem.

Abstract

The increasing generation of continuous data from sources such as IoT devices, social media platforms, and financial transactions has led to the need for advanced data stream mining techniques. In particular, handling imbalanced regression data streamswhere rare and extreme values are often more significant than frequent oneshas emerged as a critical challenge. While much-existing research has focused on imbalanced classification, this thesis addresses the relatively unexplored domain of imbalanced regression in data streams.

This research introduces two key contributions aimed at improving the prediction accuracy for rare and extreme target values in real-time data streams. First, Chebyshev-based undersampling (`ChebyUS`) and oversampling (`ChebyOS`) strategies are proposed, leveraging Chebyshev's inequality to differentiate between frequent and rare cases based on the distance of target values from the mean. These strategies enable more balanced learning by either undersampling frequent cases or oversampling rare ones, improving the models focus on critical instances.

Recognizing the limitations of confining rare cases to the extreme values of the target distribution, the thesis further introduces histogram-based undersampling (`HistUS`) and oversampling (`HistOS`) strategies. These approaches dynamically adjust to the evolving nature of data streams by building an online histogram that guides the sampling process, ensuring that rare instances occurring anywhere in the distribution are adequately represented.

The effectiveness of these methods is evaluated by integrating them into four widely used regression algorithms. Extensive experiments on synthetic and real-world datasets demonstrate that the proposed sampling strategies significantly improve the performance of regression models, particularly in capturing rare and extreme values. Models trained with these strategies consistently outperform baseline models across various metrics, underscoring the importance of addressing imbalances in regression tasks.

This thesis advances the field of imbalanced data stream mining, offering practical, scalable solutions for real-time applications such as fraud detection, medical diagnostics, and environmental monitoring, where accurate predictions of rare events are crucial.

Keywords: DImbalanced Data Streams, Regression, Undersampling, and Oversampling.

Contents

List of Tables **x**

List of Figures **xiii**

1 Introduction **1**

1.1 Motivation 2

1.2 Research Questions 3

1.3 Contributions 3

1.4 Thesis Outline 4

1.5 Publications 6

2 Literature Review **9**

2.1 Problem Definition 10

2.2 Regression from Data Streams: Basic Concepts 11

2.3 Regression from Data Streams: Related Work 12

2.4 Imbalanced Regression Strategies 16

2.4.1 Imbalanced Regression Problems 16

2.4.2 Data-level Strategies 17

2.4.2.1 SMOTE-Based Methods 19

2.4.2.2 Alternative Pre-Processing Methods . . . 22
2.4.3 Algorithm-level Strategies . . . 24
2.4.4 Hybrid Strategies . . . 25
2.5 Research Gaps . . . 28

3 Chebyshev Approaches for Imbalanced Data Streams Regression 31
3.1 Chebyshev's Inequality . . . 31
3.2 Proposed Methods . . . 33
3.2.1 ChebyUS: Chebyshev-based Undersampling . . . 34
3.2.2 ChebyOS: Chebyshev-based Oversampling . . . 35
3.3 Experimental Evaluation . . . 36
3.3.1 Relevance Function . . . 37
3.3.2 Data sets . . . 38
3.3.3 Experimental Setup . . . 38
3.3.4 Experimental Results . . . 41
3.3.4.1 Results for low or high extreme values . . . 47
3.3.4.2 Comparing ChebyUS against ChebyOS . . . 49
3.4 Limitations . . . 50

4 Histogram Approaches for Imbalanced Data Streams Regression 55
4.1 Histogram Approaches . . . 56
4.1.1 Probability Function . . . 56
4.1.2 HistUS: Undersampling Approach Using Online Histograms . . . 57
4.1.3 HistOS: Oversampling Approach Using Online Histograms . . . 58
4.2 Experimental Evaluation . . . 60

4.2.1 Relevance Function 61
4.2.2 Experiments on Synthetic and Real-World Data 61
4.2.2.1 Impact of parameter β on the probability function 66
4.2.2.2 Impact of parameter α on the oversampling rate 68
4.2.3 Experiments on Benchmark Data 68
4.2.3.1 Results 70
4.2.3.2 Comparative Analysis 72
4.3 Limitations 76

5 Conclusions and Future Work **79**
5.1 Conclusions 79
5.2 Answering Research Questions 81
5.3 Key Results 83
5.4 Future Work 84

Bibliography **85**

A The results from the Chebyshev Approach **97**
A.1 More about the data sets 97
A.2 Results of the experiments: sensitivity analysis varying ϕ. 98
A.3 CD diagramss 105
A.4 Target variable information for each dataset 106
A.5 Synthetic Data Generation 120
A.6 Range Queries Aggregates Dataset 121
A.7 Benchmark Datasets 123

A.8 Additional Results 126

List of Tables

3.1 Data sets characteristics: number of cases (#cases) and number of rare cases (#rare cases) with $thr_\phi = 0.8$ that are low extremes (i.e., below the median), high extremes (i.e., above the median), or both. 39
3.2 Summary of results of Chebyshev-based Undersampling (ChebyUS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.8$) and $RMSE$. 42
3.3 Summary of results of Chebyshev-based Oversampling (ChebyOS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.8$) and $RMSE$. 42
3.4 Summary of results of Chebyshev-based Undersampling (ChebyUS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.0$) and $RMSE$. 44
3.5 Summary of results of Chebyshev-based Oversampling (ChebyOS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.0$) and $RMSE$. 44
3.6 Summary of $RMSE_\phi$ results comparing ChebyUS and Baseline considering low and high extreme rare cases ($thr_\phi = 0.8$). 47
3.7 Summary of $RMSE_\phi$ results comparing ChebyOS and Baseline considering low and high extreme rare cases ($thr_\phi = 0.8$). 48

4.1 Summary of results of no sampling (Baseline) and sampling strategies (HistUS, HistOS, ChebyUS, ChebyOS) on the synthetic and Range Queries Aggregates (RQA) datasets: mean, standard deviation and average rank of $RMSE_\phi$, $SERA$, and $RMSE$ for the FIMT-DD algorithm. 67
4.2 Summary of results of histogram-based sampling strategy HistUS compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$, $SERA$, and $RMSE$. 70
4.3 Summary of results of histogram-based sampling strategy HistOS compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$, $SERA$, and $RMSE$. 71
4.4 Comparison of histogram-based sampling strategies: average rank and number of significant wins (#w) at a 95% confidence level, according to $RMSE_\phi$, $SERA$, and $RMSE$. 74
4.5 Comparison of Chebyshev-based and histogram-based Undersampling strategies: average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ and $SERA$. 75

4.6 Comparison of Chebyshev-based and histogram-based oversampling strategies: average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ and $SERA$. 76

A.1 Data Set Characteristics 98
A.2 Data sets characteristics: number of cases (#cases) and number of rare cases (#rare cases) with $thr_\phi = 0.9$ that are low extremes (i.e., below the median), high extremes (i.e., above the median), or both. 99
A.3 Data sets characteristics: number of cases (#cases) and number of rare cases (#rare cases) with $thr_\phi = 1.0$ that are low extremes (i.e., below the median), high extremes (i.e., above the median), or both. 99
A.4 $RMSE$ and $RMSE_\phi$ estimates considering all the observations: $thr_\phi = 0$. 100
A.5 $RMSE$ and $RMSE_\phi$ estimates considering `all` (double-side) rare cases: $thr_\phi = 0.8$.101
A.6 $RMSE$ and $RMSE_\phi$ estimates considering `all` (double-side) rare cases: $thr_\phi = 0.9$.102
A.7 $RMSE$ and $RMSE_\phi$ estimates considering `all` (double-side) rare cases: $thr_\phi = 1.0$.103
A.8 $RMSE_\phi$ estimates considering `left-hand side` rare cases ($thr_\phi = 0.8$). 104
A.9 $RMSE_\phi$ estimates considering `right-hand side` rare cases ($thr_\phi = 0.8$). 104
A.10 Data sets characterization. 123
A.11 Evaluation: $RMSE_\phi$, Method: HistUS 127
A.12 Evaluation: $RMSE_\phi$, Method: HistOS 128
A.13 Evaluation: $SERA$, Method: HistUS 129
A.14 Evaluation: $SERA$, Method: HistOS 130
A.15 Evaluation: $RMSE$, Method: HistUS 131
A.16 Evaluation: $RMSE$, Method: HistOS 132
A.17 Evaluation: $RMSE_\phi$, Method: HistUS vs HistOS 134
A.18 Evaluation: $SERA$, Method: HistUS vs HistOS 135
A.19 Evaluation: $RMSE$, Method: HistUS vs HistOS 136
A.20 Evaluation: $RMSE_\phi$, Methods: ChebyUS VS HistUS 138
A.21 Evaluation: $RMSE_\phi$, Methods: ChebyOS VS HistOS 139
A.22 Evaluation: $SERA$, Methods: ChebyUS VS HistUS 140
A.23 Evaluation: $SERA$, Methods: ChebyOS VS HistOS 141

List of Figures

2.1 Taxonomy of Approaches for Imbalanced Regression 18

3.1 Box plot and Chebyshevs probability for the target value of Fried data set. 33
3.2 Rare extreme target values cases for puma32H data set identified by $\phi()$ relevance function [10] that uses the target variable boxplot distribution. Setting a relevance threshold $thr_\phi = 0.8$ it is possible to identify the cases that correspond to both extremes, low extremes (i.e. below the median) and high extremes (i.e. above the median). 40
3.3 Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from no sampling (Baseline) against Chebyshev-based sampling (ChebyUS and ChebyOS) using $RMSE_\phi$ ($thr_\phi = 0.8$) and $RMSE$. 43
3.4 Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from no sampling (Baseline) against Chebyshev-based sampling (ChebyUS and ChebyOS) using $RMSE_\phi$ ($thr_\phi = 0.0$) and $RMSE$. 45
3.5 Wins and significant wins of the proposed ChebyUS (left) and ChebyOS (right) methods for different thr_ϕ values. 46
3.6 Total number of wins/losses, according to $RMSE_\phi$, of our sampling strategies considering low and high extreme rare cases ($thr_\phi = 0.8$). 48
3.7 CD diagrams of ChebyUS and ChebyOS strategies based on obtained $RMSE_\phi$ results over different extremes types (both, low and high) of rare cases and for different levels of thr_ϕ. 49
3.8 Average relative execution time of ChebyUS and ChebyOS for Perceptron models over all data sets. The term of comparison is the Perceptron, represented by the dotted line. 50

4.1 Synthetic dataset (a), histogram of target values (b) and respective relevance values (c) of common and rare cases considering thr_ϕ of 0.9. 63
4.2 Comparison of Probability and K values between the Chebyshev-based and the histogram-based approaches for the synthetic dataset. 64
4.3 True target values and respective predictions using the Chebyshev-based sampling and histogram-based sampling for the synthetic dataset. The orange-highlighted areas in the background correspond to the rare regions in the target value domain. 65
4.4 Influence of different values of the sampling decay parameter β on the probability values obtained over the synthetic dataset. 67

4.5 Effect of different values of the reduction coefficient parameter α on the oversampling rate. The parameter β is fixed at 4, while α is varied to analyze its impact on oversampling frequency across target values. The orange-highlighted areas in the background correspond to the rare regions in the target value domain. 68
4.6 Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from no sampling (Baseline) against histogram-based sampling (HistUS and HistOS) using $RMSE_\phi$, $SERA$, and $RMSE$. 71
4.7 Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from histogram-based oversampling (HistOS) *versus* histogram-based undersampling (HistUS) using $RMSE_\phi$, $SERA$, and $RMSE$. 73
4.8 Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from Chebyshev-based sampling *versus* histogram-based undersampling using $RMSE_\phi$, $SERA$. 75

A.1 CD diagrams based on obtained $RMSE_\phi$ results considering both extreme rare cases ($thr_\phi = 0.8$) 105
A.2 CD diagrams based on obtained $RMSE_\phi$ considering low extreme rare cases ($thr_\phi = 0.8$) 105
A.3 CD diagrams based on the obtained $RMSE_\phi$ results considering high extreme rare cases ($thr_\phi = 0.8$) 105
A.4 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **puma32H** data set 106
A.5 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **cpusum** data set 107
A.6 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **elevator** data set 108
A.7 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **bike** data set 109
A.8 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **energy** data set 110
A.9 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **calhousing** data set 111
A.10 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **gasemission** data set 112
A.11 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **mv** data set 113
A.12 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **fried** data set 114
A.13 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **pollution** data set 115
A.14 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **car price** data set 116
A.15 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **query** data set 117
A.16 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **GPU** data set 118
A.17 Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **3d spatial network** data set 119

A.18 Comparison of Probability and K values between the Chebyshev-based and the histogram-based approaches for the synthetic dataset. 121
A.19 Histogram of target values (a) and respective relevance values (b) of common and rare cases considering thr_ϕ of 0.9 for "Range Queries Aggregates" dataset . . 122
A.20 Histograms of the target values for all datasets 124
A.21 Relevance values of the target variable for all datasets. The dashed orange lines represent the threshold ($thr_\phi = 0.9$) used to distinguish rare cases. 125
A.22 CD diagrams based on the obtained $RMSE_\phi$ results, illustrating the performance comparison between the HistUS (US) and HistOS (OS) sampling strategies and their respective baselines (B). 133
A.23 CD diagrams based on the obtained $SERA$ results, illustrating the performance comparison between the `HistUS` and `HistOS` sampling strategies and their respective baselines. 133
A.24 CD diagrams based on the obtained $RMSE$ results, illustrating the performance comparison between the `HistUS` and `HistOS` sampling strategies and their respective baselines. 133
A.25 CD diagrams comparing the `HistUS` and `HistOS` methods based on $RMSE_\phi$, $SERA$, and $RMSE$ evaluation metrics. 137
A.26 CD diagrams comparing ChebyUS, ChebyOS, HistUS, and HistOS methods based on $RMSE_\phi$ and $SERA$ evaluation metrics. 142

Chapter 1

Introduction

In various application domains, predicting rare events is often more critical than forecasting frequent occurrences. This significance is particularly evident in areas such as financial fraud detection [1], where identifying uncommon fraudulent activities can prevent substantial monetary losses. Similarly, the prediction of rare medical conditions [2] plays a crucial role in timely interventions and treatment advancements. Additionally, understanding extreme environmental phenomena, such as catastrophic flood [3] is essential for effective disaster management and mitigation strategies. These challenges are encapsulated within the concept of Imbalanced Domain Learning, which has emerged as a prominent concern in the machine learning community due to the complexities involved in accurately modeling and predicting such rare events [4]. In the regression context, the problem arises when the distribution of the target variable is highly skewed, leading to a scenario in which specific values or ranges of the target variable are considerably underrepresented compared to others. Unlike standard regression tasks that treat all target values equally, imbalanced regression requires special consideration of rare cases that often carry critical domain significance. The fundamental problem lies in the conflict between standard regression objectives, which aim to optimize global error metrics such as mean squared error, and the need to prioritize accurate predictions for sparsely represented target values. This imbalance poses three key challenges: i) identifying the rare and relevant regions within the continuous domain of the target variable; ii) overcoming the bias of standard regression algorithms towards predicting the most frequent cases; and, iii) the need for proper evaluation metrics given the inadequacy of standard error metrics in measuring the models' performance on these rare values.

Tackling imbalanced regression is especially challenging in the context of data streams, where information arrives continuously and models must process data in real-time. The digital age has resulted in an influx of data from sources such as IoT devices, social media, and sensors, leading to continuous, high-speed, and potentially limitless data streams. In data stream regression, the goal is to predict a target variable from sequentially incoming data instances continuously. This paradigm diverges fundamentally from traditional batch regression by imposing three critical operational constraints: single-pass instance processing, strict memory limitations, and continuous adaptation to non-stationary data distributions [5, 6]. Incremental learning emerges as the core methodology addressing these operational requirements. It is categorized into *instance-incremental* learning, where the model updates with each new instance, and *batch-incremental* learning, which updates in small chunks to balance efficiency and robustness [5, 7]. Unlike traditional batch evaluation, data stream regression employs *prequential evaluation* [8], where predictions are made before true labels are revealed, enabling real-time model updates [9].

This Thesis tackles the combined challenges of data stream learning and imbalanced domain learning by focusing on *imbalanced regression in data streams*, a domain where prior works have largely centered on classification [10], leaving regression understudied. In the following section, we present the motivation behind this research, highlighting the challenges and significance of imbalanced regression in data streams. Next, we introduce the key research questions that guide this thesis, followed by a summary of the main contributions. The chapter concludes with an overview of the thesis structure and a list of publications resulting from this work.

1.1 Motivation

While extensive research has been dedicated to imbalanced learning in classification tasks, the regression counterpart remains relatively underexplored. Existing methods for imbalanced regression, such as SMOTE-based techniques and relevance-weighted sampling, are primarily designed for static datasets. On the other hand, the existing research on the data stream context does not address imbalanced learning. In other words, while previous research exists on data streams and imbalanced learning separately, the intersection of these two areasimbalanced learning within a data stream contexthas not, to our knowledge, been explored.

This thesis is motivated by the need to develop effective learning strategies for imbalanced regression in data streams. The research aims to design novel sampling techniques that

enhance model performance on rare and extreme target values while maintaining computational efficiency for real-time applications. By leveraging statistical principles and adaptive sampling mechanisms, this work seeks to bridge the gap between theoretical advancements in imbalanced learning and the practical demands of streaming environments, ensuring that predictive models remain robust, adaptive, and aligned with domain-specific priorities. This research advances the field of data stream mining and empowers decision-makers to act on critical insights derived from the most consequentialyet underrepresentedaspects of their data.

1.2 Research Questions

The primary objective of this thesis is to develop effective strategies for learning from imbalanced regression data streams. Addressing this challenge requires a comprehensive understanding of how different factors influence the learning process. To this end, the following research questions are posed:

- **RQ1**: How do probability distributions make traditional learning methods unsuitable for learning from imbalanced data streams?
- **RQ2**: How do different sets of rare examples impact the performance of the proposed methods?
- **RQ3**: How does the choice of sampling strategy (Chebyshev-based vs. Histogram-based) affect the predictive performance of regression models?
- **RQ4**: In general, are the proposed methods effective in coping with imbalanced regression problems?

1.3 Contributions

The contributions of this thesis are twofold. First, it introduces a Chebyshev-based sampling strategy for imbalanced data stream regression. This approach leverages Chebyshev's inequality to guide the selection of rare and extreme target values during training, ensuring that models pay adequate attention to these critical instances. This method is particularly effective in scenarios where rare events are confined to the extremes of the target distribution, such as in financial loss predictions or extreme weather forecasting. Second, the thesis presents a novel histogram-based

approach to sampling for imbalanced regression tasks in data streams. This method overcomes the limitations of previous techniques by allowing rare cases to appear anywhere within the target variables distribution, not just at the extremes. By constructing an online histogram that dynamically adjusts to incoming data, the proposed technique enables both undersampling and oversampling based on the frequency of target values. This approach is more flexible and generalizable, making it applicable to a broader range of real-world scenarios where rare but important events can occur throughout the data distribution. Both approaches aim to improve the performance of regression models in imbalanced data streams by ensuring that rare cases are adequately represented during the learning process. This work advances the state-of-the-art in imbalanced data stream mining, particularly in the regression domain, and offers practical solutions for real-time applications that require accurate predictions of rare and significant events.

1.4 Thesis Outline

The first chapter lays the groundwork for the research by defining the problem and its significance. It underscores the challenges of imbalanced regression in streaming environments, where traditional machine learning models often fail to generalize effectively due to the skewed distribution of target values. The chapter emphasizes the critical need for predictive models capable of accurately identifying and adapting to rare and extreme target values in real time, a capability essential for applications such as anomaly detection, financial forecasting, and environmental monitoring. Additionally, it presents the key research questions guiding this study and outlines the contributions of the thesis, providing a clear roadmap for the subsequent chapters. Finally, the chapter concludes with a list of publications stemming from this research.

The second chapter reviews existing approaches to handling imbalanced regression. It discusses data-level methods that manipulate training data distribution, algorithm-level techniques that modify learning algorithms, and hybrid strategies that combine both approaches. Several well-known methods, including SMOTE-based techniques, relevance-based sampling, and ensemble methods, are examined in detail. The chapter also identifies the limitations of these existing methods, particularly their applicability to continuous data streams where real-time learning is required.

The third chapter introduces the first proposed approach based on Chebyshev's inequality. The theoretical foundation of this statistical measure is explained, demonstrating how it can be leveraged to define rare instances in a regression context. Two novel sampling techniques, ChebyUS and ChebyOS, are proposed to enhance learning by selectively undersampling frequent cases and oversampling rare ones. The implementation of these methods is described, and their effectiveness is evaluated through extensive experiments, analyzing their impact on predictive accuracy and robustness across various datasets.

The fourth chapter presents the second approach employing histogram-based sampling. Unlike the Chebyshev-based approach, which relies on statistical assumptions, the histogram-based methods dynamically construct an online histogram to guide the sampling process. Two techniques, HistUS and HistOS, are introduced, providing an adaptive mechanism for identifying and resampling rare cases. The chapter describes the implementation of these methods and explores their advantages in handling data streams where the rare regions may be located anywhere in the distribution of target values. A detailed experimental evaluation is conducted, comparing the performance of these methods with the previously proposed approaches and other baseline models.

The final chapter summarizes the key findings of this research. It discusses the effectiveness of the proposed methods in addressing the challenges of imbalanced regression in data streams and highlights the trade-offs associated with different sampling strategies. The chapter also outlines the limitations of the study, considering factors such as parameter sensitivity and the potential for further improvements. Future research directions are proposed, including developing adaptive frameworks that adjust to concept drift in streaming data environments.

1.5 Publications

The following publications have resulted from this research:

Journal Articles

- Ehsan Aminian, Rita P. Ribeiro, and João Gama. "Chebyshev approaches for imbalanced data streams regression models," *Data Mining and Knowledge Discovery*, vol. 35, no. 6, pp. 2389–2466, 2021. [Springer]

Conference Papers

- Rita P. Ribeiro, Saulo Martiello Mastelini, Narjes Davari, Ehsan Aminian, Bruno Veloso, and João Gama. "Online anomaly explanation: a case study on predictive maintenance," *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 383–399, 2022. [Springer]
- João Gama, Bruno Veloso, Ehsan Aminian, and Rita P. Ribeiro. "Current trends in learning from data streams," *International Conference on Big Data Analytics*, pp. 183–193, 2021. [Springer]
- Ehsan Aminian, Rita P. Ribeiro, and João Gama. "A study on imbalanced data streams," *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases - Workshops*, pp. 380–389, 2019. [Springer]

Under Review

- Ehsan Aminian, Rita P. Ribeiro, and João Gama. "Histogram Approaches for Imbalanced Data Streams Regression" submitted to *Machine Learning*, Springer, 2025.

Chapter 2

Literature Review

This chapter provides an overview of regression from data streams, addressing both fundamental principles and advanced strategies for handling challenges such as evolving data distributions and imbalanced target variables. We begin with a mathematical definition of the problem in Section 2.1, introducing key concepts such as the relevance function and the distribution of rare target values. The discussion then transitions into a discussion on the basic concepts of regression in data streams, including the unique constraints posed by continuous and high-velocity data in Section 2.2.

Following this, the chapter presents a review of existing approaches to regression in data streams, covering both foundational algorithms and more recent advancements in Section 2.3. Traditional methods like incremental regression trees and ensemble learning techniques are discussed alongside adaptive strategies designed to handle concept drift.

We move our focus into another challenge termed *Imbalanced regression* in Section 2.4. A critical aspect of imbalanced regression is the identification of rare instances within a continuous target space. Unlike classification problems, where the disproportionate representation of classes defines the imbalance, regression involves a continuous distribution, making the identification of rare values more complex. Specialized methodologies are required to determine what constitutes a rare instance within a given data distribution. The section explores various techniques for assessing instance rarity, including statistical measures, density estimation methods, and relevance-based functions that help differentiate between frequent and infrequent values. The section then transitions into an exploration of strategies for handling imbalanced

regression. Various approaches are discussed, including data-level methods that manipulate training distributions through sampling techniques, algorithm-level modifications that adjust model learning processes, and hybrid strategies that combine both perspectives.

Finally, Section 2.5 concludes by identifying research gaps in the field of data stream regression, emphasizing the need for scalable, adaptive, and interpretable methods that can effectively handle both evolving distributions and imbalanced target values.

2.1 Problem Definition

Existing methodologies for imbalanced regression predominantly assume static datasets where global statistics and complete data access enable precise rarity estimation. However, these assumptions collapse in online mode. The challenge of *imbalanced domain learning* arises within the context of predictive tasks. Given a training set $D = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathcal{X}$ represents a feature vector consisting of predictor (independent) variables, and $y_i \in \mathcal{Y}$ represents the target (dependent) variable, the objective is to accurately approximate the unknown function $f : \mathcal{X} \rightarrow \mathcal{Y}$. Depending on the nature of the target variable, this leads to a classification problem (if $\mathcal{Y}$ is discrete) or a regression problem (if $\mathcal{Y}$ is continuous) [11].

In standard regression, the optimal approximation to f is typically attained by minimizing overall error estimates, such as mean absolute error or mean squared error. Consequently, the model prioritises minimizing prediction errors for frequent cases, significantly impacting the overall performance metrics. However, this approach does not directly translate to *imbalanced regression* domains, where the distribution of the target variable values is highly uneven, and accurate predictions on the underrepresented (rare) target values are of particular importance [10, 12].

The complexity increases in the context of *online imbalanced regression*, where the entire dataset is not available at once for model training. Instead, a data-generating process provides a sequence of examples in the form of pairs $\langle \mathbf{x}_t, y_t \rangle$, one at a time, drawn from an unknown probability distribution $p_t(\mathbf{x}, y)$. Here, $\mathbf{x}_t \in \mathcal{X}$ represents a p-dimensional feature vector observed at time t, and $y_t \in \mathbb{R}$ denotes its corresponding continuous target value.

In this setting, the learner aims to approximate the function f sequentially. Let H_t be the online learner at time t, maintaining a hypothesis $H_t : \mathcal{X} \rightarrow \mathcal{Y}$, which is updated with each new

example. When an example $\langle \mathbf{x}_t, y_t \rangle$ arrives at time step t, the learner predicts the output value $\hat{y}_t = H_t(\mathbf{x}_t)$. After making the prediction, the learner receives the true value y_t, which is then used to evaluate the predictive performance and to update the hypothesis for the next time step, H_{t+1}. This iterative process continues as new data arrives.

The problem addressed in this work is the development of effective methods for *online imbalanced regression*, where the goal is to improve predictive performance, particularly on the underrepresented (rare) target values, in a sequential learning environment. This involves designing algorithms that can learn from data streams, handle the imbalanced distribution of the target variable, and update the model incrementally as new data becomes available.

2.2 Regression from Data Streams: Basic Concepts

Regression in data streams refers to the task of continuously predicting a target variable from incoming data instances, where data arrives sequentially in real-time. Unlike traditional batch regression, this paradigm requires algorithms to process instances in a single pass, maintain limited memory footprints, and dynamically adapt to evolving data distributions [5, 6]. A data stream is characterized by its unbounded nature, requiring models that can learn incrementally without the possibility of storing or revisiting past observations. Since data evolves over time, a phenomenon known as concept drift can occur, where the underlying distribution of the target variable changes, leading to degradation in model performance. Efficient mechanisms like drift detection (e.g., ADWIN2 [13]) or model reset strategies [5] for adaptation to these changes is essential for maintaining predictive accuracy.

Incremental learning is the core approach for data stream regression. Two main strategies exist: instance-incremental learning, where the model updates continuously with each new instance, and batch-incremental learning, where updates occur over small chunks of data to balance efficiency and robustness. These approaches ensure that models remain up-to-date without requiring full retraining [5, 7]. Handling memory constraints is another critical aspect of data stream regression. Since storing an entire data stream is infeasible, models rely on strategies such as sliding windows [14], landmark windows [15], and damped windows [16] to retain only relevant data points. Sliding windows keep only the most recent observations, while landmark windows define a reference point from which data is retained. Damped windows assign decreasing weights to older data, ensuring that recent observations have a greater

influence on model updates. Evaluation in data stream regression differs from traditional batch settings. The prequential evaluation method is commonly used, where the model predicts each incoming instance before receiving its true label, updating the model dynamically [9]. This approach allows for real-time performance assessment without requiring separate training and testing phases. Metrics such as mean squared error and root mean squared error remain standard, but additional considerations, such as adaptive drift detection mechanisms, help monitor performance over time.

In summary, regression from data streams presents unique challenges requiring models to be adaptive, memory-efficient, and capable of handling evolving distributions. The following section provides an overview of related work addressing these challenges and advancements in this field.

2.3 Regression from Data Streams: Related Work

Online Bagging (OBag) [17] is a streaming adaptation of bootstrap aggregation designed to enhance model diversity in data streams. Unlike traditional bagging, which resamples the training set, OBag assigns Poisson-distributed instance weights to simulate resampling in an online setting. This approach allows base learners, such as FIMT-DD or Hoeffding Trees, to be trained on dynamically weighted subsets, improving their generalization. However, OBag does not inherently address concept drift and may struggle in scenarios with delayed labels, where the relevance of instances changes over time.

The Fast and Incremental Model Trees with Drift Detection (FIMT-DD) [18] represents a foundational approach for regression in data streams. It incrementally constructs regression trees using Hoeffding bounds to determine statistically significant splits based on variance reduction. Leaf nodes accumulate sufficient statistics, including means and variances, until a predefined grace period expires, after which potential splits are evaluated. To address concept drift, FIMT-DD employs an adaptive mechanism that monitors variance changes in leaf nodes. Underperforming subtrees are pruned and reset when significant drift is detected to ensure the model adapts to recent trends. While FIMT-DD is computationally efficient, its reliance on manually set drift thresholds and its sensitivity to abrupt changes can limit its applicability in highly dynamic environments.

The Online Random Forest (ORF) [19] extends FIMT-DD by constructing an ensemble of incrementally trained regression trees, where each tree receives a random subset of instances for training. By leveraging diversity in tree structures, ORF enhances robustness against noise and varying data distributions. However, ORF lacks explicit drift detection and adaptation mechanisms, relying solely on ensemble redundancy to mitigate performance degradation. As a result, it may struggle with rapidly evolving data distributions, and older trees may become obsolete over time.

Online Option Trees with Averaging (ORTO-A) [19] introduces a hierarchical ensemble structure in which option trees generate multiple predictions per instance. These predictions are averaged to produce final outputs, increasing robustness to noise and local concept shifts. By maintaining multiple hypotheses in parallel, ORTO-A adapts better to uncertain environments. However, this redundancy requires additional memory, as multiple active tree branches must be stored and updated continuously.

The Adaptive Random Forest Regressor (ARF-Reg) [20] improves upon ORF by integrating the ADaptive WINdowing (ADWIN) drift detection method into each tree. ADWIN continuously monitors changes in error rates, triggering model updates when a significant shift is detected. ARF-Reg discards the affected tree when drift occurs and replaces it with a background model trained on the most recent data. This dynamic adaptation improves performance in non-stationary environments, reducing error rates compared to static ensembles. However, ARF-Reg increases computational overhead due to the continuous evaluation of drift detection and background model maintenance.

Osojnik et al. [21] propose *iSOUP-PCT*, an incremental predictive clustering tree method for online semi-supervised multi-target regression. This method addresses the challenge of learning in data streams where labelled data is scarce and costly while unlabeled data is abundant. Extending the supervised *iSOUP-Tree*, *iSOUP-PCT* incorporates the predictive clustering framework, modifying the splitting heuristic to simultaneously optimise homogeneity in input attributes and target variables. This adaptation enables the model to leverage unlabeled examples effectively, improving predictive performance in partially supervised settings. A primary innovation of *iSOUP-PCT* is the adaptation of the intra-cluster variance reduction (ICVR) heuristic, balancing variance reduction in both the target and input spaces through a weighted parameter w. This mechanism allows the semi-supervised variant (*iSOUP-PCT*^{SSL}) to refine its structure using unlabeled examples while maintaining predictive consistency with the available

labelled data. Additionally, the authors introduce an "oracle" baseline (*iSOUP-Tree*^{Oracle}), which assumes access to all labels, serving as an upper bound for evaluating performance.

Osojnik et al. [22] also introduces another method abbreviated *iSOUP-SymRF* for online feature ranking in the context of multi-target regression. Unlike traditional batch learning approaches, which have been extensively studied for feature ranking, online learning settings have received less attention, particularly for structured output prediction tasks such as multi-target regression. The proposed method, *iSOUP-SymRF*, leverages the structure of a random forest composed of *iSOUP-Trees* to estimate feature importance scores. These scores are derived from the frequency and position of features in the ensemble's split nodes of the trees. The method is designed to handle data streams and can be extended to other structured output prediction tasks, such as multi-label classification and hierarchical multi-target regression, due to the versatility of the underlying *iSOUP-Tree* model.

The work by Stepišnik et al. [23]. Introduces *Option Predictive Clustering Trees* (OPCTs), another approach to MTR that addresses the limitations of traditional decision trees while preserving interpretability. Standard predictive clustering trees (PCTs) for MTR suffer from myopia due to their greedy split selection, often resulting in suboptimal models. To mitigate this, OPCTs incorporate *option nodes*, which allow multiple candidate splits within a single node, effectively balancing interpretability and predictive performance. These nodes aggregate predictions from alternative splits, acting as a condensed representation of an ensemble while retaining a tree-like structure. The authors propose that OPCTs reduce variance by leveraging overlapping hierarchical clusters, thereby approximating the benefits of ensemble methods like bagging and random forests without sacrificing transparency.

Self Parameter Tuning (SPT) [24] automates hyperparameter optimization for streaming regression models, dynamically adjusting parameters such as ensemble size and learning rates based on real-time performance feedback. It employs meta-learning strategies to balance exploration and exploitation, optimizing model behaviour without manual intervention. While SPT reduces the burden of parameter tuning, it introduces additional computational costs due to continuous evaluation and adaptation.

The paper by Stevanoski et al. [25] addresses the challenge of concept drift in multi-target regression (MTR) tasks within data streams. While existing methods for multi-target prediction on data streams have emerged, they often lack mechanisms for detecting and adapting to such changes. The authors propose two novel families of methods for change detection and

adaptation in the context of incremental online learning of decision trees for MTR. The first approach is ensemble-based, utilizing the ADWIN change detection mechanism, while the second approach integrates the Page-Hinckley test directly into the iSOUP-Tree framework. The ensemble-based approach, iSOUP-ADWIN, combines the iSOUP-Tree method with ADWIN. This parameter-free change detection mechanism uses a sliding window to monitor the error of base models. When a change is detected, the least accurate model in the ensemble is replaced. The second approach, Adaptive-iSOUP, extends the iSOUP-Tree by incorporating the Page-Hinckley test for change detection at every internal node of the tree. This allows for local and global concept drift detection, with adaptation achieved by growing alternative subtrees in parallel when drift is detected. The authors also introduce ensemble variants of Adaptive-iSOUP, namely Adaptive-iSOUP-Bag and Adaptive-iSOUP-RF, which apply the adaptive tree method within bagging and random forest frameworks.

Histograms are widely used in *batch processing* to approximate data distributions by partitioning static datasets into discrete bins. However, adapting histograms to *evolving data streams* poses significant challenges, as traditional methods rely on multiple data passes and full dataset storageboth infeasible in streaming environments. In online settings, histograms must incrementally update their bin structures in a single pass while operating under strict memory constraints. Garofalakis et al. [26] pioneered *incremental histogram techniques* to address these challenges, enabling dynamic adjustments to bin boundaries and frequencies as new data arrives. While their methods advanced real-time summarization, they struggled with *non-uniform or skewed distributions*, where abrupt variations in data density or concept drift (e.g., sudden shifts in underlying patterns) rendered static binning strategies ineffective [27].

A substantial body of literature addresses classification tasks in data streams with a focus on handling class imbalance. To conclude this section, we briefly review resampling methods which are more relevant to our study. Resampling techniques are widely employed to mitigate class imbalance in data streams by either oversampling the minority class or undersampling the majority class. Incremental Oversampling for Data Streams (IOSDS) [28] selectively replicates minority instances while filtering out noisy or overlapping examples to maintain model stability. Selection-Based Resampling (SRE) [29] iteratively removes majority instances, preventing excessive bias toward the minority class. Streaming adaptations of the well-known SMOTE algorithm have also been developed to address evolving class distributions. Adaptive Windowing SMOTE (AWSMOTE) [30] and Continuous SMOTE (C-SMOTE) [31] dynamically generate synthetic instances based on changes in class distribution. Very Fast Continuous SMOTE

(VFC-SMOTE) [32] further enhances adaptability by incorporating data sketching techniques, improving responsiveness to sudden shifts in data distributions.

2.4 Imbalanced Regression Strategies

In this section, we define imbalanced regression problems and elucidate the key distinctions that set them apart from standard regression tasks. We then explore the inherent challenges associated with imbalanced regression and outline various strategies to address them. Following this, we outline the primary strategy employed to address these challenges and provide an overview of different methods designed to tackle imbalanced regression problems based on those strategies.

2.4.1 Imbalanced Regression Problems

Imbalanced regression problems arise when the distribution of the target variable in a regression task is highly skewed, leading to a scenario where certain values or ranges of the target variable are significantly underrepresented compared to others. Unlike standard regression tasks that treat all target values equally, imbalanced regression requires special consideration of rare cases that often carry critical domain significance, such as extreme weather events in climatology or fraudulent transactions in finance. As previously discussed, this imbalance presents three major difficulties. First, identifying the rare and relevant regions within the continuous domain of the target variable is nontrivial. Second, standard regression algorithms are inherently biased toward predicting the most frequent cases, often leading to poor performance on rare values. Finally, traditional error metrics are inadequate for evaluating model performance in these scenarios, highlighting the need for more suitable evaluation measures.

To address the first challenge, researchers have developed three principal strategies for imbalanced regression. Kernel Density Estimation (KDE) approaches like *DenseWeight* [33] quantify instance rarity through probability density functions, automatically weighting low-density regions without distributional assumptions. Relevance-based methods, exemplified by Ribeiro [34]'s boxplot-derived functions and the SmoteR framework [35], employ piecewise mappings that assign importance scores based on statistical dispersion measures or domain knowledge. These functions enable explicit control over rare value thresholds through parameters like the

interquartile range multiplier. The third strategy combines these concepts through distribution-aware binning techniques like *Relevance-based Stratification* [36], which partitions the target space into regions of varying importance.

To address the second challenge, several approaches have been proposed, which can be broadly categorized into data-level, algorithm-level, and hybrid techniques. The relationships between these different approaches are illustrated in Figure 2.1, which provides a taxonomy of existing methods for handling imbalanced regression. This classification highlights the diverse range of techniques that have been proposed and sets the foundation for the discussion of their details in subsequent sections. However, this classification faces challenges due to significant overlap among ensemble learning, data-level, and algorithm-level strategies. For instance, ensemble techniques frequently integrate algorithmic-level adaptations (e.g., modifying loss functions) or employ data pre-processing (e.g., resampling subsets of data)[37] when training individual base models. The next subsections explore these methods in detail.

The third challenge lies in effectively evaluating model performance in predicting rare target values. To address this, specialized evaluation criteria have been proposed, including the $\phi()$-weighted Root Mean Square Error ($RMSE_\phi$) [34] and the Squared Error Relevance Area ($SERA$) [10]. $RMSE_\phi$ emphasizes underrepresented regions by incorporating a relevance function $\phi() \in [0, 1]$, derived from the target values' boxplot, into the standard $RMSE$. It only considers instances with relevance values above a given threshold, reweighting their prediction errors by multiplying the prediction error for each example by the relevance value assigned to it. $SERA$, in contrast, evaluates errors across the entire domain while placing greater emphasis on relevant instances. Unlike $RMSE_\phi$, which applies a single threshold, $SERA$ progressively integrates errors at multiple relevance levels, capturing both overall predictive performance and the models ability to handle rare cases. Consequently, $SERA$ serves as an intermediate metric between standard $RMSE$ and $RMSE_\phi$, balancing global error assessment with a focus on rare target values.

2.4.2 Data-level Strategies

Extensive research has been conducted to address the challenge of imbalanced learning, with data-level approaches being one of the most widely used strategies in both classification and regression tasks [38]. These methods modify the dataset itself rather than altering the learning algorithm, making them model-agnostic solutions for handling imbalanced distributions. In

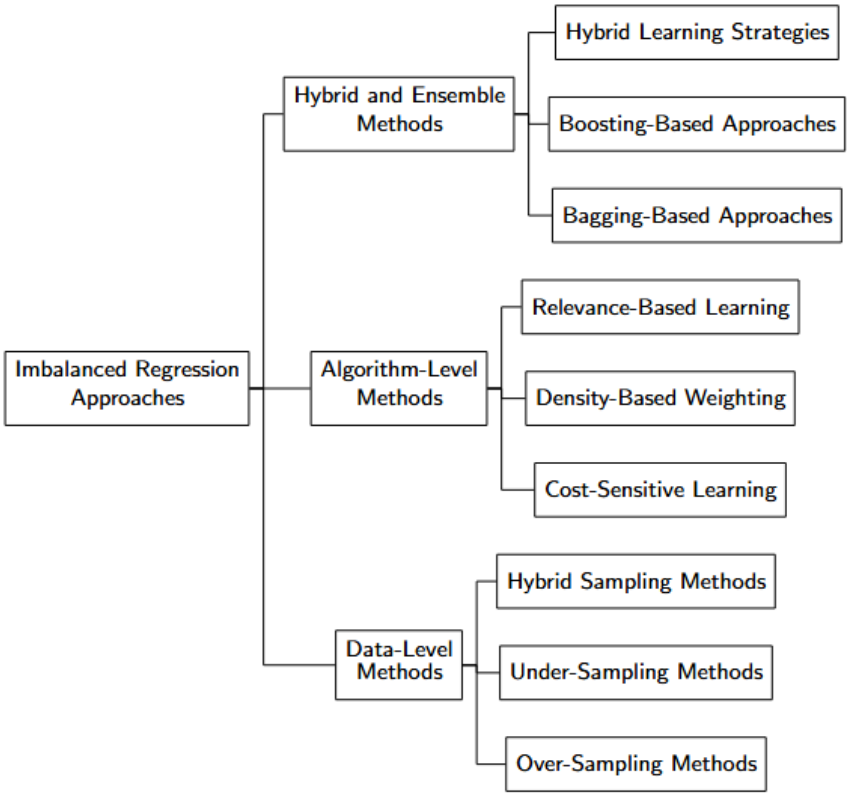


FIGURE 2.1: Taxonomy of Approaches for Imbalanced Regression

other words, these techniques offer the flexibility to use a wide range of learning algorithms, without compromising model interpretability [36]. Among data-level strategies, sampling techniques are the most commonly applied. Oversampling increases the presence of rare instances by replicating them within the dataset, allowing the learning model to better capture their characteristics [39]. In contrast, undersampling mitigates the dominance of frequent instances by selectively removing them [40], thereby preventing the model from being overly biased toward majority cases. Some approaches also combine both techniques to balance the dataset more effectively [41], ensuring that rare instances gain better representation without excessively inflating the data size.

Since many data-level techniques in regression are inspired by the Synthetic Minority Oversampling Technique (SMOTE), I have structured the following discussion into two categories. First, in the *SMOTE-Based Methods* subsection, we present SMOTE and its adaptations specifically designed for regression tasks. Then, in the *Alternative Pre-Processing Methods* subsection, we explore other resampling techniques that do not rely on SMOTE but offer different strategies to address the imbalance problem.

2.4.2.1 SMOTE-Based Methods

The Synthetic Minority Oversampling Technique (SMOTE) [41] is a widely recognized method designed to address the challenge of class imbalance in classification tasks. Unlike traditional oversampling methods that duplicate minority class instances, SMOTE generates synthetic examples of the minority class, thereby reducing the risk of overfitting and improving the model's ability to generalize. The process begins by identifying instances of the minority class. For each minority instance x_i, SMOTE locates its k-nearest neighbours within the same class, where k is a user-defined parameter typically set to 5. Next, synthetic instances are created by interpolating between x_i and a randomly selected neighbour x_{zi}. This interpolation is achieved using the formula:

$$x_{\text{new}} = x_i + \delta \times (x_{zi} - x_i),$$

where δ is a random number between 0 and 1. This ensures that the synthetic instance x_{new} lies somewhere along the line segment connecting x_i and x_{zi}, introducing variability while maintaining the minority class's distribution. The process is repeated iteratively for each minority instance until the dataset is sufficiently balanced. One advantage of SMOTE is its ability to improve the representation of the minority class without merely duplicating existing instances. This approach helps machine learning models learn a more robust decision boundary, enhancing their performance on imbalanced datasets. Additionally, by generating synthetic samples, SMOTE introduces diversity into the minority class, which mitigates the risk of overfitting to specific examples. However, SMOTE is not without limitations. A notable drawback is the potential for class overlap, as the method generates synthetic samples without considering the distribution of the majority class. This can lead to ambiguous regions in the feature space, increasing the likelihood of misclassifications. Furthermore, SMOTE's reliance on interpolation makes it unsuitable for datasets with categorical variables unless modified to handle mixed data types. Another challenge arises when the minority class is sparse, as the synthetic instances generated may not accurately reflect the true underlying distribution, potentially introducing noise into the dataset.

Synthetic Minority Oversampling Technique for Regression (SmoteR) [35] is an extension of the SMOTE method, specifically designed to address imbalanced datasets in regression tasks where rare instancessuch as extreme valuesare underrepresented. SmoteR tackles this issue by generating synthetic samples for these rare cases, ensuring that the model can better learn from the full spectrum of target values. The methodology of SmoteR closely follows the principles of

SMOTE but is adapted to handle continuous target variables.

The process begins by identifying rare cases using a relevance function, which assigns scores to instances based on their target values. Instances with target values that deviate significantly from the majority are flagged as "rare" or "highly relevant." Once these rare cases are identified, the dataset is partitioned into two groups: normal instances, which have target values close to the majority, and highly relevant instances, which represent the rare or extreme cases. To reduce the dominance of the majority (normal) instances, SmoteR performs random undersampling on this group, ensuring that the model is not overwhelmed by frequent cases. For the highly relevant instances, SmoteR applies a modified version of SMOTE. Two rare instances are selected, and a synthetic instance is generated by interpolating between them. The target value of the synthetic instance is calculated as a weighted average of the target values of the two selected examples. This process of undersampling normal cases and oversampling rare cases helps balance the dataset, ensuring that both frequent and rare instances are adequately represented during training. One of the advantages of SmoteR is its ability to handle continuous target variables, making it a valuable tool for regression tasks [42]. By generating synthetic samples for rare instances, SmoteR enables models to better capture the relationships between extreme target values and their associated features, improving predictive accuracy for these cases. However, SmoteR is not without its limitations. The effectiveness of the method heavily depends on the proper design of the relevance function, which may vary depending on the specific problem and domain. Additionally, like SMOTE, SmoteR introduces synthetic examples, which can increase the risk of overfitting, particularly when the minority instances are sparse, or the dataset is small.

SMOGN-a Pre-processing Approach for Imbalanced Regression Problems using SMOTE and Gaussian Noise, proposed by Branco et al. [43] as a powerful pre-processing tool, is an extension of the SMOTE specifically designed for regression tasks. It addresses the challenge of imbalanced datasets by generating synthetic instances with added Gaussian noise. The methodology of SMOGN begins by identifying rare instances using a relevance function, which classifies instances as "rare" or "frequent" based on their target values. Once these rare instances are identified, SMOGN applies the principles of SMOTE to generate synthetic examples by interpolating between two rare instances. To introduce variability and prevent overfitting, Gaussian noise is added to both the features and the target values of the synthetic instances. Additionally, frequent instancesthose with common target valuesare undersampled to reduce their dominance in the dataset, ensuring that the model is not biased toward the more prevalent

target values. One of the key advantages of SMOGN is its ability to prevent overfitting. By adding Gaussian noise to the synthetic samples, SMOGN ensures that the generated instances are diverse and do not simply replicate existing data points. This variability helps the model generalize better to unseen data.

Camacho et al. [44] propose an extension of the Geometric Synthetic Minority Oversampling Technique (G-SMOTE) [45] for regression tasks, termed *Geometric SMOTE for Regression (G-SMOTE-R)*. As the name indicates, G-SMOTE-R addresses this issue by creating synthetic samples in the rare regions of the target variable, helping to balance the dataset and enhancing the model's capability to predict rare values. The core of G-SMOTE-R lies in the use of a *relevance function*, which maps the continuous target values to the interval $[0, 1]$. The relevance function is constructed using an automatic method based on the *box-and-whisker plot* [34] of the target variable. A user-defined relevance threshold is applied to separate rare examples (with relevance above the threshold) from normal examples (with relevance below the threshold). G-SMOTE-R generates synthetic samples for the rare regions using a geometric approach that extends the original G-SMOTE algorithm for classification. The process begins with the identification of rare examples in the dataset based on the relevance function and the user-defined threshold. For each rare example, the algorithm selects its nearest neighbours in the feature space. These neighbours can be either rare or normal examples, depending on the chosen strategy (minority, majority, or combined). Synthetic samples are then generated within a geometric region around the rare example rather than along the line segment between two examples (as in traditional SMOTE). This region is defined by a truncated and deformed hypersphere based on two hyperparameters: the truncation factor and the deformation factor. The truncation factor controls the level of truncation of the hypersphere, while the deformation factor determines the degree of shrinkage of the hyperspherical surface. The combination of these transformations results in a safe geometric region where synthetic samples can be generated. The target value for the synthetic sample is assigned as a weighted average of the target values of the rare example and its neighbours. The weights are determined by the Euclidean distances between the synthetic sample and the original examples, ensuring that the synthetic sample's target value is closer to the target value of the nearest original example.

Camacho and Bacao [46] propose *WSMOTER* (Weighting SMOTE for Regression), an approach that adapts the SMOTE algorithm for regression tasks. Unlike the methods that rely on arbitrary thresholds to partition the target domain into rare and common regions, WSMOTER treats each data point individually, avoiding discretization and making it more

aligned with the continuous nature of regression problems. WSMOTER operates by assigning weights to data points based on their rarity, measured using *DenseWeight*, which is a kernel density estimation (KDE)-based method. These weights determine the probability of selecting a data point as a seed for generating synthetic instances. The selection process considers both the feature space and the target value neighbourhood, ensuring that rare data points are more likely to be oversampled. The target value of the synthetic instance is determined by a weighted average of the target values of the seed points based on their Euclidean distances. This approach allows WSMOTER to generate synthetic instances that better represent rare regions of the target domain, improving model performance for underrepresented cases in imbalanced datasets.

Song et al. [47] propose *DistSMOGN*, a distributed resampling framework designed to address imbalanced regression problems in big data environments. The method extends the SMOGN algorithm to operate on large-scale datasets using Apache Spark and the MapReduce paradigm. Key steps include partitioning the dataset via *Scalable KMeans++* to maintain spatial coherence among rare cases, enabling parallel processing of clusters. Within each partition, a distance matrix is computed for nearest neighbour searches, and synthetic samples are generated by interpolating rare instances with *SmoteR* or injecting Gaussian Noise, contingent on the proximity between base and neighbour instances. The algorithm ensures localized resampling within clusters to preserve data integrity, followed by aggregation of results across partitions to produce a balanced dataset. This distributed approach optimizes computational efficiency while maintaining the quality of synthetic data generation, scaling to efficiently handle high-volume imbalanced regression tasks.

2.4.2.2 Alternative Pre-Processing Methods

Branco et al. [36] propose several pre-processing approaches for handling imbalanced distributions in regression tasks, focusing on generating synthetic data and undersampling frequent cases to improve model performance on rare target values. One such method is the *Random Oversampling and Gaussian Noise* (ROGN) which addresses imbalanced regression problems by combining random oversampling of rare cases with the injection of Gaussian noise. The method begins by using a relevance function $\phi()$ to identify rare instances in the dataset. The relevance function assigns higher scores to instances with extreme target values (either very high or very low), which are considered rare. Once these rare instances are identified, ROGN applies random oversampling by replicating them to increase their representation in the dataset. To prevent

overfitting and introduce variability, Gaussian noise is added to both the features and the target values of the replicated instances. The noise is sampled from a normal distribution with a mean of zero and a small standard deviation, ensuring that the synthetic instances remain close to the original data distribution. This noise injection step is critical, as it prevents the model from memorizing the replicated instances and encourages generalization to unseen data.

In addition to oversampling rare cases, ROGN applies random undersampling to frequent instances to reduce their dominance in the dataset. The combination of oversampling with noise injection and undersampling makes ROGN particularly effective for handling imbalanced regression tasks.

One of the key strengths of ROGN is its simplicity and computational efficiency. Unlike more complex methods that rely on interpolation or weighted sampling, ROGN uses straightforward random replication and noise injection, making it easy to implement and tune. However, the method's performance heavily depends on the proper design of the relevance function and the choice of noise parameters. If the noise level is too high, the synthetic instances may deviate significantly from the original data distribution, leading to poor model performance. Conversely, the method may fail to prevent overfitting if the noise level is too low.

Another approach introduced by the authors is the *Weighted Relevance-based Combination Strategy*(WERCS), a pre-processing method designed to address imbalanced regression problems by combining oversampling and undersampling based on instance relevance scores. The method begins by using a relevance function $\phi()$ to assign a relevance score to each instance in the dataset. Instances with extreme target values (very high or very low) are assigned higher relevance scores, as they are considered rare. In comparison, instances with target values close to the majority are assigned lower scores. Once the relevance scores are computed, WERCS assigns weights to each instance based on these scores. Rare instances receive higher weights, making them more likely to be selected for oversampling, while frequent instances receive lower weights, making them candidates for undersampling. The oversampling process in WERCS involves replicating rare instances to increase their representation. Unlike methods that rely on interpolation or noise injection, WERCS uses direct replication, which preserves the original data distribution of the rare cases. On the other hand, the undersampling process randomly removes frequent instances based on their weights, ensuring that the dataset remains balanced without losing critical information. Flexibility is the main advantage of WERCS. By using a weighted combination of oversampling and undersampling, the method can adapt to different levels of

imbalance in the dataset. Additionally, WERCS does not require complex parameter tuning, as the weights are directly derived from the relevance scores. However, the method's performance heavily depends on the proper design of the relevance function. If the relevance function fails to accurately identify rare instances, the oversampling and undersampling processes may not effectively balance the dataset.

2.4.3 Algorithm-level Strategies

Algorithm-level techniques also referred to as cost-sensitive learning, modify the learning process to improve performance on imbalanced datasets. This is achieved by adjusting the loss function to assign higher penalties to errors on rare instances, ensuring that the model prioritizes these cases during training. Typically, this involves weighting the training samples so that underrepresented instances contribute more significantly to the learning process, reducing the dominance of frequent cases [48].

To tackle the challenge of learning from imbalanced datasets in regression tasks, Steininger et al. [33] propose two approaches: *DenseWeight*, a sample weighting scheme, and *DenseLoss*, a cost-sensitive learning method for neural network regression. *DenseWeight* addresses the imbalance problem by assigning higher weights to data points with rare target values to increase their influence during model training. The rarity of a target value is estimated using kernel density estimation (KDE), which approximates the probability density function of the target variable. By normalizing the density values and applying a weighting function, *DenseWeight* ensures that rare target values receive proportionally larger weights. This weighting function is controlled by a hyperparameter α, which allows users to adjust the degree of emphasis on rare cases. For example, setting $\alpha = 0$ results in uniform weighting (no emphasis on rarity), while larger values of α increasingly prioritize rare data points. This flexibility enables users to tailor the model's focus based on the specific requirements of their application. *DenseLoss* builds on *DenseWeight* by integrating these sample weights into the loss function used during neural network training. Specifically, the loss contribution of each data point is scaled by its corresponding weight, ensuring that rare cases have a stronger influence on the gradient updates. This cost-sensitive approach encourages the model to improve its predictions for underrepresented regions of the target distribution without requiring modifications to the dataset itself. Unlike sampling-based methods, such as SMOTER or SMOGN, which create synthetic samples or remove data points to balance the dataset, *DenseLoss* operates at the

algorithm level. This avoids potential issues such as overfitting from oversampling or loss of information from undersampling.

Torgo and Ribeiro [49] propose a specialized regression tree algorithm designed to predict rare extreme values (outliers) of a continuous target variable while maintaining model interpretability. The method introduces a splitting criterion that replaces the standard least squares error minimization with an F-measure-based optimization, prioritizing outlier detection and accurate prediction. Outliers are defined using the statistical box plot method, where values exceeding the upper adjacent value (adj_H) or falling below the lower adjacent value (adj_L)calculated as $1.5\times$ the inter-quartile rangeare considered extreme. The F-measure combines precision and recall metrics adapted for regression: precision accounts for the normalized mean squared error ($NMSE_O$) of outliers, while recall measures the proportion of correctly identified outliers. The splitting criterion selects splits that maximize the F-measure in either branch of the tree, favouring partitions that isolate subsets with high outlier relevance. Tree growth terminates when a node achieves a user-defined F-measure threshold or contains no outliers, avoiding traditional pruning methods that rely on statistical significance.

The authors in [50] propose two methods for addressing Deep Imbalanced Regression (DIR): *Label Distribution Smoothing (LDS)* and *Feature Distribution Smoothing (FDS)*. LDS employs kernel density estimation to smooth the empirical label distribution, accounting for the continuity of targets by leveraging similarities between nearby labels. This produces an effective label density that better reflects the underlying data imbalance in regression tasks. FDS extends this concept to the feature space by smoothing feature statistics (mean and covariance) across neighbouring target bins using a symmetric kernel, thereby calibrating biased feature representations caused by data imbalance. Both methods exploit the inherent continuity in regression targets, enabling techniques like re-weighting or synthetic sampling to be adapted effectively.

2.4.4 Hybrid Strategies

Hybrid techniques integrate both data-level and algorithm-level approaches to enhance learning from imbalanced datasets. These methods leverage data manipulation strategies while incorporating algorithmic modifications to improve model performance on underrepresented instances. One widely adopted hybrid approach is ensemble learning [46], which improves predictive accuracy by training multiple models and combining their outputs. A well-known example is boosting, a sequential learning process where each model is trained to correct the

errors made by its predecessor. Among boosting methods, AdaBoost [51] is considered one of the most influential due to its ability to adjust instance weights, focusing more on difficult cases iteratively. However, ensemble learning alone is not sufficient to handle imbalanced data effectively. Its success depends on integrating additional techniques, such as data-level resampling or cost-sensitive learning, to ensure better representation of rare instances in the training process [52].

Orhobor et al. [53] tackle the problem of imbalanced regression by proposing an ensemble-based approach (*Imbalanced regression using regressor-classifier ensembles*) that leverages the distribution of the target variable to improve predictions for rare or extreme values. The core idea of the proposed method is to discretize the continuous target variable into multiple bins using a discretization technique (e.g., frequency-based or clustering-based). For each bin, a regressor is trained to predict the target values within that specific range, while a classifier is trained to estimate the probability that a new sample belongs to that bin. This dual approach allows the model to focus on different regions of the target distribution independently, with the classifiers acting as a mechanism to weigh the predictions of the regressors based on the rarity of the target values. Specifically, the classifiers provide a probabilistic weighting scheme that dynamically adjusts the influence of each regressor's prediction during inference. This ensures that rare or extreme values receive more attention during model training and prediction. The method operates in two phases: a *building phase* and a *prediction phase*. In the building phase, the target variable is discretized into multiple bins, and for each bin, a regressor and a classifier are trained. The classifiers are used to predict the likelihood that a new sample belongs to each bin, while the regressors provide predictions for the target values within their respective bins. During the prediction phase, the predictions from the regressors are combined using the probabilities provided by the classifiers, resulting in a weighted prediction that accounts for the rarity of the target values. The authors note that the computational cost of the method can be high, especially when using large bin sizes or complex models. Despite this, the method's ability to dynamically adjust its focus based on the rarity of the target values makes it a powerful tool for addressing imbalanced regression problems.

The method proposed by [54], known as *REsampled BAGGing* (REBAGG), combines data pre-processing strategies with bagging-based ensemble techniques to enhance predictive performance for rare instances. REBAGG also uses the relevance function defined through an automated process based on *box-and-whisker plot* [34] of the target variable with a user-defined relevance threshold to distinguish rare and frequent instances. REBAGG generates synthetic

samples for rare regions through a resampling methodology that merges bagging with data pre-processing strategies. The algorithm operates in two principal phases: initially, it constructs a set of models utilizing pre-processed samples from the training set, and subsequently, it aggregates the predictions from these models to derive the final output. The resampling techniques incorporated in REBAGG encompass four primary categories: balance, balance.SMT, variation, and variation.SMT. The *balance* methods ensure that the new training set maintains an equal representation of rare and normal cases, whereas the *variation* methods permit a fluctuating ratio of rare to normal cases, thereby introducing additional diversity into the training sets. The suffix *SMT* signifies that the SMOTER algorithm is employed to create synthetic rare cases, while methods lacking this suffix depend on exact replicas of existing rare cases. This strategy guarantees that the synthetic samples are both diverse and representative of the rare regions, thereby mitigating the risk of overfitting.

Moniz et al. [37] propose an adaptation of the SMOTEBoost approach which is used to direct the distribution of data towards rare cases. SMOTEBoost addresses imbalance learning in regression by combining boosting algorithms with SMOTE. SMOTEBoost generates synthetic samples for the rare regions using a resampling approach that combines boosting with the SMOTE algorithm. The method works by iteratively applying SMOTE to generate synthetic examples of highly relevant casesthe relevance is calculated based on the *box-and-whisker plot* [34]which are then used to train weak learners in a boosting framework. At each iteration, the boosting algorithm identifies cases that are difficult to predict, increasing their weights to emphasize their importance in the subsequent training steps. SMOTE is then applied to create additional synthetic examples of these highly relevant cases, balancing the distribution and improving the representation of extreme target values.

Huang et al. [55] propose a method for addressing imbalanced regression by integrating density-based rare instance identification with a hybrid resampling strategy. Rare instances are detected through a relevance function derived from the inverse of kernel density estimation (KDE) applied to the target values. The KDE models the data distribution, and its inverse emphasizes low-density regions corresponding to rare instances. This measure is standardized via z-score normalization to compute the relevance function $\phi()$. Instances with $\phi()$ exceeding a predefined threshold (typically set to retain approximately 10% of the data as rare) are classified as rare, while others are labeled normal. The resampling strategy combines two components. First, a boosting-weighted undersampling technique iteratively removes normal instances, prioritizing those with low relevance scores. The removal process is guided by validation

performance: batches of normal samples are discarded only if their exclusion improves or maintains the mean squared error (MSE) on a validation set, ensuring informative samples are retained. Second, synthetic rare instances are generated using a conditional variational autoencoder (CVAE) trained on categorized data (rare low, rare high, normal). The CVAE leverages target-specific structural information to produce plausible synthetic samples, while their target values are predicted using an ensemble model comprising XGBoost, SVR, and Random Forest. A boosting mechanism further refines the synthetic data by discarding samples that fail to improve validation performance.

2.5 Research Gaps

Existing methodologies for imbalanced regression predominantly assume static datasets where global statistics and full data access enable precise rarity estimation. However, these assumptions collapse in online regression scenarios, where data arrives incrementally and may alter target distributions dynamically. Data-level strategies like SMOTE-based oversampling require complete dataset access to synthesize rare instances. At the same time, density-based approaches depend on kernel density estimationa process infeasible under single-pass streaming constraints. Algorithm-level adaptations, such as cost-sensitive learning, face scalability barriers due to their dependency on batch updates for loss function recalibration. Furthermore, relevance functions that rely on boxplot-derived thresholds or fixed distributional assumptions struggle to adapt to evolving target variable characteristics, as statistical moments and outlier boundaries may shift unpredictably in data streams. These limitations underscore a critical gap: the absence of mechanisms to dynamically identify rare instances and adjust sampling strategies in real time without storing historical data.

Chapter 3

Chebyshev Approaches for Imbalanced Data Streams Regression

This chapter introduces a data-level approach to managing imbalanced regression in data streams by leveraging Chebyshev's inequality. The method focuses on adjusting the sampling of data streams, either undersampling or oversampling, guided by Chebyshev's probability, to represent better rare and extreme values, which are crucial to prediction accuracy. Through applying this technique across various regression algorithms, the study highlights improvements in handling imbalanced data streams, mainly when rare instances significantly influence the outcome.

The chapter is organized as follows. Section 3.1 introduces Chebyshev's inequality and its role in identifying rare cases of imbalanced data streams. Section 3.2 presents the proposed methods, ChebyUS and ChebyOS, describing their implementation and theoretical foundation. Section 3.3 details the experimental evaluation, including the datasets, experimental setup, and performance metrics. Finally, Section 3.4 discusses the limitations of the proposed approaches.

3.1 Chebyshev's Inequality

We propose two methods for learning regression models from imbalanced data streams. Both methods use Chebyshev's inequality to train learning models over a relatively balanced data stream once the incoming data stream is imbalanced. The mentioned inequality derived from

Markov inequality is used to bind the tail probabilities of a random variable like Y. It guarantees that in any probability distribution, 'nearly all' values are close to the mean. More precisely, no more than $\frac{1}{t^2}$ of the distribution's values can be more than t times standard deviations away from the mean. Although conservative, the inequality can be applied to completely arbitrary distributions (unknown except for mean and variance). Let Y be a random variable with finite expected value $\overline{y}$ and finite non-zero variance σ^2. Then for any real number $t > 0$, we have:

$$\Pr(|y - \overline{y}| \geq t\sigma) \leq \frac{1}{t^2} \tag{3.1}$$

Only the case $t > 1$ is useful in the above inequality. In cases $t < 1$, the right-hand side is greater than one, and thus, the statement will be "always true" as the probability of any event cannot be greater than one. Another "always true" case of inequality is when $t = 1$. In this case, the inequality changes to a statement saying that the probability of something is less than or equal to one, which is "always true".

For $t = \frac{|y-\overline{y}|}{\sigma}$ and $t > 1$, we define *frequency* score of the observation $\langle x, y \rangle$ as:

$$P(\mid \overline{y} - y \mid \geq t) = \frac{1}{\left(\frac{|y-\overline{y}|}{\sigma}\right)^2} \tag{3.2}$$

The above definition states that the probability of observing y far from its mean is small and decreases as we get farther away from the mean. In an imbalanced data stream regression scenario, considering the mean of target values of the examples in the data stream ($\overline{y}$), examples with rare extreme target values are clearly more likely to occur far from the mean. In contrast, examples with frequent target values are closer to the mean. So, given the mean and variance of a random variable, Chebyshev's inequality can indicate the degree of the rarity of an observation, such that its low and high values imply that the observation is most probably a rare or frequent case, respectively.

Figure 3.1 shows the probability values calculated from Equation 3.2 for Fried data set* along with the box plot of the target variable.

As can be seen from the figures and as we expected, Chebyshev's probability value for examples near the mean is close to one. It decreases as we get far from the mean until it gets

*https://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html

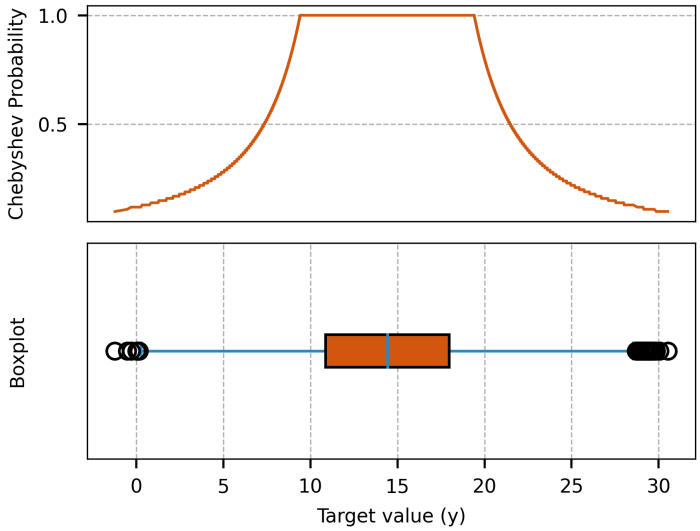


FIGURE 3.1: Box plot and Chebyshevs probability for the target value of Fried data set.

close to zero, for example, at the farthest distance to the mean. Accordingly, interpretation of the output value of Equation 3.2 for an example as its frequency score makes sense. Moreover, it meets the imbalance regression problem definition, w.r.t. rare extreme values of the target variable.

Having been equipped with the heuristic to discover if an example is rare or frequent, the next step is to utilize such knowledge in training a regression model. To do that, ChebyUS and ChebyOS are the two methods proposed in this chapter. They are described in detail in the next section.

3.2 Proposed Methods

Building on Chebyshev's inequality as a statistical foundation for rarity estimation in data streams, this section introduces two complementary sampling strategies designed to address imbalanced regression in evolving environments. The first method, *Chebyshev-based Under-sampling* (ChebyUS), strategically reduces the influence of frequent cases while preserving critical rare instances through probabilistic instance selection. Conversely, *Chebyshev-based Oversampling* (ChebyOS) amplifies rare cases by dynamically adjusting replication rates based on their statistical extremity. Both methods employ incremental computation of distributional statistics (mean μ and variance σ^2) to maintain real-time adaptability, ensuring compatibility

with streaming data constraints. The subsequent subsections formalize these approaches through algorithmic implementations.

3.2.1 ChebyUS: Chebyshev-based Undersampling

The proposed undersampling method is presented in Algorithm 3.1. This algorithm selects an incoming example for training the model if a randomly generated number in $[0, 1]$ is greater or equal to its Chebyshev's probability, which is calculated as:

$$P(\mid y - \overline{y} \mid \geq t) = \begin{cases} \frac{\sigma^2}{\mid y - \overline{y} \mid^2}, & t > 1 \\ 1, & t \leq 1 \end{cases} \tag{3.3}$$

If the example is not selected, it is assumed to be a frequent case. Still, it receives a second chance to be selected if the number of frequent cases selected so far is less than that of rare cases. If so, the example is selected with a second chance probability (input parameter sp).

Algorithm 3.1 ChebyUS: Chebyshev-based UnderSampling Algorithm for Data Streams

procedure CHEBYUS($\mathcal{S}$: Data stream, sp: second chance probability)
  $H \leftarrow CreateEmptyModel()$
  $i \leftarrow 0$
  $RCounter \leftarrow 0$ ▷ rare cases counter
  $FCounter \leftarrow 0$ ▷ frequent cases counter
  $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$ ▷ Get next example from data stream
  while $\langle \mathrm{x}, y \rangle \neq null$ **do**
    $\overline{y}, \sigma \leftarrow UpdateStatistics(y)$
    $p \leftarrow ComputeProbability(y, \overline{y}, \sigma)$ ▷ Equation 3.3
    $r \leftarrow RandomNumber()$
    if $r \geq p$ **then**
      $H \leftarrow TrainModel(H, \langle \mathrm{x}, y \rangle)$
      $RCounter \leftarrow RCounter + 1$
    else
      if $FCounter \leq RCounter$ **and** $r \leq sp$ **then**
        $H \leftarrow TrainModel(H, \langle \mathrm{x}, y \rangle)$
        $FCounter \leftarrow FCounter + 1$
      end if
    end if
    $i \leftarrow i + 1$
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$ ▷ Get next example from data stream
  end while
end procedure

The descriptive statistics (μ and σ^2) of the target variable of examples can be computed through incremental methods [56]. Particularly, they can be estimated for a data set including n examples if we have the mean and the variance values for $n-1$ examples as below:

$$\mu_n = \mu_{n-1} + \frac{1}{n}(x_n - \mu_{n-1}) \tag{3.4}$$

$$\sigma_n^2 = \sigma_{n-1}^2 + (x_n - \mu_{n-1})(x_n - \mu_n) \tag{3.5}$$

where μ and σ^2 indicate the mean and variance respectively.

The greater n we have, the more accurate the estimation will be. For the first examples, mean and variance are not accurate, and therefore, Chebyshev's probability will not be accurate enough. However, as more examples are received, those statistics (i.e. mean and variance) and, consequently, Chebyshev's probability are getting more stable and accurate.

At the end of the algorithm, which is the end of the model's training phase, we expect the model to have been trained over approximately the same portion of frequent and rare cases.

3.2.2 ChebyOS: Chebyshev-based Oversampling

Another way of making a balanced data stream is to oversample rare cases of the incoming imbalanced data stream. Since those rare cases in data streams can be discovered by their Chebyshev's probability, they can be easily oversampled by replication. Algorithm 3.2 describes our oversampling proposed method.

For each example, a t value can be calculated by Equation 3.6 which yields the result in $[0 \quad +\infty)$.

$$t = \frac{|y - \overline{y}|}{\sigma} \tag{3.6}$$

While the t value is small for examples near the mean, it would be larger for ones farther from the mean and has its largest value for examples located in the farthest distance to the mean (i.e. extreme values). We limit the function in Equation 3.6 to produce only natural numbers as follows:

Algorithm 3.2 ChebyOS: Chebyshev-based Oversampling Algorithm for Data Streams

```
1: procedure CHEBYOS(S: Data stream)
2:     H ← CreateEmptyModel()
3:     i ← 0
4:     ⟨x, y⟩ ← GetExample(S)                    ▷ Get next example from data stream
5:     while ⟨x, y⟩ ≠ null do
6:         ȳ, σ ← UpdateStatistics(y)
7:         k ← ComputeK(y, ȳ, σ)                  ▷ Equation 3.7
8:         for i ← 1 to k do
9:             H ← TrainModel(H, ⟨x, y⟩)
10:        end for
11:        i ← i + 1
12:        ⟨x, y⟩ ← GetExample(S)                 ▷ Get next example from data stream
13:    end while
14: end procedure
```

$$K = \left\lceil \frac{|y - \overline{y}|}{\sigma} \right\rceil \tag{3.7}$$

K is expected to have greater numbers for rare cases. In our proposed oversampling method, we use the K value computed for each incoming example and present that example exactly K times to the learner.

Examples not as far from the mean as the variance are most probably frequent cases. They contribute only once in the learner's training process, while the others contribute more.

3.3 Experimental Evaluation

Our experimental setting aims to answer the following fundamental research questions:

1. What is the impact of using a relevance-weighted error function compared to a standard error function?
2. How does selecting rare cases based on a threshold on the relevance function affect the performance of the proposed methods?
3. How do different types of extreme values considered for rare cases impact the performance of the proposed methods?
4. In general, are the two proposed methods effective to cope with imbalanced regression problems?

3.3.1 Relevance Function

To express non-uniform importance associated with different values of the target variable, Torgo and Ribeiro [57] have proposed a relevance function $\phi\colon \mathcal{Y} \rightarrow [0,1]$ that maps the continuous target variable domain into a scale of importance. This continuous function expresses the domain-specific bias, where 0 and 1 represent the minimum and maximum relevance, respectively.

This function allows the specification of regions of interest in the target variable. Given the infinite nature of the target variable domain, it performs a Piecewise Cubic Hermite Interpolation over a set of control points. This interpolation method is shape-preserving and, contrary to other interpolation methods (e.g. splines), is not prone to unrealistic oscillations around the control points. In fact, it ensures that the resulting interpolation is bounded by the provided points [10]. Suppose there is no domain knowledge to specify such control points. In that case, the authors propose an automatic way to infer the control points that rely on the target variable distribution based on the boxplot statistics. This automatic method assumes that the most relevant values are in the tail(s) of the target variable distribution. The control points inferred by the boxplot are the median, the lower, and the upper adjacent values. Regardless of the method employed to define the control points, the proposed interpolation method performs linear extrapolation beyond the maximum and the minimum values supplied in the set of control points, thus guaranteeing the relevance function is constant. This means that, in the case of the boxplot automatic method, the tails of the target variable distribution will have maximum relevance outside the lower and the upper adjacent values indicated by the boxplot.

This method meets the requirements of many real-world applications where the rare extreme values (i.e. rare values, which are located at probability tails of the data) are the most important values on which accurate predictions should be obtained (e.g. the prediction of pollution concentration values). The resulting function is a continuous function obtained through a smooth interpolation of the set of control points and that maps rare and extreme values (i.e. outliers according to the boxplot) with the maximum relevance 1 and the median with the minimum relevance 0. Additionally, this method also allows the user to specify the region of interest: high, low, or extreme values. Interested readers are referred to Ribeiro and Moniz [10] for further details on the $\phi()$ function.

3.3.2 Data sets

We performed our experiments over fourteen benchmark data sets: twelve real data sets along with two artificial collected from several repositories: UCI [58], KEEL* [59], Kaggle† and a regression data sets repository‡.

Table 3.1 shows, for each data set, the number of the examples and the number and percentage of the rare cases identified by using a threshold thr_ϕ on $\phi()$ relevance function. This means that the number of rare cases is obtained for each data set D by the number of cases in the subset $D_R = \left\{ \langle x_i, y_i \rangle \in D \mid \phi(y_i) \geq thr_\phi \right\}$. If we consider $thr_\phi = 0$ then we have that $D = D_R$. As we increase the value of thr_ϕ, approaching the maximum relevance value of 1, the cardinality of D_R will be smaller. Ultimately, and by the definition of the relevance function $\phi()$, it is implied that $|D \setminus D_R| >> |D_R|$.

In Table 3.1 we used $thr_\phi = 0.8$. The same information but with $thr_\phi = 0.9$ and $thr_\phi = 1$ can be found in Tables A.2 and A.3 in Appendix, respectively. Also, in the Appendix, we include the name, number of the features and the source of each data set in Table A.1. To pave the way for a deeper analysis of the total number, we have also reported the number and percentage of rare cases, considering only the target variable's low or high extreme values.

In Appendix A, Figures A.4 - A.17 illustrate the relevance values assigned to all examples of the data sets mentioned in Table 3.1 aligned with their box plot and their Chebyshev's values. The figures show that weights are small for cases with target values around the median target variable. Furthermore, the weights smoothly increase as we get farther from the median until they reach their maximum values over the cases with rare extreme target values. Different sets of the rare cases for the puma32H data set are illustrated in Figure 3.2 in which $\phi()$ threshold (thr_ϕ) has been set to 0.8, meaning that examples with relevance value equal or greater than 0.8 are considered as rare cases.

3.3.3 Experimental Setup

No pre-processing step has been done over the data set, and all features of the data sets feed the learner models by their original form and value. The proposed methods have been implemented in Kotlin 1.4.31 and Java 8 with the help of MOA – Massive Online Analysis [60],

*https://sci2s.ugr.es/keel/category.php?cat=reg
†https://www.kaggle.com/tsaustin/us-used-car-sales-data
‡https://www.dcc.fc.up.pt/ltorgo/Regression/DataSets.html

TABLE 3.1: Data sets characteristics: number of cases (#cases) and number of rare cases (#rare cases) with $thr_\phi = 0.8$ that are low extremes (i.e., below the median), high extremes (i.e., above the median), or both.

Data set	#cases	#rare cases ($thr_\phi = 0.8$)		
		Low extremes	High extremes	Both extremes
puma32H	8,192	422 (5.15%)	435 (5.31%)	857 (10.46%)
cpusum	8,192	722 (8.82%)	849 (10.36%)	1,571 (19.18%)
elevator	16,599	169 (1.02%)	2,266 (13.65%)	2,435 (14.67%)
bike	17,389	4,433 (25.49%)	1,301 (7.48%)	5,734 (32.97%)
energy	19,735	352 (1.78%)	2,665 (13.50%)	3,017 (15.28%)
calhousing	20,640	816 (3.95%)	1,821 (8.82%)	2,637 (12.77%)
gasemission	36,733	199 (0.54%)	1,967 (5.36%)	2,166 (5.90%)
mv	40,768	1,414 (3.47%)	9,037 (22.17%)	10,451 (25.64%)
fried	40,768	739 (1.81%)	747 (1.83%)	1,486 (3.64%)
pollution	41,757	7,462 (17.86%)	3,663 (8.77%)	11,125 (26.63%)
car price	122,144	19,205 (15.72%)	14,156 (11.59%)	33,361 (27.31%)
query	199,843	5,616 (2.81%)	3,115 (1.56%)	8,731 (4.37%)
GPU	241,600	18,033 (7.46%)	36,493 (15.10%)	54,526 (22.56%)
3d	434,873	30 (0.01%)	30,718 (7.06%)	30,748 (7.07%)

open-source software for mining and analyzing large data sets in a stream-like manner and RxJava–Reactive Extensions for the JVM [61] libraries. Regarding the $\phi()$ function, we have used the implementation available in the R package UBL. For error computation and analysis of the results, we used R programming language (R version 4.4.2 (2024-10-31 ucrt) – "Pile of Leaves"). For reproducibility purposes, the code is publicly available in a GitHub repository *. All experiments were conducted on a 2.6-GHz core i7 PC with 8 GB of RAM.

As the evaluation metric, we have used both the standard $RMSE$ and the $RMSE_\phi$ - $\phi()$ weighted version of this function where the prediction error for each example is multiplied by the relevance value assigned to that example, as follows:

$$RMSE_\phi = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\phi(y_i) \times (y_i - \hat{y}_i)^2} \tag{3.8}$$

where y_i, $\hat{y}_i$ and $\phi(y_i)$ refer to the true value, the predicted value and the relevance value for i-th example in the data set, respectively.

Also, to alleviate the impact of algorithm-dependent biases, which may distort the results, we have used four regression models as the base learner models for evaluating our two proposed

*https://github.com/ehaminian/Learning-Strategies-for-Imbalanced-Regression-Data-Streams

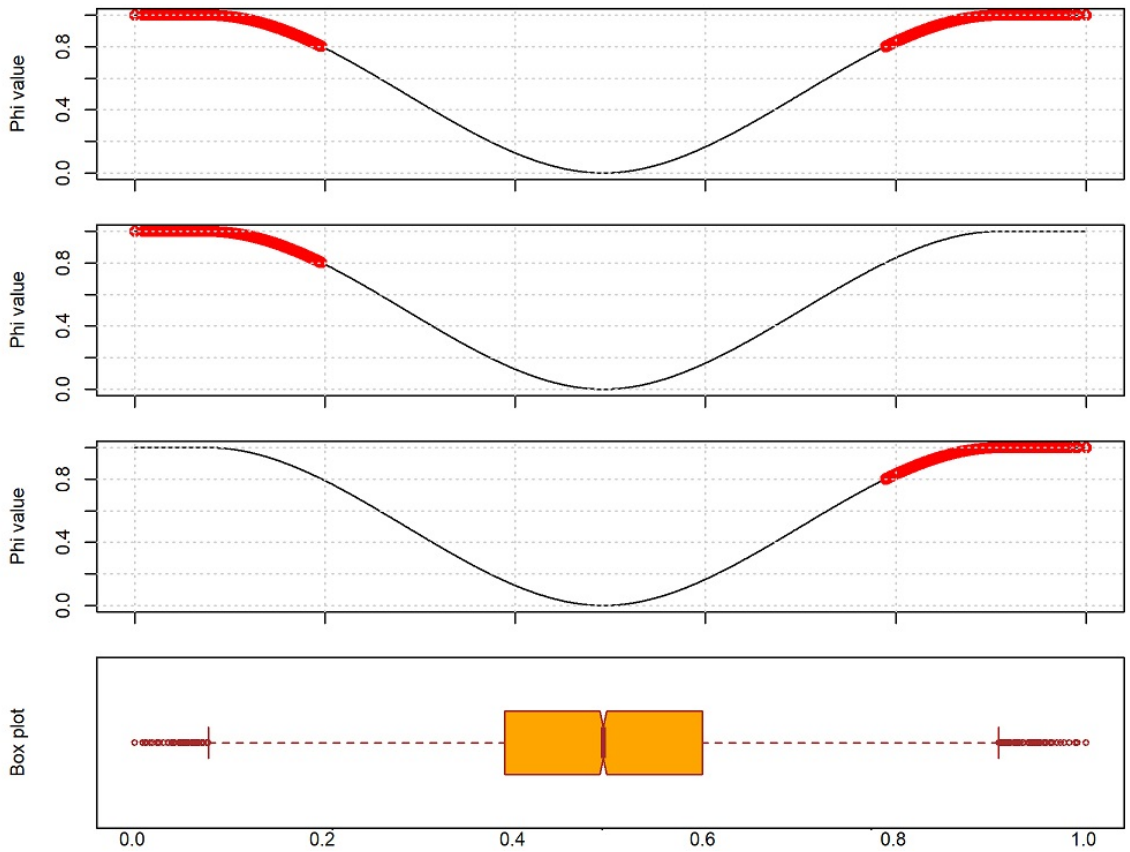


FIGURE 3.2: Rare extreme target values cases for puma32H data set identified by $\phi()$ relevance function [10] that uses the target variable boxplot distribution. Setting a relevance threshold $thr_\phi = 0.8$ it is possible to identify the cases that correspond to both extremes, low extremes (i.e. below the median) and high extremes (i.e. above the median).

sampling strategies: perceptron [62], fast incremental model trees with drift detection (FIMT-DD) [63], TargetMean (returns the mean of the target variable of the training examples) and AMRules [64].

These regression algorithms were applied to an imbalanced data stream where the goal is to learn models that are especially accurate at predicting rare extreme target values. In each experiment, three instances of each learner were created: two of them were trained using our sampling strategies and the other one - we call it Baseline - was trained with all examples of the data set. $RMSE$ and $RMSE_\phi$ were used to assess the performance of the models. Each experiment was run ten times, and the average and standard deviation of the obtained results were written down.

In all experiments, we have used a prequential evaluation procedure [8] in which each example has been used for testing and training. First, the model is evaluated on the instance, and then a single incremental learning step is carried out. Also, 80% of examples in each data

set have been considered for training and the remaining 20% for testing the models.

Regarding the threshold parameter (`thr`) that we have as the input parameter of our undersampling function in Algorithm 3.1, we set it to 0.15 in all experiments. This value was empirically tuned by the trial-and-error method. Higher values force the algorithm to pick up immediate incoming frequent cases once a rare case is observed. On the other hand, lower values of this threshold may result in training the model over much fewer frequent cases than the rare ones.

One of our goals is to explore the impact of the threshold selected for $\phi()$ function which is used to identify rare cases. We want to study the performance of the proposed methods over the examples with different type of extreme values (high, low or both) using different error functions (i.e. $RMSE$ and $RMSE_\phi$). For that purpose, we have conducted a series of experiments varying each of those parameters.

More specifically, we evaluate our *two* proposed methods over *three* different sets of rare cases, i.e. low (extreme target values below the median), high (extreme target values above the median) and both. To define the set of rare cases, we consider four different thresholds of relevance thr_ϕ: 0, 0.8, 0.9 and 1.

3.3.4 Experimental Results

This section provides an overview of the obtained results, with full details available in Tables A.4 to A.9 in the Appendix. To facilitate comparison, we present graphical summaries illustrating the number of wins and losses for each learner when using our sampling strategies against its corresponding baseline model. These graphs also highlight statistically significant wins and losses at a 95% confidence level, based on the Wilcoxon signed-rank test. Additionally, we include Critical Difference (CD) diagrams from the post-hoc Nemenyi test [65] to compare the relative performance of different learners across experimental conditions.

The results for predictions on both sides of extreme rare cases in the datasets are summarized in Tables 3.2 and 3.3, with their graphical representations provided in Figure 3.3. These tables report the performance of various algorithms under both sampling approaches, considering average rank and the number of significant wins. In these experiments, rare cases are identified using a threshold of $thr_\phi = 0.8$, and prediction errors are measured using $RMSE_\phi$. To address the first research question, we maintain all experimental conditions unchanged except for the

TABLE 3.2: Summary of results of Chebyshev-based Undersampling (ChebyUS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.8$) and $RMSE$.

Algorithm	Approach	$RMSE_\phi$		$RMSE$	
		avg_rank	#w	avg_rank	#w
AMRules	Baseline	6.32	(5)	6.14	(5)
	ChebyUS	**5.61**	(8)	**5.50**	(8)
Perceptron	Baseline	4.68	(3)	4.71	(3)
	ChebyUS	**2.25**	(9)	**2.50**	(10)
FIMT-DD	Baseline	3.68	(4)	3.50	(4)
	ChebyUS	**2.68**	(10)	**2.86**	(9)
TargetMean	Baseline	5.82	(4)	5.64	(5)
	ChebyUS	**4.96**	(8)	**5.14**	(8)

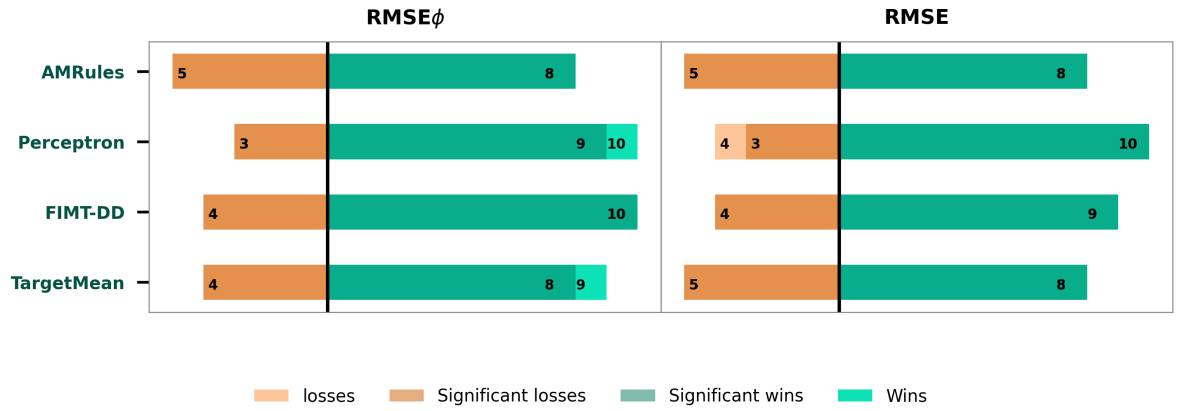
TABLE 3.3: Summary of results of Chebyshev-based Oversampling (ChebyOS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.8$) and $RMSE$.

Algorithm	Approach	$RMSE_\phi$		$RMSE$	
		avg_rank	#w	avg_rank	#w
AMRules	Baseline	6.57	(3)	6.54	(3)
	ChebyOS	**5.43**	(10)	**5.39**	(10)
Perceptron	Baseline	4.79	(3)	4.93	(3)
	ChebyOS	**3.00**	(10)	**3.04**	(11)
FIMT-DD	Baseline	3.64	(2)	3.61	(2)
	ChebyOS	**2.07**	(11)	**2.00**	(11)
TargetMean	Baseline	6.07	(3)	6.04	(3)
	ChebyOS	**4.43**	(10)	**4.46**	(10)

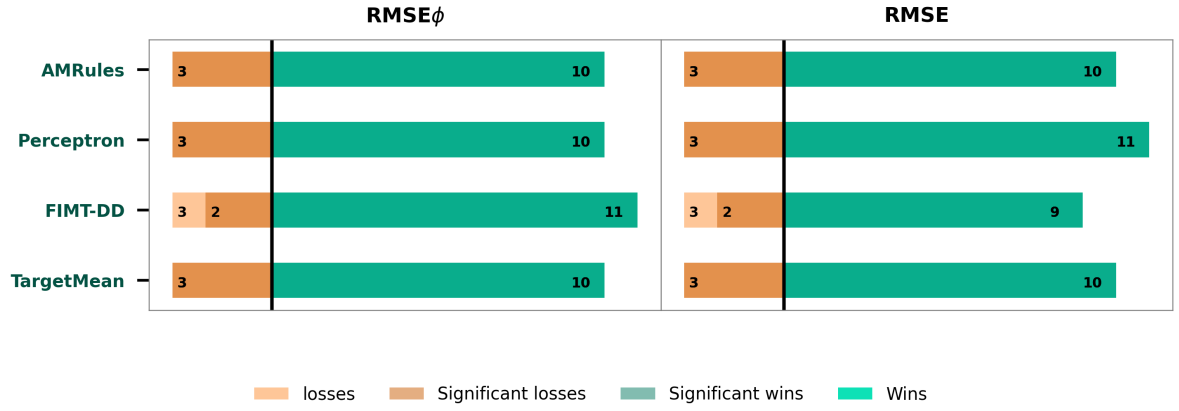
error function, replacing $RMSE_\phi$ with the standard $RMSE$. That is, rare cases are first identified based on the defined threshold, after which prediction errors are computed using $RMSE$.

From the $RMSE_\phi$ perspective, the results indicate that models trained with our proposed sampling strategies outperform their baseline counterparts in predicting rare cases. However, when using standard $RMSE$, an initial analysis suggests no significant difference compared to $RMSE_\phi$. This outcome is mainly influenced by the relevance values assigned to rare examples in the experiments. Specifically, when comparing $RMSE$ and $RMSE_\phi$, the former can be seen as a special case of the latter (Equation 3.8) where $\phi(y) = 1$ for all $y \in Y$. Since thr_ϕ is set to 0.8 in these experiments, all examples contributing to the error calculation have relevance values of at least 0.8, which are close to 1. As thr_ϕ increases toward 1, the differences between the two

error functions diminish, whereas as thr_ϕ decreases toward 0, the discrepancies between them become more pronounced.



(A) Baseline *versus* ChebyUS



(B) Baseline *versus* ChebyOS

FIGURE 3.3: Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from no sampling (Baseline) against Chebyshev-based sampling (ChebyUS and ChebyOS) using $RMSE_\phi$ ($thr_\phi = 0.8$) and $RMSE$.

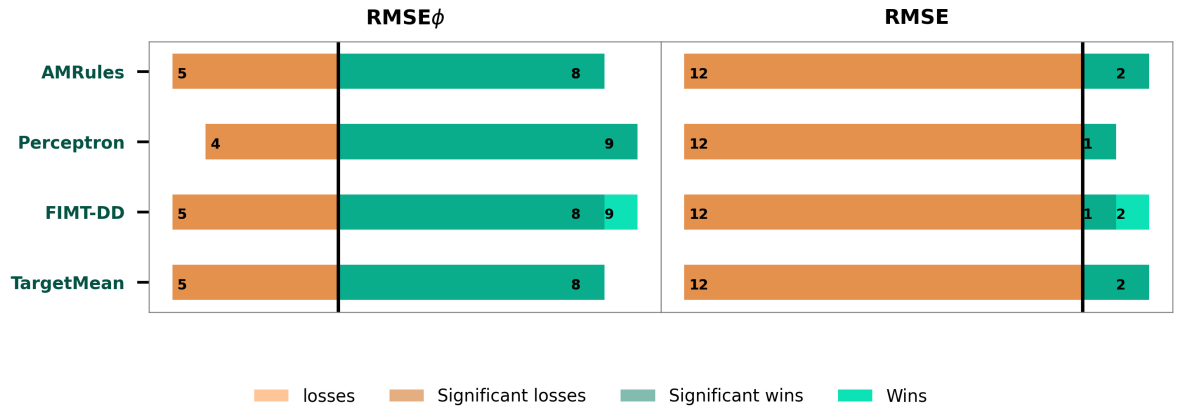
To examine this effect, we analyze the results for $thr_\phi = 0$ and illustrate how changes in this threshold influence the performance metrics. When a lower value of thr_ϕ is used to identify rare cases in the datasets, the results differ significantly. As mentioned earlier, setting $thr_\phi = 0$ ensures that all examples contribute to prediction error computation. However, in this case, relevance values range from 0 to 1, with frequent cases assigned values close to 0 and rare cases approaching 1. As a result, a substantial difference emerges between the outcomes of $RMSE_\phi$ and $RMSE$. This trend is confirmed by the results in Tables 3.4 and 3.5, as well as the graphical representations in Figure 3.4. While models trained using our sampling strategies outperform their counterparts based on $RMSE_\phi$, they are outperformed when evaluated using standard $RMSE$.

TABLE 3.4: Summary of results of Chebyshev-based Undersampling (ChebyUS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.0$) and $RMSE$.

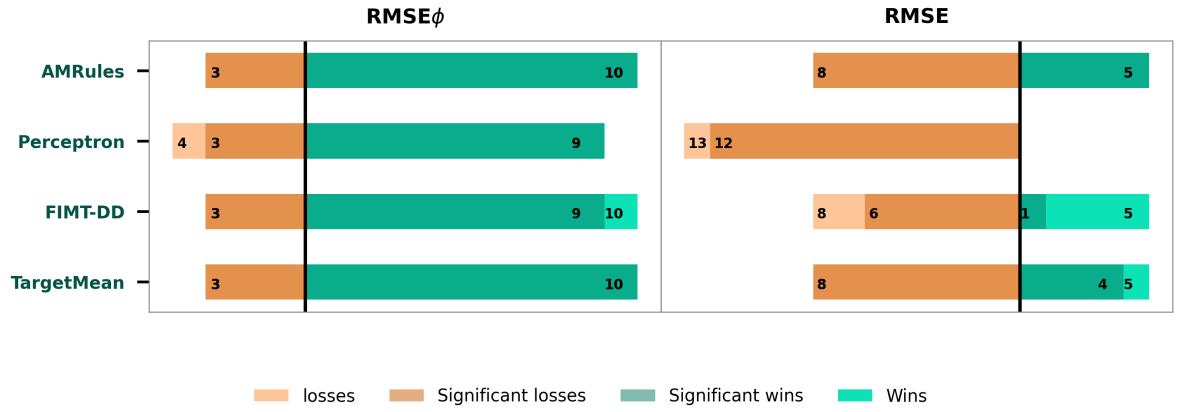
Algorithm	Approach	$RMSE_\phi$		$RMSE$	
		avg_rank	#w	avg_rank	#w
AMRules	Baseline	6.32	(5)	**4.54**	(12)
	ChebyUS	**5.50**	(8)	6.39	(2)
Perceptron	Baseline	4.29	(4)	**2.71**	(12)
	ChebyUS	**3.04**	(9)	4.54	(1)
FIMT-DD	Baseline	3.18	(5)	**2.25**	(12)
	ChebyUS	**2.96**	(8)	4.93	(1)
TargetMean	Baseline	5.68	(5)	**4.14**	(12)
	ChebyUS	**5.04**	(8)	6.50	(2)

TABLE 3.5: Summary of results of Chebyshev-based Oversampling (ChebyOS) compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ ($thr_\phi = 0.0$) and $RMSE$.

Algorithm	Approach	$RMSE_\phi$		$RMSE$	
		avg_rank	#w	avg_rank	#w
AMRules	Baseline	6.71	(3)	**5.50**	(8)
	ChebyOS	**5.68**	(10)	6.32	(5)
Perceptron	Baseline	4.32	(3)	**3.32**	(12)
	ChebyOS	**3.04**	(9)	4.61	(0)
FIMT-DD	Baseline	3.14	(3)	**2.64**	(6)
	ChebyOS	**2.11**	(9)	3.18	(1)
TargetMean	Baseline	6.11	(3)	**4.82**	(8)
	ChebyOS	**4.89**	(10)	5.61	(4)



(A) Baseline *versus* ChebyUS



(B) Baseline *versus* ChebyOS

FIGURE 3.4: Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from no sampling (Baseline) against Chebyshev-based sampling (ChebyUS and ChebyOS) using $RMSE_\phi$ ($thr_\phi = 0.0$) and $RMSE$.

However, regarding the fact that in imbalanced domain learning, prediction errors over rare cases should be more costly, a standard error metric such as $RMSE$, which gives the same weight to all examples, is not an appropriate evaluation metric for this type of tasks [12]. We only consider performance estimates obtained by $RMSE_\phi$ to evaluate the performance of our proposed methods.

To address the second research question, we investigate the impact of thr_ϕ on the results by considering four threshold levels: 0, 0.8, 0.9, and 1. The charts in Figure 3.5 illustrate the number of wins and significant wins achieved by learners trained using our sampling strategies at different thr_ϕ levels. Across all learners, threshold levels, and sampling strategies, models trained using our approaches consistently perform better. Out of 14 datasets, our methods achieve at least nine wins in every case. Additionally, the charts show an upward trend in both wins and significant wins as thr_ϕ increases from 0 to 1. This trend aligns with the assumption that rare cases lie farther from the median. As thr_ϕ increases, examples closer to the median are excluded from evaluation, reinforcing Chebyshevs probability definition. In our undersampling method, examples farther from the mean are more likely to be selected for training, while in oversampling, models are trained more frequently on these examples. Consequently, as thr_ϕ increases, the focus on these distant examples is amplified, leading to improved performance on rare cases.

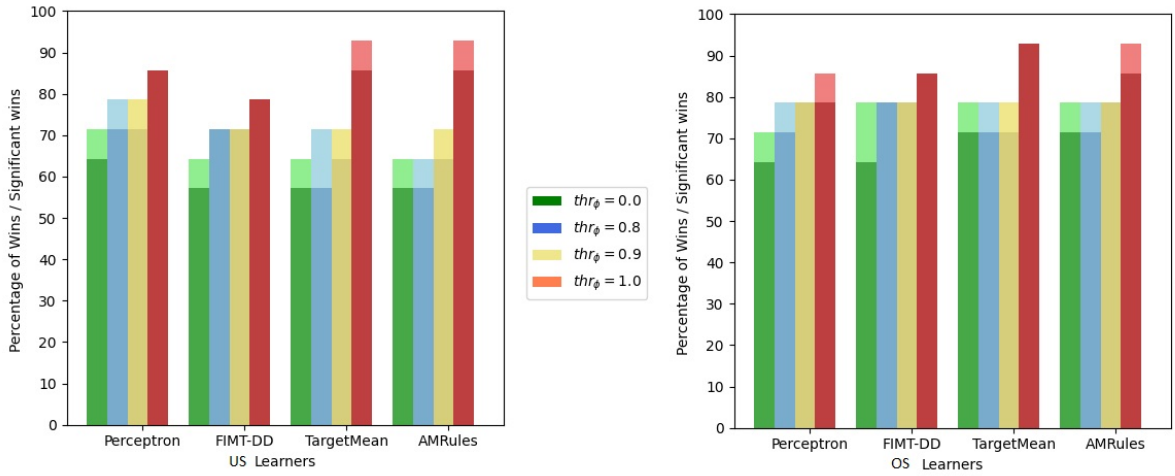


FIGURE 3.5: Wins and significant wins of the proposed ChebyUS (left) and ChebyOS (right) methods for different thr_ϕ values.

3.3.4.1 Results for low or high extreme values

To assess the effectiveness of our strategies across different types of extreme values (low or high) considered as rare cases and to address the third research question, we conducted a new set of experiments under the same conditions. In this evaluation, beyond Chebyshev's value, the extremity of the target value in the dataset also determines how each example is processed. More specifically, for an example to contribute to model training based on its Chebyshev's value, it must also belong to one of the extreme regions of the data (either the low or high extreme). This requires modifications to the sampling strategies in our algorithms. That means the condition in line 10 of Algorithm 3.1 "**if** $RandomNumber \geq p$ **then**" should be modified either to "**if** $RandomNumber \geq p$ **and** $y \geq \overline{y}$ **then**" or to "**if** $RandomNumber \geq p$ **and** $y \leq \overline{y}$ **then**", according to the type of extremes we are focusing on. Also, line 7 of Algorithm 3.2 "$k \leftarrow ComputeK(y, \overline{y}, \sigma)$" should be changed to "**if** $y \geq \overline{y}$ **then** $k \leftarrow ComputeK(y, \overline{y}, \sigma)$ **else** $k \leftarrow 1$" or to "**if** $y \leq \overline{y}$ **then** $k \leftarrow ComputeK(y, \overline{y}, \sigma)$ **else** $k \leftarrow 1$". The experimental results for low and high extreme values are summarized in Tables 3.6 and 3.7, with corresponding graphical representations in Figure 3.6.

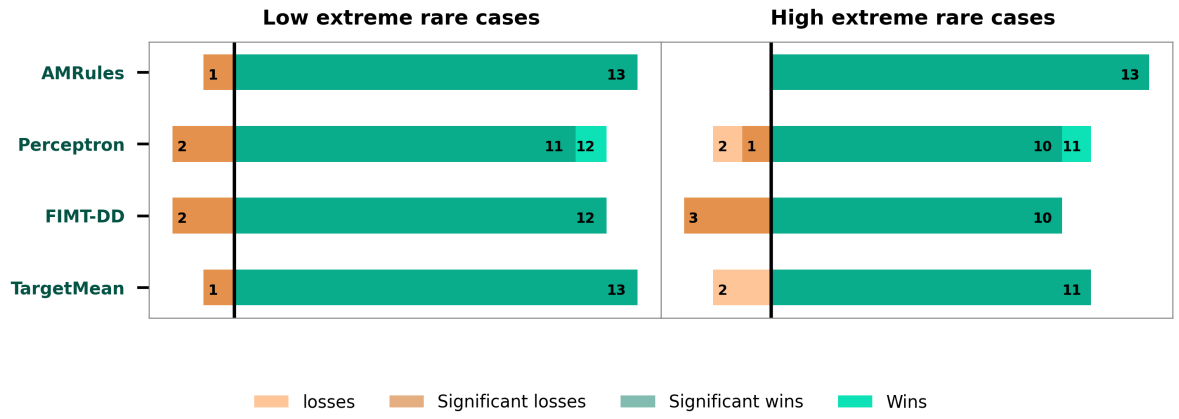
TABLE 3.6: Summary of $RMSE_\phi$ results comparing ChebyUS and Baseline considering low and high extreme rare cases ($thr_\phi = 0.8$).

Algorithm	Approach	Low Extreme Cases		High Extreme Cases	
		avg_rank	#w	avg_rank	#w
AMRules	Baseline	7.04	(1)	6.82	(0)
	ChebyUS	**3.57**	(13)	**3.43**	(13)
Perceptron	Baseline	5.11	(2)	6.07	(1)
	ChebyUS	**2.29**	(11)	**2.61**	(10)
FIMT-DD	Baseline	4.32	(2)	4.89	(3)
	ChebyUS	**3.39**	(12)	**2.57**	(10)
TargetMean	Baseline	6.32	(1)	6.50	(0)
	ChebyUS	**3.96**	(13)	**3.11**	(11)

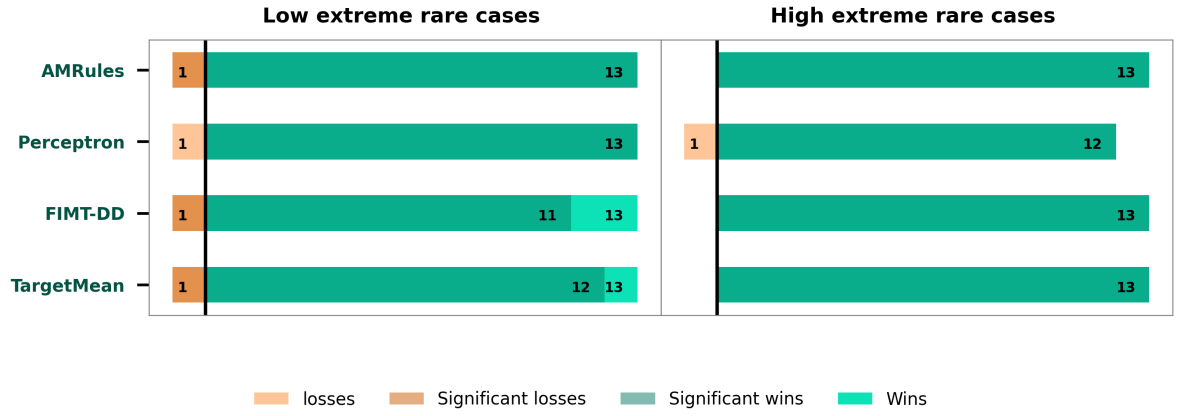
Upon examining the number of wins and significant wins illustrated in the charts and comparing them with the reference results, it becomes evident that applying our methods to only one side of the data is more effective. This observation is corroborated by the critical difference (CD) diagrams (Figures A.2 and A.3 in Appendix A.3), where, in contrast to the reference results, most models benefiting from our strategies demonstrate significantly better performance than their baseline counterparts.

TABLE 3.7: Summary of $RMSE_\phi$ results comparing ChebyOS and Baseline considering low and high extreme rare cases ($thr_\phi = 0.8$).

Algorithm	Approach	Low Extreme Cases		High Extreme Cases	
		avg_rank	#w	avg_rank	#w
AMRules	Baseline	7.39	(0)	6.86	(1)
	ChebyOS	**3.89**	(13)	**4.00**	(13)
Perceptron	Baseline	5.54	(0)	6.07	(0)
	ChebyOS	**2.64**	(12)	**2.25**	(13)
FIMT-DD	Baseline	4.71	(0)	4.89	(1)
	ChebyOS	**1.89**	(13)	**1.71**	(11)
TargetMean	Baseline	6.68	(0)	6.57	(1)
	ChebyOS	**3.25**	(13)	**3.64**	(12)



(A) Baseline *versus* ChebyUS

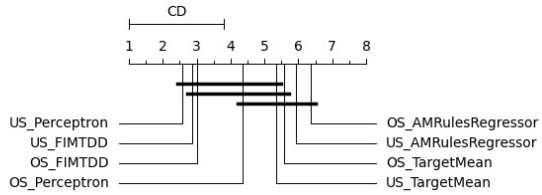


(B) Baseline *versus* ChebyOS

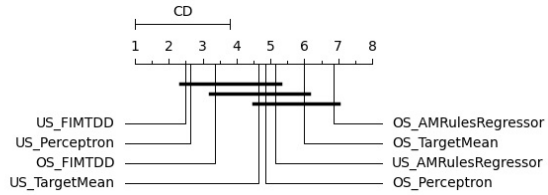
FIGURE 3.6: Total number of wins/losses, according to $RMSE_\phi$, of our sampling strategies considering low and high extreme rare cases ($thr_\phi = 0.8$).

3.3.4.2 Comparing ChebyUS against ChebyOS

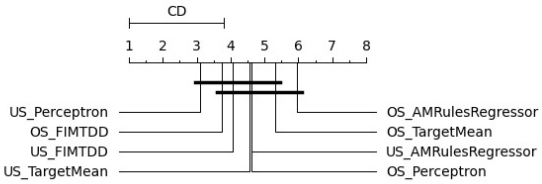
Its also worth to have a comparison between the two proposed methods. Figure 3.7 presents CD diagrams drawn based on the obtained results of all models trained by our two proposed sampling strategies. Regarding the ranks of the algorithms depicted in the diagrams, the undersampling strategy seems to offer more promising results than oversampling. Except for FIMT-DD in Figure 3.7c, all models trained by the undersampling method have outperformed their pairs counterparts and have achieved a better result in all cases. Regarding the time complexity of the algorithms, they both have $O(1)$ for each incoming instance in the stream. The incremental computation of the mean and standard deviation of the examples can be executed in constant time. However, in practice, undersampling is faster, as shown in Figure 3.8. This figure illustrates the execution times for both algorithms applied for training a Perceptron over the data sets.



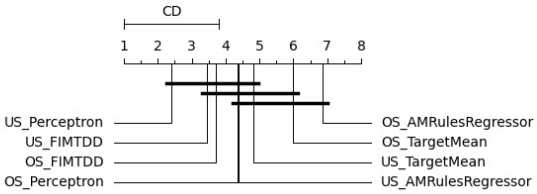
(A) both extreme rare cases ($thr_\phi = 0.8$)



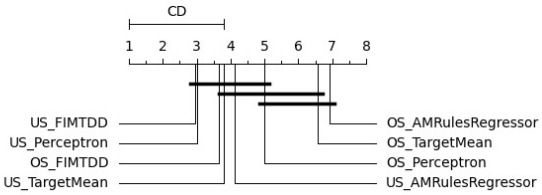
(B) both extreme rare cases ($thr_\phi = 1$)



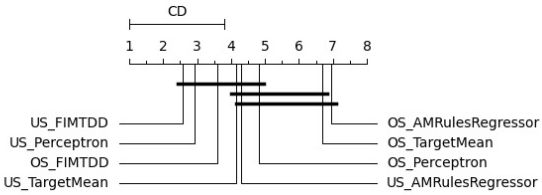
(C) low extreme rare cases ($thr_\phi = 0.8$)



(D) low extreme rare cases ($thr_\phi = 1$)



(E) high extreme rare cases ($thr_\phi = 0.8$)



(F) high extreme rare cases ($thr_\phi = 1$)

FIGURE 3.7: CD diagrams of ChebyUS and ChebyOS strategies based on obtained $RMSE_\phi$ results over different extremes types (both, low and high) of rare cases and for different levels of thr_ϕ.

Having studied the obtained results, inspected the different factors that may affect them, and analysed the time and space complexity of the algorithms, we can say yes to the fourth research question. In detail, with an acceptable time and reasonable space, the models trained

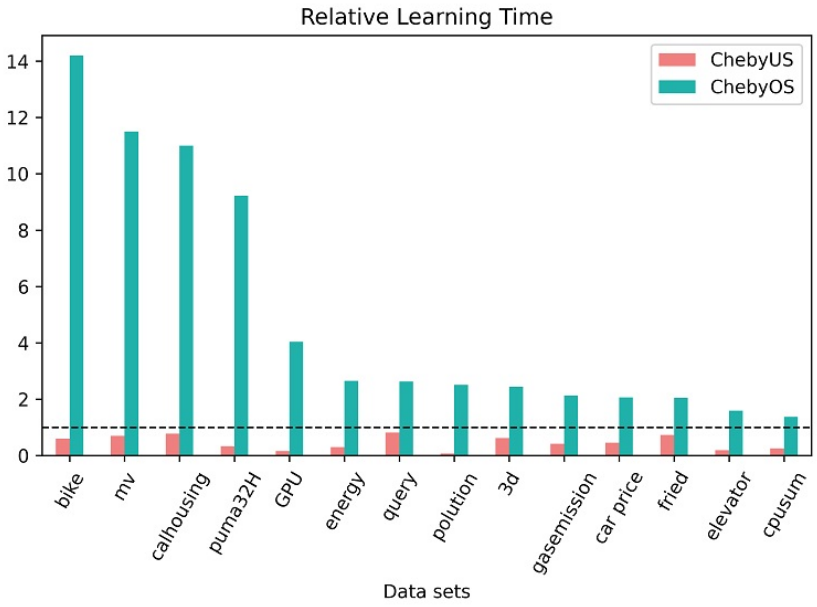


FIGURE 3.8: Average relative execution time of ChebyUS and ChebyOS for Perceptron models over all data sets. The term of comparison is the Perceptron, represented by the dotted line.

using one of our methods produce less error for rare cases which are the most important cases in the imbalanced regression problems.

3.4 Limitations

In this work, we assume that the rare cases have extreme target values, i.e. they are located at the tails of the distribution of the target variable. This is the major limitation of our work. Given that the value of t should be greater than one in Equation 3.1, rare cases need to be located at least farther from the σ of the distribution of target value to effectively contribute to the training of the models. Even though this can be seen as a limitation, it addresses many real-world applications where rare extreme values are the most important. Examples are found in the areas of meteorology, environment, and finance.

Another limitation is the concept of evolution when its source is evolving the target variable's distribution, which heavily affects the so-far computed statistics (i.e., mean and variance). However, this is mitigated after visiting sufficient new incoming examples and adding a forgotten factor in calculating the mean and variance.

The last limitation we are aware of is related to the method we use to evaluate the methods. As we mentioned before, the calculation of the relevance value for the examples is done by the $\phi()$ function. The implementation of this function in this paper is based on the box-plot statistics (especially median). On the other hand, Chebyshevs inequality is based on the target variable's mean and variance. If there is a significant difference between the median and mean of the target variable in any data set, our proposed methods might fail to produce acceptable results based on $RMSE_\phi$. Moreover, our evaluations rely on establishing a threshold for relevance function thr_ϕ, which creates artificial and crisp divisions on the continuous target variable domain. The truth is that this is still a challenging problem in the literature, and only a few proposals exist [10]. Further work will focus on this issue.

Chapter 4

Histogram Approaches for Imbalanced Data Streams Regression

In the previous chapter, we presented our initial attempt to address imbalanced data streams in the regression domain specifically. We introduced two sampling strategies based on Chebyshev's inequality: Chebyshev-based Undersampling (ChebyUS) and Chebyshev-based Oversampling (ChebyOS). ChebyUS leverages Chebyshev's inequality to calculate a probability based on the data stream's mean and variance. Incoming examples are then selectively included in the training set by comparing a randomly generated number to this Chebyshev probability, effectively reducing the dominance of frequent cases. Conversely, ChebyOS focuses on oversampling rare cases identified by their Chebyshev probability. A value K is calculated based on this probability, with higher values assigned to rarer cases. The model is then trained on the rare example K times, increasing its representation in the training data.

Using ChebyUS and ChebyOS, our previous work aimed to achieve a balanced training set with similar proportions of frequent and rare cases. This strategy helps mitigate bias towards frequent cases and potentially improves the model's performance in handling imbalanced regression tasks. However, a notable limitation of these approaches is the assumption that rare cases are confined solely to the extreme values (tails) of the target variable's distribution.

This chapter proposes novel data-level sampling strategies that overcome this limitation. Our new methods eliminate the assumption that rare cases reside only at the distribution tails, acknowledging that they may occur anywhere within the target variable's range. By enabling

the identification and handling of rare cases regardless of their location in the distribution, these advancements pave the way for more accurate and robust models.

The remainder of this chapter is organized as follows. Section 4.1 introduces the histogram-based approaches, detailing the formulation of the probability function and the proposed methods, HistUS and HistOS. Section 4.2 describes the experimental evaluation, including the relevance function, dataset characteristics, and evaluation methodology which is followed by presenting and analyzing the experimental results, highlighting the impact of the parameters on performance and a comparison with Chebyshev-based methods.

4.1 Histogram Approaches

Leveraging online histograms produced by the *Partitional Incremental Discretization* (PiD) method described in [66], we propose two novel data-level sampling strategies to tackle imbalanced regression in data streams. The first approach, termed Histogram-based Undersampling (HistUS), reduces the prevalence of frequent instances, while the second approach, called Histogram-based Oversampling (HistOS), increases the representation of rare cases in the training of the learner models. The following subsections provide a detailed explanation of each method.

4.1.1 Probability Function

We propose to provide informed sampling approaches for building regression models from imbalanced data streams, guided by the *PiD* method [66]. By employing counts and bins extracted from a dynamically created online histogram, we compute a probability value that offers insight into the degree of rarity associated with each example. A high probability suggests that the example is frequent, while a low probability indicates that it is rare. The primary objective of this probability function is to regulate the selection of instances for training, thereby implementing sampling strategies based on instance rarity. It leverages count values derived from the histogram constructed by PiD and is designed to be adaptable, enabling fine-tuning based on domain-specific knowledge of the underlying data distribution.

Assuming that the target value of the i-th instance is associated with the j-th bin, i.e. $y_i \in \left[b_{j-i}, b_j\right)$, the probability for that instance is computed by

$$P(i) = \exp(-\beta \times \rho_j) \tag{4.1}$$

where β is a parameter that controls the degree of sampling, and ρ_j represents the relative density of the j-th bin to which y_i belongs that is defined by

$$\rho_j = \frac{n_j}{\max\{n_l : l = 1, \ldots, k\}} \tag{4.2}$$

where n_j and n_l refer to the number of instances in the j-th and the l-th bin, respectively and k is the total number of bins. This scaling ensures that the count values are normalized between 0 and 1. The exponential function in Equation 4.1 assigns a higher probability to instances in bins with lower counts (rarer instances) and a lower probability to instances in bins with higher counts (more frequent instances). The parameter β adjusts the steepness of this probability decay.

4.1.2 HistUS: Undersampling Approach Using Online Histograms

Drawing on the probability function outlined in the previous section, we now introduce our undersampling method, HistUS. An initial histogram is generated based on samples from the data stream using the *PiD* algorithm. This histogram serves as a heuristic for assessing the expected rarity of incoming examples.

Algorithm 4.1 presents the main procedure, HistUS, which initializes the model and histogram and orchestrates the training and testing phases. Training the learner model without a prior estimation of the *PiD* model is ineffective. So, the warming phase is introduced to construct an initial estimation of the histogram built by the *PiD* model. During the training phase, the learner model is trained using an undersampling strategy based on probabilities derived from the histogram. Simultaneously, the *PiD* model is updated throughout the training process and, as we also see in the testing phase, to ensure that the probabilities remain accurate and reflective of the evolving data distribution. Throughout the testing phase, the model continues to process the data stream, making predictions and updating itself. During the training and testing phases, the probability value (i.e. p) is calculated for each target value y, representing its frequency within the data stream. A random number r is then generated, and if $r \leq p$, the

Algorithm 4.1 HistUS: Histogram-based Undersampling Algorithm for Data Streams

procedure HISTUS($\mathcal{S}$: data stream, n_w: warming set size, n_t: training set size, β: sampling decay)
  $model \leftarrow InitializeModel()$ ▷ Create an empty predictive model
  $hist \leftarrow PiD()$ ▷ Create an empty histogram model
  $i \leftarrow 0$
  while $i \leq n_w$ **do** ▷ Warming Phase
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$ ▷ Get next example from data stream
    $hist \leftarrow UpdatePiDModel(hist, y)$
    $i \leftarrow i + 1$
  end while
  $i \leftarrow 0$
  while $i \leq n_t$ **do** ▷ Training Phase
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
    $hist \leftarrow UpdatePiDModel(hist, y)$
    $p \leftarrow ComputeProbability(y, hist, \beta)$
    $r \leftarrow RandomNumber()$
    if $r \leq p$ **then**
      $model \leftarrow TrainModel(model, \langle \mathrm{x}, y \rangle)$
    end if
    $i \leftarrow i + 1$
  end while
  $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
  while $\langle \mathrm{x}, y \rangle \neq null$ **do** ▷ Testing Phase
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
    $\hat{y} \leftarrow Predict(model, y)$
    $hist \leftarrow UpdatePiDModel(hist, y)$
    $p \leftarrow ComputeProbability(y, hist, \beta)$
    $r \leftarrow RandomNumber()$
    if $r \leq p$ **then**
      $model \leftarrow TrainModel(model, \langle \mathrm{x}, y \rangle)$
    end if
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
  end while
end procedure

instance is selected for training. This undersampling approach mitigates the over-representation of frequent cases, resulting in a more balanced training dataset.

4.1.3 HistOS: Oversampling Approach Using Online Histograms

The estimated histogram can also be employed to oversample rare cases. The learning model may undergo multiple training iterations for a single incoming example by utilizing probability values derived from the histogram's counts and bins. The implementation of this approach is detailed in Algorithm 4.2, which closely resembles the structure of the undersampling approach

while introducing a reduction coefficient α. This parameter regulates the oversampling rate by progressively reducing the computed probability for each incoming example until the termination condition in the training and test phases is satisfied.

During the training phase, the data stream is processed sequentially, updating both the histogram model (*hist*) based on the *PiD* method and the learner model (*model*). Each example is used for training at least once. However, rare examples that have a higher probability value p may undergo multiple training iterations, enhancing the model's ability to learn from infrequent cases. The training process continues until the predefined training set size (n_t) is reached. The probability p is computed based on the histogram and adjusted using the reduction coefficient α and the sampling decay parameter β. If a randomly generated number r satisfies $r \leq p$, the model undergoes additional training iterations, with p being progressively reduced by a factor of α after each iteration. This mechanism ensures that rare instances contribute more significantly to the training process while preventing excessive oversampling. A smaller α value results in more frequent training iterations for rare cases, improving the model's capacity to learn from infrequent events. The while loop continues until the condition $r > p$ is met, ensuring that additional training iterations are proportional to the rarity of the example. The testing phase continuously processes the data stream, making predictions, updating the histogram, and retraining the model. The oversampling mechanism remains consistent with the training phase, ensuring continued model adaptation.

Algorithm 4.2 HistOS: Histogram-based Oversampling Algorithm for Data Streams

procedure HISTOS($\mathcal{S}$: data stream, n_w: warming set size, n_t: training set size, α: reduction coefficient, β: sampling decay)
  $model \leftarrow InitializeModel()$ ▷ Create an empty predictive model
  $hist \leftarrow PiD()$ ▷ Create an empty histogram model
  $i \leftarrow 0$
  while $i \leq n_w$ **do** ▷ Warming Phase
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$ ▷ Get next example from data stream
    $hist \leftarrow UpdatePiDModel(hist, y)$
    $i \leftarrow i + 1$
  end while
  $i \leftarrow 0$
  while $i \leq n_t$ **do** ▷ Training Phase
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
    $hist \leftarrow UpdatePiDModel(hist, y)$
    $model \leftarrow TrainModel(model, \langle \mathrm{x}, y \rangle)$
    $p \leftarrow ComputeProbability(y, hist, \alpha, \beta)$
    $r \leftarrow RandomNumber()$
    while $r \leq p$ **do**
      $model \leftarrow TrainModel(model, \langle \mathrm{x}, y \rangle)$
      $p \leftarrow p / \alpha$
    end while
    $i \leftarrow i + 1$
  end while
  $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
  while $\langle \mathrm{x}, y \rangle \neq null$ **do** ▷ Testing Phase
    $\hat{y} \leftarrow Predict(model, y)$
    $hist \leftarrow UpdatePiDModel(hist, y)$
    $p \leftarrow ComputeProbability(y, hist, \alpha, \beta)$
    $r \leftarrow RandomNumber()$
    while $r \leq p$ **do**
      $model \leftarrow TrainModel(model, \langle \mathrm{x}, y \rangle)$
      $p \leftarrow p / \alpha$
    end while
    $\langle \mathrm{x}, y \rangle \leftarrow GetExample(\mathcal{S})$
  end while
end procedure

4.2 Experimental Evaluation

This section presents the experimental evaluation of the proposed histogram-based methods for handling imbalanced regression in data streams. The primary objective of this evaluation is to assess the effectiveness of the histogram-based undersampling (HistUS) and oversampling (HistOS) strategies in improving the predictive performance of regression models, particularly for rare and extreme target values. To this end, we conduct extensive experiments on synthetic

and real-world datasets, integrating the proposed methods into widely used regression algorithms. The evaluation focuses on key performance metrics, including relevance-weighted error functions, to evaluate the models based on their ability to capture rare instances. The following subsections provide a detailed description of the experimental setup, the used datasets, the evaluation metrics, the obtained results, and a comprehensive analysis of the findings.

4.2.1 Relevance Function

In our pursuit of conveying the varying importance attributed to different values of a target variable, we propose a new method to define the so-called relevance function $\phi\colon \mathcal{Y} \to [0,1]$ that maps the continuous target variable domain into a scale of importance [10]. Through our method, the relevance linked to each target value is contingent on the number of instances within the same range, using counts derived from the Equal-width Histogram method. We begin by constructing an equal-frequency histogram for the domain of the target variable, defined by a set of k bins. Comparing the count values associated with each bin interval can give us insights into each interval's density. So, depending on the specific bin interval in which a target value resides, we gauge its relative density to determine its relevance. The relevance value for a target value y_i belonging to the j-th bin is estimated by

$$\phi(y_i) = 1 - \rho_j \tag{4.3}$$

where ρ_j represents the relative density of the j-th bin, as defined by Equation 4.2.

The function $\phi()$ is a non-continuous function that conveys domain-specific bias, with values of 0 and 1 representing the minimum and maximum relevance, respectively. The relevance values for bins containing a high number of examples, typically considered frequent cases, are low, whereas the values are high for bins with a small number of examples.

4.2.2 Experiments on Synthetic and Real-World Data

To evaluate the effectiveness of our approach in handling data streams where rare cases may emerge at any pointcontrary to our previous assumption that rarity is confined to extreme valueswe conduct experiments on both a synthetic and a real-world dataset titled *Range Queries Aggregates* [58] . In this section, we provide a detailed analysis of the experimental setup and the impact of our sampling strategies, focusing on the synthetic dataset. This choice is

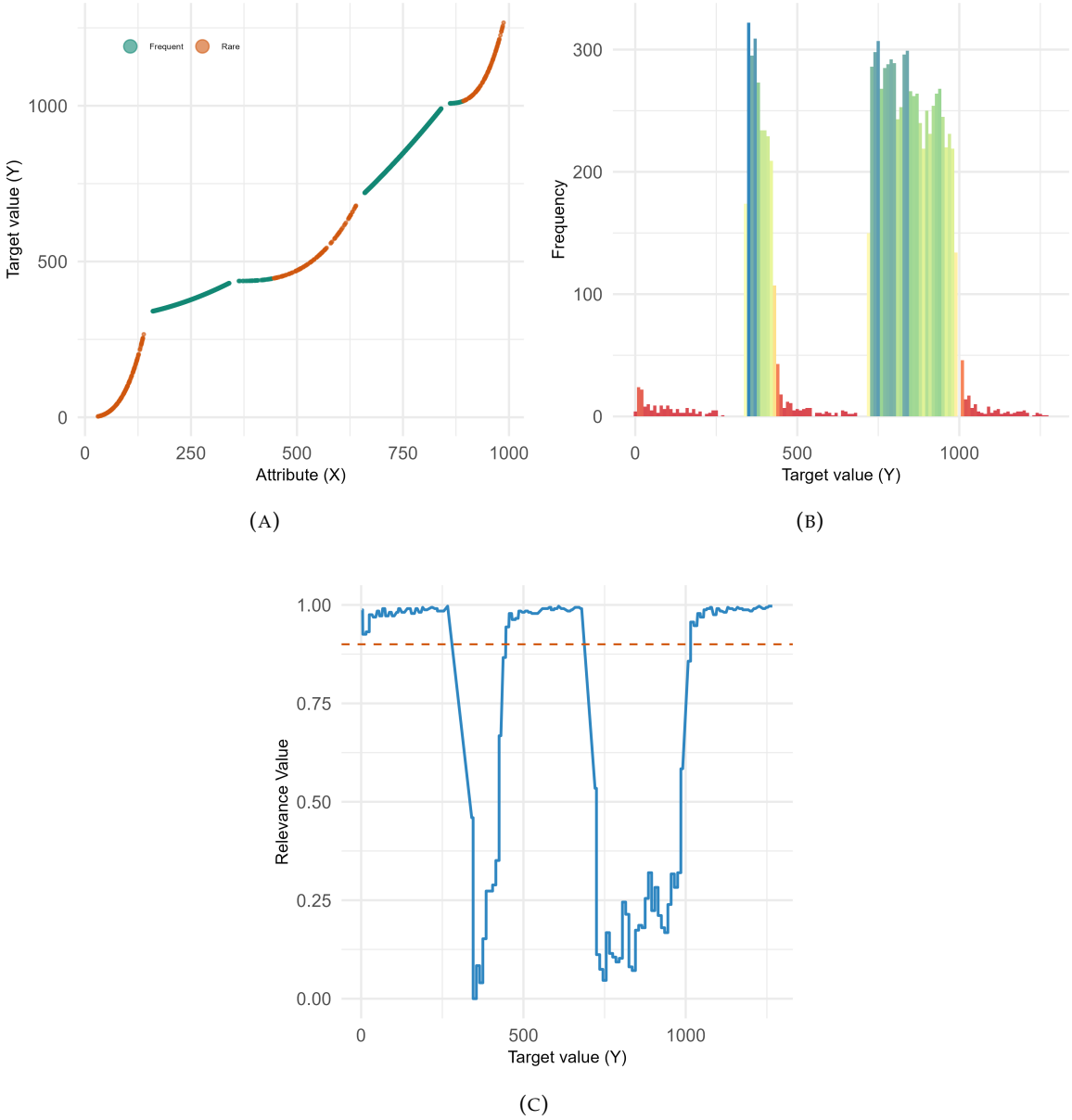
motivated by its lower dimensionality and the well-defined relationship between its independent and dependent variables, which facilitate visualization and interpretation. Nevertheless, the observed behavior is consistent across the real-world dataset. Further details on the Range Queries Aggregates dataset are provided in Appendix A.6.

We created a simple synthetic dataset and the details of the dataset are provided in Appendix A.5. Figure 4.1 illustrates the distribution of target values y in the synthetic dataset, emphasizing the distinction between frequent and rare instances through color-coded segments. A threshold (thr_ϕ) is applied to the relevance function to identify rare values, with all instances where $thr_\phi \geq 0.9$ classified as rare cases.

Figure 4.1a illustrates the relationship between the independent variable x and the dependent variable y, highlighting the distinction between rare and frequent cases. Figure 4.1b depicts the histogram of y, demonstrating the concentration of frequent instances within specific intervals and the scattered distribution of rare cases. Figure 4.1c represents the relevance values assigned to different target values, showing a sharp decline in relevance around the densely populated frequent regions, while maintaining high values in the sparse rare areas. The dashed orange line represents the threshold (thr_ϕ) used to distinguish rare cases.

To illustrate the limitations of Chebyshev-based approaches for this type of dataset, we depict its derived values in Figure 4.2. The first two plots at the top of the figure depict the inverse Chebyshev probability ($1 - \text{Probability}$) and the K values. The inverse Chebyshev probability governs the selection likelihood of instances in the ChebyUS method, prioritizing rare examples (with high inverse probability values). Conversely, K dictates the replication frequency of examples during training in the ChebyOS method, where larger K values correspond to repeated oversampling of rare instances. These plots highlight how the Chebyshev-based probability measure assigns low values in the central region of the distribution while increasing toward the tails. Despite the presence of rare cases in the central region confirmed by high relevance values in Fig 4.2 the Chebyshev probability and K values remain low, leading to an underestimation of rare instances in these regions. This results in inadequate model training, as the approach assumes that rarity is confined to the distribution's tails.

The bottom plots in the figure illustrate the probability values and the number of oversampled cases obtained through histogram-based approaches. Unlike Chebyshev, these methods effectively detect rare cases across the entire distribution, including the central region. The higher probability and oversampling values in this region demonstrate the superiority of the



(A) (B)

(C)

FIGURE 4.1: Synthetic dataset (a), histogram of target values (b) and respective relevance values (c) of common and rare cases considering thr_ϕ of 0.9.

histogram-based approach in capturing rare instances, ensuring sufficient training data across all rare areas of the dataset. Consequently, we expect these methods to provide a more reliable framework for training machine learning models in scenarios where rare cases are not at the tails of the distribution.

Figure 4.3 presents the predicted values for the target variable using FIMT-DD models with default parameters, trained with the proposed Chebyshev- and histogram-based strategies. Since the target variable domain consists of distinct regions governed by different nonlinear

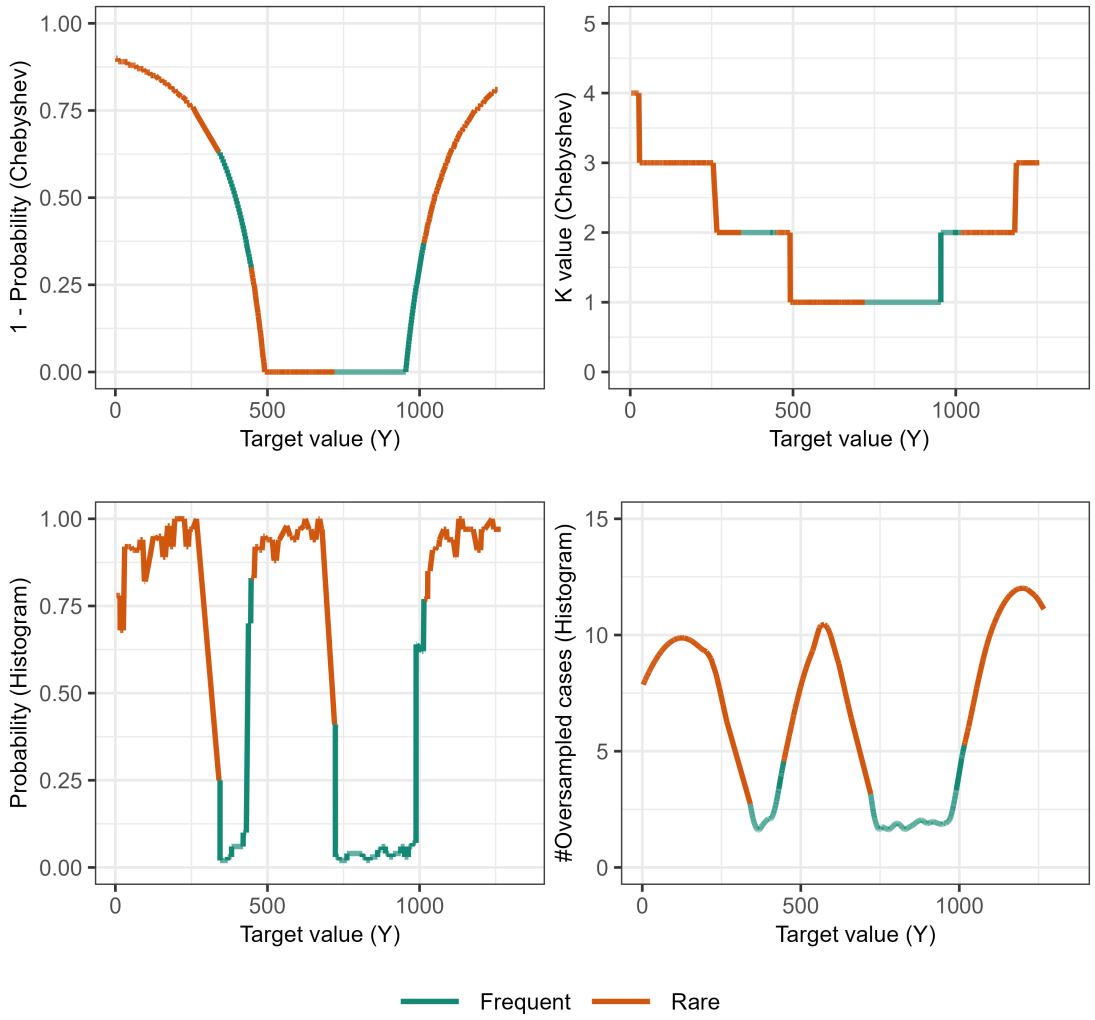


FIGURE 4.2: Comparison of Probability and K values between the Chebyshev-based and the histogram-based approaches for the synthetic dataset.

functions and FIMT-DD can identify these splits and fit local models for each segment, we selected this model. The top plots in the figure represent models trained with the Chebyshev-based approach, while the bottom plots illustrate the performance of models trained using histogram-based methods.

A clear discrepancy is observed between the predictions: histogram-based models exhibit a superior ability to capture rare cases in the central region of the target variable. This aligns with our earlier findings in Figure 4.2, where the Chebyshev-based probability measure failed to effectively identify rare cases in this region. As a result, models trained with the Chebyshev approach show higher prediction errors, as reflected in the deviation of predicted values from the true values. In contrast, histogram-based models successfully learn from rare cases throughout

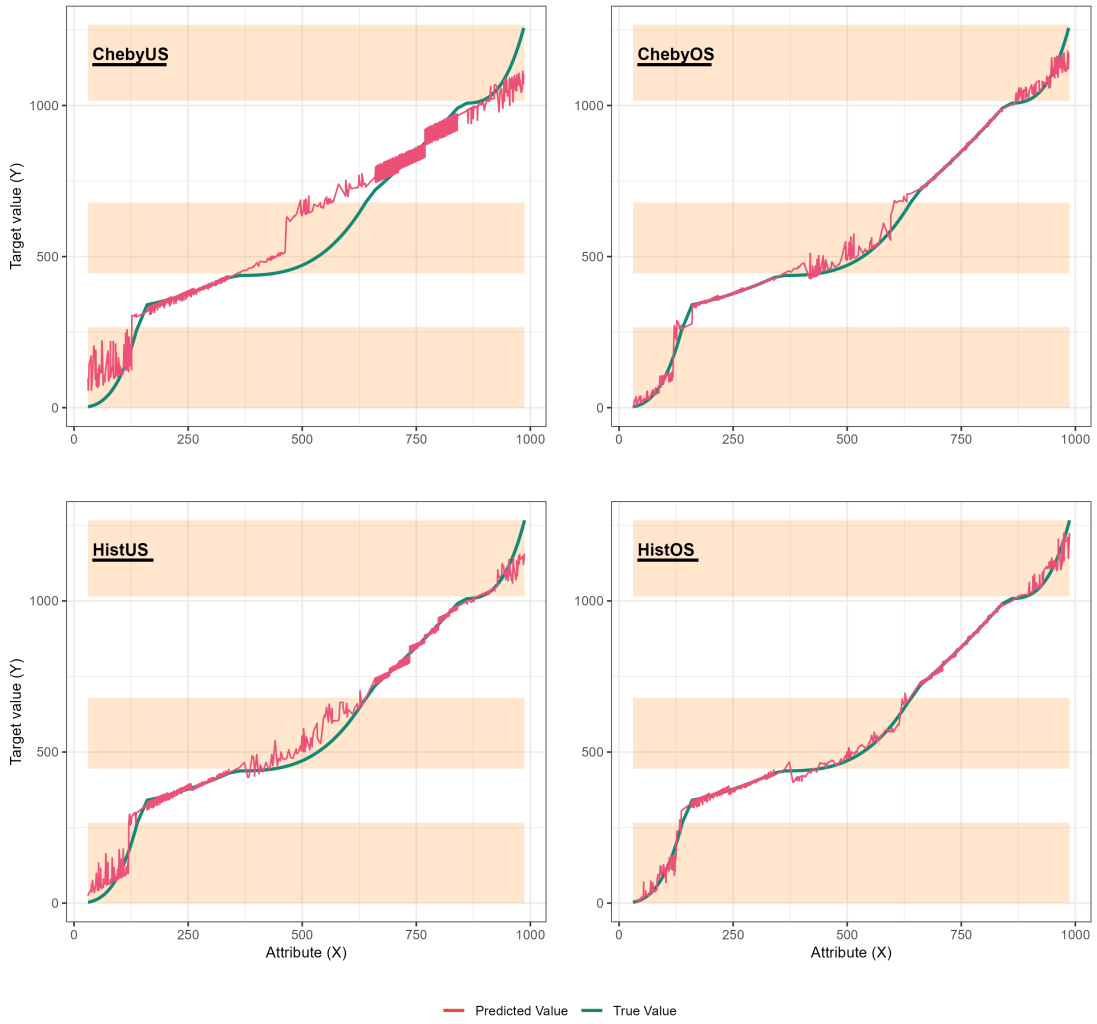


FIGURE 4.3: True target values and respective predictions using the Chebyshev-based sampling and histogram-based sampling for the synthetic dataset. The orange-highlighted areas in the background correspond to the rare regions in the target value domain.

the distribution, resulting in improved predictive performance and better generalization.

The results for the both datasets (synthetic and Range Queries Aggregates) are summarized in Table 4.1, which presents the average errors obtained from FIMT-DD models trained with different sampling strategies over ten runs, evaluated using $RMSE_{\phi}$, $SERA$, and $RMSE$ metrics. The table also includes the standard deviations and the ranking of each method based on predictive performance, where lower values indicate better performance. In our implementation, the β and α parameters for HistUS and HistOS are set to 4 and 1.02, respectively. For the ChebyUS approach, we set the second chance parameter to 0.15, following the value used in the previous chapter.

The results indicate that histogram-based sampling methods (HistUS and HistOS) consistently outperform their Chebyshev-based counterparts (ChebyUS and ChebyOS) across all evaluation metrics. Among them, HistOS achieves the lowest $RMSE$ and $RMSE_\phi$ values, demonstrating its effectiveness in improving predictive accuracy and minimizing error propagation. It is important to note the difference in magnitude between $RMSE$ and $RMSE_\phi$. The $RMSE_\phi$ metric only considers the prediction errors for instances with a relevance value greater than a predefined threshold (thr_ϕ), set to 0.90 in this paper, whereas $RMSE$ accounts for the prediction error across the entire dataset. Since most instances belong to frequent regions, and models tend to learn these regions well due to the abundance of training examples, the overall $RMSE$ remains lower. In contrast, rare regions, which are given higher importance in $RMSE_\phi$, are more challenging to learn due to fewer training examples, leading to a higher error magnitude in $RMSE_\phi$ compared to $RMSE$.

For the $SERA$ metric, ChebyUS achieves the lowest score, suggesting its strength in capturing overall errors across the entire domain. However, when evaluating predictive performance specifically in rare regions, as measured by $RMSE_\phi$, ChebyUS performs significantly worse than histogram-based methods, ranking last in this metric. Meanwhile, ChebyOS, although more accurate than undersampling approaches (ChebyUS and HistUS) in rare regions, still falls short of outperforming its histogram-based counterpart, HistOS. Overall, the findings confirm the superiority of histogram-based methods, particularly HistOS, in achieving lower errors across different evaluation criteria.

To finalize this analysis, it is essential to emphasize that the computed errors account for all rare cases across the entire distribution, including the extreme values in the tails of the target value distribution. As a result, in certain models and datasets, Chebyshev-based methods may produce lower errors for extreme rare values compared to histogram-based methods, leading to overall lower metric values.

In the following subsections, we conduct a detailed investigation into the influence of the parameters β and α on this synthetic dataset. Specifically, we analyze their impact on the probability values and the oversampling rates calculated in the HistUS and HistOS algorithms.

4.2.2.1 Impact of parameter β on the probability function

Figure 4.4 illustrates the influence of different values of the parameter β on the probability values obtained over our synthetic dataset. The parameter β governs how frequent and rare

TABLE 4.1: Summary of results of no sampling (Baseline) and sampling strategies (HistUS, HistOS, ChebyUS, ChebyOS) on the synthetic and Range Queries Aggregates (RQA) datasets: mean, standard deviation and average rank of $RMSE_\phi$, $SERA$, and $RMSE$ for the FIMT-DD algorithm.

Dataset	Approach	$RMSE_\phi$	avg.rank	$SERA$	avg.rank	$RMSE$	avg.rank
Synthetic	Baseline	61.16 ± 5.12	(4.0)	0.87 ± 0.00	(4.2)	13.93 ± 0.28	(4.0)
	HistUS	51.38 ± 10.22	(3.0)	0.78 ± 0.00	(2.7)	12.62 ± 0.44	(3.0)
	HistOS	**26.17 ± 2.02**	(1.0)	0.76 ± 0.00	(2.3)	**6.82 ± 0.28**	(1.0)
	ChebyUS	90.31 ± 83.02	(5.0)	**0.59 ± 0.00**	(1.0)	26.63 ± 3.33	(5.0)
	ChebyOS	37.87 ± 8.09	(2.0)	0.89 ± 0.00	(4.8)	8.60 ± 0.26	(2.0)
RQA	Baseline	105.15 ± 15.85	(4.0)	0.90 ± 0.00	(5.0)	65.90 ± 3.41	(3.9)
	HistUS	98.23 ± 7.60	(2.9)	0.88 ± 0.00	(2.8)	63.20 ± 1.28	(3.0)
	HistOS	**90.57 ± 4.24**	(1.2)	0.87 ± 0.00	(2.5)	**59.55 ± 1.73**	(1.3)
	ChebyUS	119.31 ± 12.29	(5.0)	**0.83 ± 0.00**	(1.0)	86.57 ± 2.39	(5.0)
	ChebyOS	92.93 ± 6.44	(1.9)	0.88 ± 0.00	(3.7)	60.27 ± 2.38	(1.8)

cases are favored in the selection process. As shown in the figure, higher values of β result in a steeper decline in probability, increasing the likelihood of selecting rare cases and reducing the selection of frequent cases. Conversely, lower values of β flatten the probability curve, making the selection more uniform across different bins.

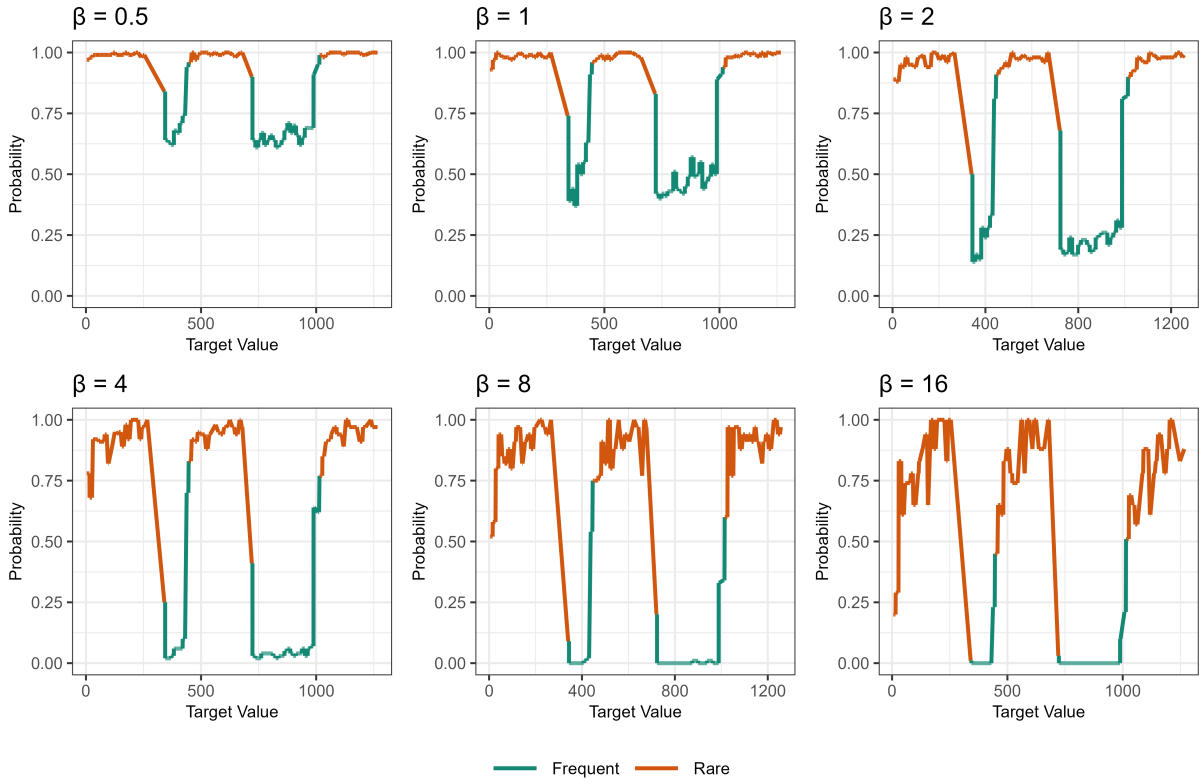


FIGURE 4.4: Influence of different values of the sampling decay parameter β on the probability values obtained over the synthetic dataset.

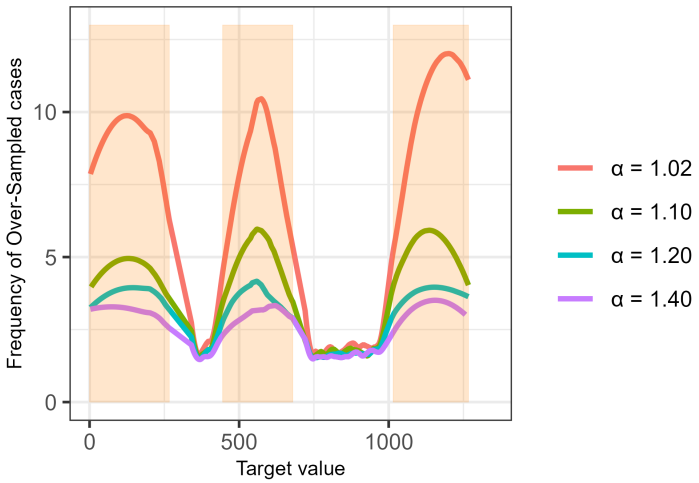


FIGURE 4.5: Effect of different values of the reduction coefficient parameter α on the oversampling rate. The parameter β is fixed at 4, while α is varied to analyze its impact on oversampling frequency across target values. The orange-highlighted areas in the background correspond to the rare regions in the target value domain.

4.2.2.2 Impact of parameter α on the oversampling rate

Figure 4.5 illustrates the effect of different values of the parameter α on the frequency of oversampling across the target values of the generated synthetic dataset. The x-axis represents the target values, while the y-axis denotes the times each target value has been oversampled. The curves correspond to different α values, where lower α leads to a higher oversampling frequency. This trend highlights the role of α in controlling the balance between rare and frequent target values, with smaller α intensifying the oversampling of rare cases. It is worth mentioning that the numbers on the y-axis have been smoothed to enhance clarity, ensuring a more interpretable representation of the underlying trends.

4.2.3 Experiments on Benchmark Data

Our experiments were conducted on twelve benchmark datasets sourced from the UCI repository [58]. Our analysis designated examples with a relevance value exceeding 0.9 as rare cases. It is important to note that no preprocessing step has been done on the data set, and all features of the data sets feed the learner models by their original form and value, ensuring the thoroughness and reliability of our experimental setup. Appendix A.7 provides key dataset statistics, including the number of attributes, examples, and the count and percentage of rare cases for each dataset as well as target value distributions and the relevance values assigned

to each target value. The proposed methodologies were implemented in Kotlin 1.4.10 using MOA framework [60], an open-source tool designed for mining and analyzing large datasets streamingly. We used R programming language - version 4.4.2 for error computation and result analysis.

In alignment with our previous chapter, we utilized four diverse regression models as base learners to alleviate potential algorithm-dependent biases that could introduce distortions to the results. These models included the perceptron [62], fast incremental model trees with drift detection (FIMT-DD) [18], TargetMean, and AMRules [64]. The TargetMean algorithm is a simple baseline for data stream regression. It incrementally updates the mean of the target variable using an online update rule, predicting the running mean for each incoming instance. The use of these diverse models is meant to assess the robustness of our study. In each experiment, we employed two instances of each learner with identical initial parameters. One instance was trained using our HistOS sampling method, while the other, referred to as the Baseline, underwent training with all examples in the dataset.

To evaluate each model, we used the $RMSE$, $RMSE_\phi$ [11], and $SERA$ [10] evaluation functions. For the $RMSE_\phi$, The threshold thr_ϕ is set to 0.9 in our experiments. In the $SERA$ calculation, we normalize the computed errors relative to the total error when all data points are considered. This normalization is achieved by dividing each error value by the initial full-dataset error, ensuring that the largest error is approximately one and all others are expressed as fractions of this value. Thus, the $SERA$ scores become independent of absolute error scales, enabling meaningful comparisons between models and datasets.

Each experiment was conducted ten times, and the results were reported as averages and standard deviations. Throughout all experiments, we adhered to a prequential evaluation procedure [8], where models were continuously updated after processing each example. Specifically, each example was used for both testing and training. The evaluation involved initially assessing the model on the instance, followed by a single incremental learning step. Moreover, 15% of examples in each dataset were allocated to the warming phase, 20% for training purposes, and the remaining 65% earmarked for model testing. Regarding the parameter β within the probability function, as previously highlighted, its value signifies a balance between the algorithm's capability to detect rare cases and its inclination to disregard frequent ones. We maintained a consistent value of 4 for β across all experiments after minor adjustments to our datasets. Nonetheless, optimizing this parameter could yield improved outcomes in specific

TABLE 4.2: Summary of results of histogram-based sampling strategy HistUS compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$, $SERA$, and $RMSE$.

Algorithm	Approach	$RMSE_\phi$		$SERA$		$RMSE$	
		avg_rank	#w	avg_rank	#w	avg_rank	#w
AMRules	Baseline	3.83	(0)	6.25	(0)	**2.38**	(8)
	HistUS	**2.42**	(8)	**3.58**	(10)	3.83	(1)
Perceptron	Baseline	5.58	(1)	5.50	(1)	**3.92**	(10)
	HistUS	**3.83**	(9)	**2.75**	(10)	5.50	(0)
FIMT-DD	Baseline	3.92	(1)	5.33	(0)	**2.79**	(8)
	HistUS	**2.96**	(7)	**3.0**	(12)	4.67	(1)
TargetMean	Baseline	7.46	(0)	6.62	(1)	**5.83**	(9)
	HistUS	**6.00**	(10)	**2.96**	(9)	7.08	(0)

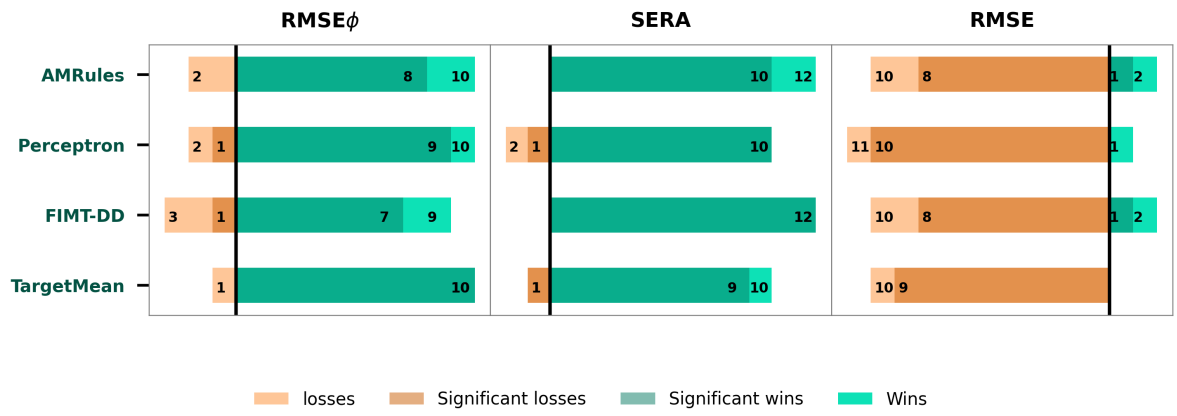
scenarios, albeit these are not detailed here. Similarly, for the parameter α in HistOS, we set its value to 1.02 across all the experiments.

4.2.3.1 Results

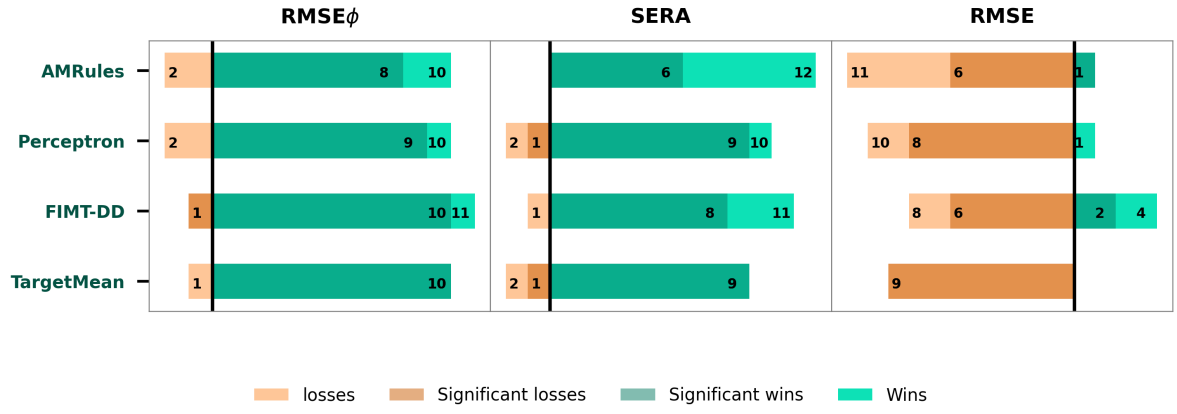
We now provide a concise overview of our results, while a comprehensive breakdown of all findings is shown in Tables A.11-A.16 and Figures A.22-A.26 in Appendix A.8.

To enhance the clarity and interpretability of our results, we have transformed each table into a graphical representation. Figure 4.6 illustrate the wins and losses observed for each learner employing our sampling strategies compared to its corresponding baseline model. The significance of these outcomes, assessed at a 95% confidence level using the Wilcoxon signed-rank test, is also depicted. The bar charts categorize the results into significant wins (substantially better performance), wins (better performance but not statistically significant), significant losses (substantially worse performance), and losses (worse performance but not statistically significant). Additionally, Critical Difference (CD) diagrams, employing the post-hoc Nemenyi test [65], provide a comparative assessment of the learners' contributions across different result groups in the experiments in Figures A.22-A.24 in Appendix A.8.

Tables 4.2 and 4.3 also summarize the experimental results, presenting the average ranks and the number of significant wins for various regression algorithms under three conditions: no sampling (baseline), histogram-based undersampling (HistUS), and histogram-based oversampling (HistOS). The results indicate that both HistUS and HistOS outperform the baseline in predicting



(A) Baseline *versus* HistUS



(B) Baseline *versus* HistOS

FIGURE 4.6: Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from no sampling (Baseline) against histogram-based sampling (HistUS and HistOS) using $RMSE_\phi$, *SERA*, and *RMSE*.

TABLE 4.3: Summary of results of histogram-based sampling strategy HistOS compared with no sampling (Baseline): average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$, *SERA*, and *RMSE*.

Algorithm	**Approach**	$RMSE_\phi$		*SERA*		*RMSE*	
		avg_rank	#w	avg_rank	#w	avg_rank	#w
AMRules	Baseline	4.25	(0)	5.92	(0)	**2.29**	(6)
	HistOS	**2.58**	(8)	**3.92**	(6)	3.67	(1)
Perceptron	Baseline	5.50	(0)	5.42	(1)	**4.21**	(8)
	HistOS	**3.75**	(9)	**2.75**	(9)	5.71	(0)
FIMT-DD	Baseline	4.33	(1)	5.17	(0)	**2.88**	(6)
	HistOS	**2.33**	(10)	**3.25**	(8)	4.0	(2)
TargetMean	Baseline	7.38	(0)	6.46	(1)	**6.04**	(9)
	HistOS	**5.88**	(10)	**3.12**	(9)	7.21	(0)

rare cases, as reflected by their lower average ranks and higher numbers of significant wins in the $RMSE_\phi$ and $SERA$ metrics. This suggests that the proposed histogram-based sampling strategies are effective in addressing the challenges of imbalanced regression data. Despite the improvements in rare case prediction, the baseline models tend to achieve better overall regression performance, as measured by the standard $RMSE$ metric. This suggests that while histogram-based sampling improves performance in prediction of rare cases, it may compromise overall regression accuracy.

Overall, these results highlight the effectiveness of the proposed histogram-based sampling strategies, HistUS and HistOS, particularly in addressing the online imbalanced regression problem and improving predictions for rare cases, though at the potential cost of overall regression performance.

4.2.3.2 Comparative Analysis

This section presents a comprehensive comparative analysis of the proposed histogram-based sampling strategies (HistUS and HistOS) against each other and their Chebyshev-based counterparts (ChebyUS and ChebyOS). The evaluation focuses on three key performance metrics $RMSE_\phi$, $SERA$, and traditional $RMSE$ to assess their effectiveness in rare case prediction and overall regression performance. We first compare HistUS and HistOS to understand their relative strengths in balancing rare case emphasis with model generalization. Subsequently, we contrast the histogram-based methods with the earlier Chebyshev-based approaches, analyzing their predictive outcomes across diverse regression algorithms, including AMRules, Perceptron, FIMT-DD, and TargetMean.

Comparison between HistUS and HistOS methods The results of the comparison, summarized in Figure 4.7 and Table 4.4, provide insights into the effectiveness of these strategies in enhancing predictive performance for rare cases. The analysis of $RMSE_\phi$ demonstrates that HistOS consistently outperforms HistUS in rare case prediction, as indicated by its lower average ranks and a higher number of significant wins across all four regression algorithms. Notably, HistOS significantly improves FIMT-DD, securing the highest number of wins. On the other hand, HistUS demonstrates superior performance based on the $SERA$ metric, particularly for AMRules, Perceptron, and FIMT-DD. FIMT-DD achieves the highest number of significant wins (9) with HistUS, followed closely by AMRules with 8 wins, indicating that HistUS is more

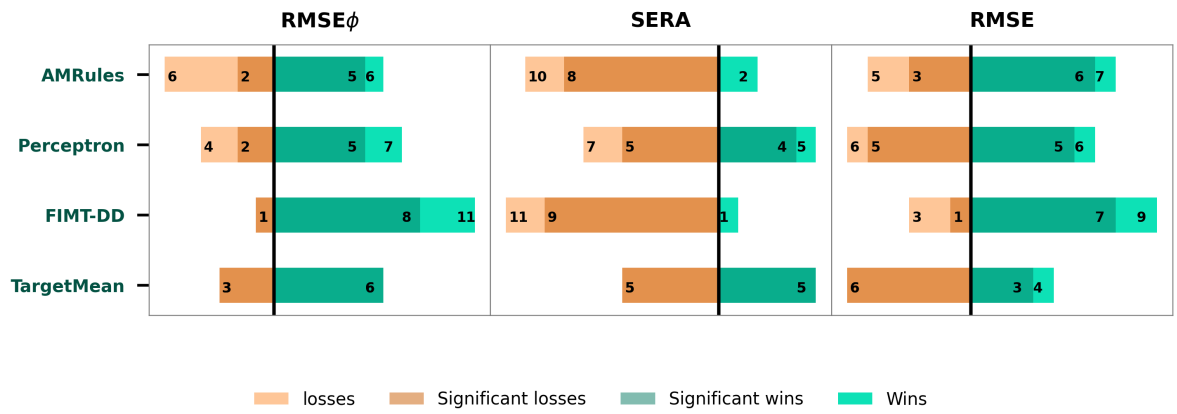


FIGURE 4.7: Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from histogram-based oversampling (HistOS) *versus* histogram-based undersampling (HistUS) using $RMSE_\phi$, $SERA$, and $RMSE$.

effective in optimizing $SERA$ for these models. However, for TargetMean, the choice between HistUS and HistOS has little impact, as their average ranks and significant wins remain similar.

Conversely, when evaluating overall regression performance using $RMSE$, the performance gap between HistUS and HistOS varies across models. HistOS again demonstrates strong results for FIMT-DD and AMRules while achieves a more balanced performance across Perceptron. However, the results for TargetMean indicate that HistOS may lead to performance degradation in $RMSE$, evidenced by its higher rank and fewer significant wins. The details of the results are presented in Tables A.17 to A.19 and Figure A.25 in Appendix A.8.

Overall, the results highlight a clear trade-off: while HistOS generally improves rare case prediction and shows greater significance in $RMSE_\phi$, HistUS performs better in terms of $SERA$. Meanwhile, the impact on $RMSE$ varies depending on the model.

Comparing Histogram and Chebyshev-based approaches This section presents a comparative analysis of histogram-based and Chebyshev-based sampling strategies in both undersampling (ChebyUS, HistUS) and oversampling (ChebyOS, HistOS) scenarios. The evaluation is conducted using $RMSE_\phi$ and $SERA$ metrics across four regression algorithms to assess the effectiveness of each approach in addressing rare case predictions. To ensure a fair comparison, all datasets are designed with rare cases positioned exclusively in the extreme tails of the target value distribution, as Chebyshev-based methods can only detect rare regions in these areas. The results, presented in Figure 4.8 and Tables 4.5 and 4.6, highlight how effectively each method addresses rare case predictions.

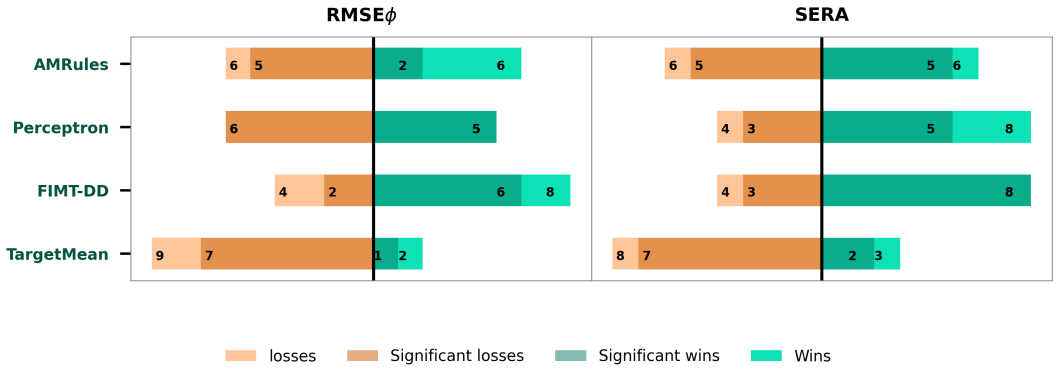
TABLE 4.4: Comparison of histogram-based sampling strategies: average rank and number of significant wins (#w) at a 95% confidence level, according to $RMSE_\phi$, $SERA$, and $RMSE$.

Algorithm	Approach	$RMSE_\phi$		$SERA$		$RMSE$	
		avg_rank	#w	avg_rank	#w	avg_rank	#w
AMRules	HistUS	3.25	(2)	**4.75**	(8)	3.25	(3)
	HistOS	**3.17**	(5)	6.42	(0)	**2.75**	(6)
Perceptron	HistUS	4.71	(2)	**3.75**	(5)	4.83	(5)
	HistOS	**4.38**	(5)	4.33	(4)	**4.50**	(5)
FIMT-DD	HistUS	4.42	(1)	**3.75**	(9)	4.33	(1)
	HistOS	**2.83**	(8)	5.67	(1)	**3.25**	(7)
TargetMean	HistUS	6.83	(3)	3.67	(5)	**6.50**	(6)
	HistOS	**6.42**	(6)	3.67	(5)	6.58	(3)

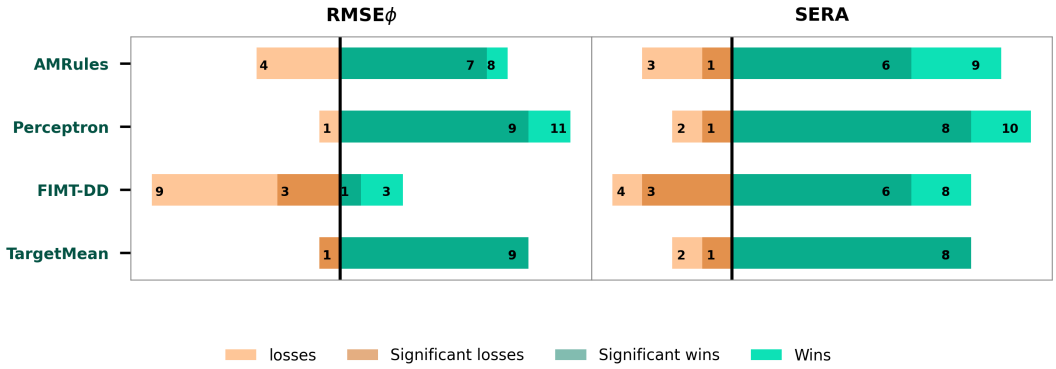
In the undersampling scenario, ChebyUS generally outperforms HistUS in terms of $RMSE_\phi$, achieving lower average ranks and a higher number of significant wins across most regression algorithms. Specifically, ChebyUS demonstrates superior performance for AMRules (5 wins), Perceptron (6 wins), and TargetMean (7 wins) when compared to HistUS. An exception is observed in the case of FIMT-DD, where HistUS achieves a better results for $RMSE_\phi$ (6 wins vs. 2 for ChebyUS). These findings suggest that Chebyshev-based undersampling is more effective in reducing prediction errors for rare cases. Regarding $SERA$, the performance of ChebyUS and HistUS varies across models. While ChebyUS exhibits better results for AMRules and TargetMean, HistUS demonstrates superior performance for FIMT-DD and Perceptron.

In the oversampling scenario, HistOS exhibits superior performance across both $RMSE_\phi$ and $SERA$ in almost all cases. The only exception is observed in the comparison for FIMT-DD using $RMSE_\phi$, where ChebyOS achieves slightly better results than HistOS.

Overall, the findings indicate that Chebyshev-based undersampling (ChebyUS) is particularly effective for rare case prediction, consistently outperforming the histogram-based method in terms of $RMSE_\phi$. However, the effectiveness based on $SERA$ is model-dependent, with HistOS demonstrating competitive results in certain cases. In oversampling settings, the histogram-based method (HistOS) proves to be more efficient in rare case prediction across both evaluation metrics.



(A) ChebyUS *vs* HistUS



(B) ChebyOS *vs* HistOS

FIGURE 4.8: Comparison of wins and losses, along with significant outcomes at a 95% confidence level, derived from Chebyshev-based sampling *versus* histogram-based undersampling using $RMSE_\phi$, *SERA*.

TABLE 4.5: Comparison of Chebyshev-based and histogram-based Undersampling strategies: average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ and *SERA*.

Algorithm	**Approach**	$RMSE_\phi$		*SERA*	
		avg_rank	#w	avg_rank	#w
AMRules	ChebyUS	**2.83**	(5)	**4.92**	(5)
	HistUS	3.08	(2)	5.42	(5)
Perceptron	ChebyUS	**4.21**	(6)	4.42	(3)
	HistUS	4.46	(5)	4.42	(5)
FIMT-DD	ChebyUS	4.71	(2)	4.83	(3)
	HistUS	**3.79**	(6)	**4.50**	(8)
TargetMean	ChebyUS	**5.92**	(7)	**3.29**	(7)
	HistUS	7.00	(1)	4.21	(2)

TABLE 4.6: Comparison of Chebyshev-based and histogram-based oversampling strategies: average rank and number of significant wins (#w) at 95% confidence level, according to $RMSE_\phi$ and $SERA$.

Algorithm	**Approach**	$RMSE_\phi$		$SERA$	
		avg_rank	#w	avg_rank	#w
AMRules	ChebyOS	3.50	(0)	6.00	(1)
	HistOS	**3.00**	(7)	**4.50**	(6)
Perceptron	ChebyOS	5.42	(0)	5.08	(1)
	HistOS	**4.33**	(9)	**3.08**	(8)
FIMT-DD	ChebyOS	**2.92**	(3)	5.08	(3)
	HistOS	3.50	(1)	**3.58**	(6)
TargetMean	ChebyOS	7.25	(1)	5.33	(1)
	HistOS	**6.08**	(9)	**3.33**	(8)

4.3 Limitations

The histogram-based approaches introduced in this chapter, while effective in dynamically identifying rare cases across arbitrary regions of the target distribution, exhibit several constraints inherent to their design. First, the reliance on incremental histogram construction introduces sensitivity to initial data segments during the warming phase, where sparse or unrepresentative early samples may skew bin boundaries and density estimates. This effect is exacerbated in non-stationary environments with abrupt concept drift, as the online histogram's adaptive binning mechanism—though designed for gradual updates—may lag behind sudden distributional shifts. Second, the probability function's dependence on parameters β (sampling steepness) and α (oversampling decay) necessitates empirical tuning, as their optimal values vary across datasets and streaming conditions, limiting plug-and-play applicability. Computational overhead also arises from maintaining and querying the histogram during high-velocity streams, particularly for multi-dimensional feature spaces where joint distributions between features and targets would require multivariate histograms—a complexity avoided in this work. Finally, the evaluation framework inherits limitations from the $\phi()$ relevance function's dependency on equal-width histogram binning, which may misalign with domain-specific rarity definitions that require nonlinear or context-aware relevance mappings.

Chapter 5

Conclusions and Future Work

The proliferation of streaming data in domains such as IoT networks, financial transactions, and environmental monitoring has necessitated robust solutions for imbalanced regression tasks, where accurate prediction of rare but critical target values remains paramount. This thesis addressed this challenge through the development of adaptive sampling strategies tailored for non-stationary data streams, proposing two complementary paradigms: Chebyshev-based methods leveraging statistical bounds for extreme value identification, and histogram-based approaches enabling dynamic rare case detection across evolving distributions. Experimental validation across synthetic and real-world benchmarks demonstrated these methods' capacity to enhance model focus on underrepresented regions while maintaining computational efficiencycritical for real-time applications. This chapter synthesizes the key findings of this research, revisits the original research questions with evidence-based answers, and outlines pragmatic implications for practitioners. Finally, it identifies limitations of the current approaches and proposes directions for extending this work to address emerging challenges in streaming imbalanced learning.

5.1 Conclusions

This thesis addressed the critical challenge of handling imbalanced regression data streams, where rare instances are often of greater importance than frequent ones. While most prior research in the field of data stream mining has focused on classification tasks, this work extended

the investigation to the regression setting, where the target variable is continuous, and rare cases can appear at any point in the distribution.

We proposed two distinct strategies for mitigating the effects of imbalance in regression data streams: (i) Chebyshev-based sampling methods, including undersampling (ChebyUS) and oversampling (ChebyOS), which leverage Chebyshev's inequality to identify rare cases based on their deviation from the mean, and (ii) histogram-based sampling methods, including undersampling (HistUS) and oversampling (HistOS), which dynamically adjust to the evolving nature of the data stream by maintaining an online histogram of target values. These approaches were designed to ensure that rare instances are adequately represented in the training process, thus improving model performance in such cases.

Extensive empirical evaluations were conducted using synthetic and real-world datasets, integrating the proposed methods into four widely used regression algorithms: AMRules, Perceptron, FIMT-DD, and TargetMean. The learners were compared against their baseline versions, with no sampling, based on $RMSE$ and its variant which is called $RMSE_\phi$, and also *sera* evaluation functions.

For Chebyshev-based methods, We explored the impact of our sampling strategies on different types of rare extreme values of the target variable in each data set: one set contained both rare extreme values cases, one set contained low rare extreme values cases, and one set contained high rare extreme values cases. Regardless of the type of the rare extreme values cases chosen and irrespective of the learner applied, the experimental results showed a reduction of the prediction errors over the rare cases. The results also suggested that the proposed sampling methods can be more efficient when considering only one type of rare extreme values.

For Histogram-based approaches, our study highlights the role of two key parameters, β and α, which influence the probability estimation and oversampling rate in our histogram-based methods. A deeper exploration of their impact revealed that tuning these parameters can significantly enhance the effectiveness of rare case detection and model training.

The results demonstrated that histogram-based methods provided a more flexible and effective solution for capturing rare instances across the entire distribution, whereas Chebyshev-based approaches were more suited for cases where rare instances are confined to the distribution tails.

Despite these promising results, several areas remain open for future research. One critical

direction is to extend our methods to handle concept drift, a common challenge where data streams are evolving. In this case, computation of the descriptive statistics of the data (i.e. μ and σ) in ChebyUS and ChebyOS may not be accurate if all examples from the beginning are considered and this may have an impact on calculating Chebyshev's probability. For HistUS and HistOS approaches, while the online histogram inherently adapts to changes in data distribution, a more formalized drift detection mechanism could further improve performance under non-stationary conditions.

The second area that can be improved is the investigation of adaptive parameter tuning in the algorithms. In more detail, the optimum value for the second chance probability parameter in Algorithm 3.1 may also be better explored. Also, α and β could be dynamically optimized during the learning process in the HistUS and HistOS rather than being set empirically. Lastly, validating the proposed methods on a broader range of real-world applications, such as predictive maintenance, anomaly detection, and personalized recommendation systems, would provide further insights into their practical utility.

In conclusion, this thesis contributes to the field of imbalanced regression in data streams by introducing novel sampling strategies that improve the learning process for rare cases. The proposed methods serve as a foundation for future research, enabling more accurate and reliable predictive models in dynamic and evolving environments.

5.2 Answering Research Questions

In this section, we answer the research questions posed in the Introduction chapter to evaluate the effectiveness of the proposed methods in addressing imbalanced regression problems in data streams. The research questions and the answers are as follows:

- **RQ1**: How do probability distributions make traditional learning methods unsuitable for learning from imbalanced data streams?

  Traditional learning methods struggle with imbalanced datasets primarily due to the unequal distribution of target values, which leads to biased models that prioritize frequent instances, resulting in poor generalization for rare instances. This issue has been addressed by many existing techniques, such as resampling, cost-sensitive learning, and ensemble methods that have been discussed in chapter 2. However, these methods are not directly

applicable to streaming environments due to the nature of data streams. In static datasets (where data that is not actively being transmitted or processed and is stored in a stable, non-moving state), the entire dataset is available for training, and for example, resampling techniques can be applied globally, ensuring that rare cases are sufficiently represented in the training process. However, in a streaming context, data arrives sequentially and the inability to store and revisit past data prevents batch resampling methods from being applied effectively in data streams. By leveraging Chebyshev's inequality and online histogram techniques, the proposed strategies can effectively capture rare cases in real-time, ensuring that the model is trained on a balanced representation of the target variable.

- **RQ2**: How do different sets of rare examples impact the performance of the proposed methods?

  The impact of different sets of rare examples on the performance of the proposed methods was analyzed by varying the threshold of the relevance function, which defines the rarity of instances. Experimental results demonstrated that as the threshold increased, the models became more focused on rare cases, improving accuracy in the prediction of their values but sometimes at the cost of overall predictive accuracy.

- **RQ3**: How does the choice of sampling strategy (Chebyshev-based vs. Histogram-based) affects the predictive performance of regression models?

  The effectiveness of a sampling strategy in imbalanced regression largely depends on the distribution of rare cases within the target variable. When rare instances are concentrated at the extremes of the distributioneither in the lower or upper tailsboth Chebyshev-based and Histogram-based approaches can be applied. In such scenarios, Chebyshev-based undersampling tends to achieve superior performance in predicting rare cases, as evidenced by its lower errors in the $RMSE_\phi$ and $SERA$ evaluation metrics. The reliance on Chebyshevs inequality allows this method to effectively differentiate extreme values from frequent instances, leading to more precise rare-instance modeling.

  However, for oversampling strategies, the Histogram-based approach demonstrates a stronger performance, particularly when evaluated using the $SERA$ metric. Nevertheless, when considering $RMSE_\phi$, both approaches remain competitive, with no significant loss of predictive accuracy in the Chebyshev-based method.

  While Chebyshev-based techniques are well-suited for cases where rare instances are confined to the tails, their applicability is inherently limited when rare values appear

throughout the distribution. If rare cases are not restricted to the extremes, Histogram-based methods become the only viable solution, as they dynamically adapt to the data distribution and effectively capture rare regions irrespective of their position within the target space. This flexibility makes Histogram-based approaches particularly advantageous in real-world applications where the location of rare instances is not predetermined and may shift over time.

- **RQ4**: In general, are the proposed methods effective in coping with imbalanced regression problems?

  The effectiveness of the proposed methods in addressing imbalanced regression problems was evaluated through extensive experiments on multiple datasets. The results showed that both Chebyshev-based and histogram-based strategies significantly improved the prediction of rare values compared to baseline models that did not incorporate resampling. The proposed approaches consistently outperformed baseline learning methods, particularly in scenarios where rare target values carried higher relevance. Additionally, the ability to operate in real-time while maintaining computational efficiency demonstrated the applicability of these methods to real-world streaming environments. The findings confirm that the proposed strategies successfully mitigate the challenges posed by imbalanced regression in data streams, offering a robust solution for improving predictive accuracy in rare cases.

5.3 Key Results

The experimental evaluation conducted in this thesis demonstrates the effectiveness of the proposed sampling strategies in addressing imbalanced regression. The results highlight that both Chebyshev-based and Histogram-based approaches significantly improve the predictive performance of regression models when rare instances are underrepresented.

One of the key findings is that Chebyshev-based undersampling achieves superior results when rare instances are located at the extremes of the target distribution. This approach effectively improves predictive accuracy for rare values, as evidenced by its performance in the $RMSE_\phi$ and $SERA$ evaluation metrics. However, for oversampling, the histogram-based approach proves to be more effective in reducing errors across the entire target range, making it more suitable when rare cases are distributed throughout the domain.

Further analysis reveals that Histogram-based sampling is the only viable solution when rare instances are not confined to the tails of the distribution. Unlike Chebyshev-based techniques, which rely on statistical assumptions about extreme values, Histogram-based methods dynamically adjust to evolving distributions, demonstrating higher adaptability across different datasets.

Additionally, results show that Chebyshev-based methods are computationally more efficient due to their reliance on statistical bounds, whereas Histogram-based methods, while slightly more computationally demanding, provide greater flexibility in real-world applications. The experimental comparisons also indicate that the choice of relevance function significantly impacts model performance, influencing the ability to correctly identify rare instances.

These findings collectively confirm that the proposed methods successfully enhance learning from imbalanced regression data streams, providing a structured approach to handling rare cases in predictive modeling.

5.4 Future Work

Despite these promising results, several areas remain open for future research. One critical direction is to extend our methods to handle concept drift, a common challenge where data streams are evolving. In this case, computation of the descriptive statistics of the data (i.e. μ and σ) in ChebyUS and ChebyOS may not be accurate if all examples from the beginning are considered and this may have an impact on calculating Chebyshev's probability. For HistUS and HistOS approaches, while the online histogram inherently adapts to changes in data distribution, a more formalized drift detection mechanism could further improve performance under non-stationary conditions.

The second area that can be improved is the investigation of adaptive parameter tuning in the algorithms. In more detail, the optimum value for the second chance probability parameter in Algorithm 3.1 may also be better explored. Also, α and β could be dynamically optimized during the learning process in the HistUS and HistOS rather than being set empirically. Lastly, validating the proposed methods on a broader range of real-world applications, such as predictive maintenance, anomaly detection, and personalized recommendation systems, would provide further insights into their practical utility.

Another direction for future research is the comparative analysis of the time complexity of histogram-based and Chebyshev-based sampling approaches. While both strategies aim to enhance predictive performance in imbalanced regression data streams, their computational efficiency remains an open question. Investigating their scalability and processing overhead in real-time streaming environments particularly for Histogram-based methods can provide valuable insights into their practical applicability. Additionally, exploring optimizations to reduce computational costs while maintaining predictive accuracy could further improve their usability. Such an analysis would contribute to a deeper understanding of the trade-offs between accuracy and efficiency in data stream mining.

In conclusion, this thesis contributes to the field of imbalanced regression in data streams by introducing novel sampling strategies that improve the learning process for rare cases. The proposed methods serve as a foundation for future research, enabling more accurate and reliable predictive models in dynamic and evolving environments.

Bibliography

[1] L. Hernandez Aros, L. X. Bustamante Molano, F. Gutierrez-Portela, J. J. Moreno Hernandez, and M. S. Rodríguez Barrero, "Financial fraud detection through the application of machine learning techniques: a literature review," *Humanities and Social Sciences Communications*, vol. 11, no. 1, pp. 1–22, 2024. [Cited on page 1.]

[2] A. Gupta, N. Chaithra, J. Jha, A. Sayal, V. Gupta, and M. Memoria, "Machine learning algorithms for disease diagnosis using medical records: A comparative analysis," in *2023 4th International Conference on Intelligent Engineering and Management (ICIEM)*. IEEE, 2023, pp. 1–6. [Cited on page 1.]

[3] E. Snieder, K. Abogadil, and U. T. Khan, "Resampling and ensemble techniques for improving ann-based high-flow forecast accuracy," *Hydrology and Earth System Sciences*, vol. 25, no. 5, pp. 2543–2566, 2021. [Cited on page 1.]

[4] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in artificial intelligence*, vol. 5, no. 4, pp. 221–232, 2016. [Cited on page 1.]

[5] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014. [Cited on pages 2 and 11.]

[6] A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer, "Machine learningfor data streams: with practical examples in moa," 2018. [Cited on pages 2 and 11.]

[7] J. Montiel, J. Read, A. Bifet, and T. Abdessalem, "Scikit-multiflow: A multi-output streaming framework," *Journal of Machine Learning Research*, vol. 19, no. 72, pp. 1–5, 2018. [Online]. Available: http://jmlr.org/papers/v19/18-251.html [Cited on pages 2 and 11.]

[8] J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Mach. Learn.*, vol. 90, no. 3, pp. 317–346, 2013. [Cited on pages 2, 40, and 69.]

[9] T. Kolajo, O. Daramola, and A. Adebiyi, "Big data stream analysis: a systematic literature review," *Journal of Big Data*, vol. 6, no. 1, p. 47, 2019. [Cited on pages 2 and 12.]

[10] R. P. Ribeiro and N. Moniz, "Imbalanced regression and extreme value prediction," *Mach. Learn.*, vol. 109, no. 9-10, pp. 1803–1835, 2020. [Cited on pages xiii, 2, 10, 17, 37, 40, 51, 61, and 69.]

[11] E. Aminian, R. P. Ribeiro, and J. Gama, "Chebyshev approaches for imbalanced data streams regression models," *Data Mining and Knowledge Discovery*, vol. 35, pp. 2389–2466, 2021. [Cited on pages 10 and 69.]

[12] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Comput. Surv.*, vol. 49, 2016. [Cited on pages 10 and 46.]

[13] A. Bifet and R. Gavalda, "Learning from time-changing data with adaptive windowing," in *Proceedings of the 2007 SIAM international conference on data mining*. SIAM, 2007, pp. 443–448. [Cited on page 11.]

[14] W. Ng and M. Dash, *Discovery of Frequent Patterns in Transactional Data Streams*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–30. [Cited on page 11.]

[15] C. C. Aggarwal, *Data streams: models and algorithms*. Springer Science & Business Media, 2007, vol. 31. [Online]. Available: https://link.springer.com/book/10.1007/978-0-387-47534-9 [Cited on page 11.]

[16] M. Bahri, "Improving iot data stream analytics using summarization techniques," Ph.D. dissertation, Institut Polytechnique de Paris, 2020. [Online]. Available: https://theses.hal.science/tel-02865982v1 [Cited on page 11.]

[17] N. Oza, "Online bagging and boosting," in *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, 2005, pp. 2340–2345 Vol. 3. [Cited on page 12.]

[18] E. Ikonomovska, J. Gama, and S. Džeroski, "Learning model trees from evolving data streams," *Data mining and knowledge discovery*, vol. 23, pp. 128–168, 2011. [Cited on pages 12 and 69.]

[19] ——, "Online tree-based ensembles and option trees for regression on evolving data streams," *Neurocomputing*, vol. 150, pp. 458–470, 2015. [Cited on page 13.]

[20] H. M. Gomes, J. P. Barddal, L. E. B. Ferreira, and A. Bifet, "Adaptive random forests for data stream regression," in *ESANN 2018 proceedings*, 2018. [Online]. Available: https://www.ppgia.pucpr.br/~jean.barddal/assets/pdf/arf_regression.pdf [Cited on page 13.]

[21] A. Osojnik, P. Panov, and S. Džeroski, "Incremental predictive clustering trees for online semi-supervised multi-target regression," *Machine Learning*, vol. 109, no. 11, pp. 2121–2139, 2020. [Cited on page 13.]

[22] ——, "isoup-symrf: Symbolic feature ranking with random forests in online multi-target regression," in *International Conference on Discovery Science*. Springer, 2023, pp. 48–63. [Cited on page 14.]

[23] T. Stepišnik, A. Osojnik, S. Džeroski, and D. Kocev, "Option predictive clustering trees for multi-target regression," *Computer Science and Information Systems*, vol. 17, no. 2, pp. 459–486, 2020. [Cited on page 14.]

[24] B. Veloso, J. Gama, B. Malheiro, and J. Vinagre, "Hyperparameter self-tuning for data streams," *Information Fusion*, vol. 76, pp. 75–86, 2021. [Cited on page 14.]

[25] B. Stevanoski, A. Kostovska, P. Panov, and S. Džeroski, "Change detection and adaptation in multi-target regression on data streams," *Machine Learning*, pp. 1–38, 2024. [Cited on page 14.]

[26] M. Garofalakis, J. Gehrke, and R. Rastogi, "Querying and mining data streams: you only get one look a tutorial," in *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, 2002, pp. 635–635. [Cited on page 15.]

[27] M. Bahri, A. Bifet, J. Gama, H. M. Gomes, and S. Maniu, "Data stream analysis: Foundations, major tasks and tools," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 3, p. e1405, 2021. [Cited on page 15.]

[28] N. Anupama and S. Jena, "A novel approach using incremental oversampling for data stream mining," *Evolving Systems*, vol. 10, pp. 351–362, 2019. [Cited on page 15.]

[29] S. Ren, W. Zhu, B. Liao, Z. Li, P. Wang, K. Li, M. Chen, and Z. Li, "Selection-based resampling ensemble algorithm for nonstationary imbalanced stream data learning," *Knowledge-Based Systems*, vol. 163, pp. 705–722, 2019. [Cited on page 15.]

[30] J.-B. Wang, C.-A. Zou, and G.-H. Fu, "Awsmote: An svm-based adaptive weighted smote for class-imbalance learning," *Scientific Programming*, vol. 2021, no. 1, p. 9947621, 2021. [Cited on page 15.]

[31] A. Bernardo, H. M. Gomes, J. Montiel, B. Pfahringer, A. Bifet, and E. Della Valle, "C-smote: Continuous synthetic minority oversampling for evolving data streams," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 483–492. [Cited on page 15.]

[32] A. Bernardo and E. Della Valle, "Vfc-smote: very fast continuous synthetic minority oversampling for evolving data streams," *Data Mining and Knowledge Discovery*, vol. 35, no. 6, pp. 2679–2713, 2021. [Cited on page 16.]

[33] M. Steininger, K. Kobs, P. Davidson, A. Krause, and A. Hotho, "Density-based weighting for imbalanced regression," *Machine Learning*, vol. 110, pp. 2187–2211, 2021. [Cited on pages 16 and 24.]

[34] R. P. A. Ribeiro, "Utility-based regression," Ph.D. dissertation, Department of Computer Science - Faculty of Sciences University of Porto, 2011. [Online]. Available: https://www.dcc.fc.up.pt/~rpribeiro/publ/rpribeiroPhD11.pdf [Cited on pages 16, 17, 21, 26, and 27.]

[35] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, "Smote for regression," in *Portuguese conference on artificial intelligence*. Springer, 2013, pp. 378–389. [Cited on pages 16 and 19.]

[36] P. Branco, L. Torgo, and R. P. Ribeiro, "Pre-processing approaches for imbalanced distributions in regression," *Neurocomputing*, vol. 343, pp. 76–99, 2019. [Cited on pages 17, 18, and 22.]

[37] N. Moniz, R. Ribeiro, V. Cerqueira, and N. Chawla, "Smoteboost for regression: Improving the prediction of extreme values," in *2018 IEEE 5th international conference on data science and advanced analytics (DSAA)*. IEEE, 2018, pp. 150–159. [Cited on pages 17 and 27.]

[38] T. R. Hoens and N. V. Chawla, "Imbalanced datasets: from sampling to classifiers," *Imbalanced learning: Foundations, algorithms, and applications*, pp. 43–59, 2013. [Cited on page 17.]

[39] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004. [Cited on page 18.]

[40] M. Kubat, S. Matwin *et al.*, "Addressing the curse of imbalanced training sets: one-sided selection," in *Icml*, vol. 97. Nashville, USA, 1997, pp. 179–186. [Online]. Available: https://sci2s.ugr.es/keel/pdf/algorithm/congreso/kubat97addressing.pdf [Cited on page 18.]

[41] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority oversampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002. [Cited on pages 18 and 19.]

[42] P. R. Bal and S. Kumar, "Cross project software defect prediction using extreme learning machine: An ensemble based study." in *ICSOFT*, 2018, pp. 354–361. [Cited on page 20.]

[43] P. Branco, L. Torgo, and R. P. R. Ribeiro, "Smogn: a pre-processing approach for imbalanced regression," in *First international workshop on learning with imbalanced domains: Theory and applications*. PMLR, 2017, pp. 36–50. [Online]. Available: https://proceedings.mlr.press/v74/branco17a/branco17a.pdf [Cited on page 20.]

[44] L. Camacho, G. Douzas, and F. Bacao, "Geometric smote for regression," *Expert Systems with Applications*, vol. 193, p. 116387, 2022. [Cited on page 21.]

[45] G. Douzas and F. Bacao, "Geometric smote a geometrically enhanced drop-in replacement for smote," *Information sciences*, vol. 501, pp. 118–135, 2019. [Cited on page 21.]

[46] L. Camacho and F. Bacao, "Wsmoter: a novel approach for imbalanced regression," *Applied Intelligence*, vol. 54, no. 19, pp. 8789–8799, 2024. [Cited on pages 21 and 25.]

[47] X. Y. Song, N. Dao, and P. Branco, "Distsmogn: Distributed smogn for imbalanced regression problems," in *Fourth International Workshop on Learning with Imbalanced Domains: Theory and Applications*. PMLR, 2022, pp. 38–52. [Online]. Available: https://proceedings.mlr.press/v183/song22a.html [Cited on page 22.]

[48] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, F. Herrera, A. Fernández, S. García, M. Galar, R. C. Prati *et al.*, "Cost-sensitive learning," *Learning from imbalanced data sets*, pp. 63–78, 2018. [Cited on page 24.]

[49] L. Torgo and R. Ribeiro, "Predicting outliers," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2003, pp. 447–458. [Cited on page 25.]

[50] Y. Yang, K. Zha, Y. Chen, H. Wang, and D. Katabi, "Delving into deep imbalanced regression," in *International conference on machine learning*. PMLR, 2021, pp. 11 842–11 851. [Cited on page 25.]

[51] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997. [Cited on page 26.]

[52] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging, boosting, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 42, no. 4, pp. 463–484, 2012. [Cited on page 26.]

[53] O. I. Orhobor, N. F. Grinberg, L. N. Soldatova, and R. D. King, "Imbalanced regression using regressor-classifier ensembles," *Machine Learning*, vol. 112, no. 4, pp. 1365–1387, 2023. [Cited on page 26.]

[54] P. Branco, L. Torgo, and R. P. Ribeiro, "Rebagg: Resampled bagging for imbalanced regression," in *Proceedings of the Second International Workshop on Learning with Imbalanced Domains: Theory and Applications*, ser. Proceedings of Machine Learning Research, L. Torgo, S. Matwin, N. Japkowicz, B. Krawczyk, N. Moniz, and P. Branco, Eds., vol. 94. PMLR, 10 Sep 2018, pp. 67–81. [Online]. Available: https://proceedings.mlr.press/v94/branco18a.html [Cited on page 26.]

[55] Y. Huang, D.-R. Liu, S.-J. Lee, C.-H. Hsu, and Y.-G. Liu, "A boosting resampling method for regression based on a conditional variational autoencoder," *Information Sciences*, vol. 590, pp. 90–105, 2022. [Cited on page 27.]

[56] T. Finch, "Incremental calculation of weighted mean and variance.(2009)," *Available also from: http://nfs-uxsup. csx. cam. ac. uk/˜ fanf2/hermes/doc/antiforgery/stats. pdf*, 2009. [Cited on page 35.]

[57] L. Torgo and R. Ribeiro, "Utility-based regression," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2007, pp. 597–604. [Cited on page 37.]

[58] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml [Cited on pages 38, 61, and 68.]

[59] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework." *Journal of Multiple-Valued Logic & Soft Computing*, vol. 17, 2011. [Cited on page 38.]

[60] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: massive online analysis," *J. Mach. Learn. Res.*, vol. 11, pp. 1601–1604, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?id=1859903 [Cited on pages 38 and 69.]

[61] A. Maglie, *ReactiveX and RxJava*, 11 2016, pp. 1–9. [Cited on page 39.]

[62] H. D. Block, "Neurocomputing: Foundations of research," J. A. Anderson and E. Rosenfeld, Eds. Cambridge, MA, USA: MIT Press, 1988, ch. The Perceptron: A Model for Brain Functioning. I, pp. 135–150. [Online]. Available: http://dl.acm.org/citation.cfm?id=65669.104392 [Cited on pages 40 and 69.]

[63] E. Ikonomovska, J. Gama, and S. Dzeroski, "Learning model trees from evolving data streams," *Data Min. Knowl. Discov.*, vol. 23, no. 1, pp. 128–168, 2011. [Cited on page 40.]

[64] J. Duarte, J. Gama, and A. Bifet, "Adaptive model rules from high-speed data streams," *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 3, pp. 30:1–30:22, 2016. [Cited on pages 40 and 69.]

[65] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006. [Online]. Available: https://dl.acm.org/doi/10.5555/1248547.1248548 [Cited on pages 41 and 70.]

[66] J. Gama and C. Pinto, "Discretization from data streams: applications to histograms and data mining," in *Proceedings of the 2006 ACM symposium on Applied computing*, 2006, pp. 662–667. [Cited on page 56.]

Appendix

Appendix A

The results from the Chebyshev Approach

This appendix provides supplementary material to support the experimental evaluation presented in the main body of the thesis. It includes detailed table results of the experiments conducted to evaluate the performance of the proposed sampling strategies. Specifically, the tables present the results of sensitivity analyses, varying key parameters such as the relevance threshold (thr_ϕ), and compare the performance of the proposed methods against baseline models. The tables are organized to facilitate a deeper understanding of the experimental outcomes, including the average and standard deviation of error metrics such as $RMSE_\phi$ and $RMSE$, as well as the statistical significance of the results. These findings provide additional insights into the effectiveness of the proposed methods and their robustness across different datasets and experimental conditions.

A.1 More about the data sets

Since we have used short names to refer to the data sets in the main text and make it easier for the readers to find the data sets, we report their complete name, source, and the number of their attributes in Table A.1.

*https://sci2s.ugr.es/keel/datasets.php
†https://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html
‡https://archive.ics.uci.edu/ml/datasets.php
§https://www.kaggle.com/tsaustin/us-used-car-sales-data

TABLE A.1: Data Set Characteristics

Data set	Data Set Full Name	#Attributes	#Examples
puma32H*	Pumadyn	32	8,192
cpusum†	Computer Activity	21	8,192
elevator[1]	Elevators	18	16,599
bike‡	Bike Sharing	16	17,379
energy[3]	Appliances Energy Prediction	29	19,735
calhousing[1]	California Housing	8	20,639
gasemission[3]	Gas Turbine CO and NOx Emission	11	36,733
mv[2]	MV Artificial Domain	10	40,768
fried[2]	Friedman Artificial Domain	10	40,768
pollution[3]	Beijing PM2.5 Data	13	41,757
car price§	US Used Car Sales Data	12	122,144
query[3]	Query Analytics Workloads	8	199,843
GPU[3]	SGEMM GPU Kernel Performance	18	241,600
3d[3]	3D Road Network	4	434,873

A.2 Results of the experiments: sensitivity analysis varying ϕ.

Tables A.4 - A.9 show the results of the paired comparisons for the four learner models trained by our (Under/Over)-sampling strategies, ChebyUS and ChebyOS, against their baseline versions. The symbols $\triangleright$ and $\triangleleft$ indicate that the ChebyUS or ChebyOS method is significantly better or worse, respectively, compared to the baseline. In each experiment, the value obtained by the error function (i.e. $RMSE_\phi$ or $RMSE$) for prediction of the learner mentioned in the column header over the data set specified in the corresponding row has been reported. The numbers inside the cells are the average and corresponding standard deviation of the results on ten rounds of experiments. We report the statistical significance (level 95%) of the difference between each pair using the two symbols $\triangleright$ and $\triangleleft$, pointing to the significantly better method. As we have reported results for different levels of thr_ϕ, information about the number/percentage of the rare cases observed in each data set presented in Table A.2 and Table A.3.

TABLE A.2: Data sets characteristics: number of cases (#cases) and number of rare cases (#rare cases) with $thr_\phi = 0.9$ that are low extremes (i.e., below the median), high extremes (i.e., above the median), or both.

Data set	#cases	#rare cases ($thr_\phi = 0.9$)		
		Low extremes	High extremes	Both extremes
puma32H	8,192	266 (3.25%)	273 (3.33%)	539 (6.58%)
cpusum	8,192	621 (7.58%)	433 (5.29%)	1,054 (12.87%)
elevator	16,599	55 (0.33%)	2,025 (12.20%)	2,080 (12.53%)
bike	17,379	3,623 (20.85%)	981 (5.64%)	4,604 (26.49%)
energy	19,735	9 (0.05%)	2,442 (12.37%)	2,451 (12.42%)
calhousing	20,639	127 (0.62%)	1,515 (7.34%)	1,642 (7.96%)
gasemission	36,733	47 (0.13%)	1,434 (3.90%)	1,481 (4.03%)
mv	40,768	537 (1.32%)	5,108 (12.53%)	5,645 (13.85%)
fried	40,768	278 (0.68%)	313 (0.77%)	591 (1.45%)
polution	41,757	4,467 (10.70%)	2,929 (7.01%)	7,396 (17.71%)
car price	122,144	12,499 (10.23%)	11,849 (9.70%)	24,348 (19.93%)
query	199,843	3,084 (1.54%)	1,356 (0.68%)	4,440 (2.22%)
GPU	241,600	8,014 (3.32%)	36,065 (14.93%)	44,079 (18.25%)
3d	434,873	3 (0.00%)	23,182 (5.33%)	23,185 (5.33%)

TABLE A.3: Data sets characteristics: number of cases (#cases) and number of rare cases (#rare cases) with $thr_\phi = 1.0$ that are low extremes (i.e., below the median), high extremes (i.e., above the median), or both.

Data set	#cases	#rare cases ($thr_\phi = 1.0$)		
		Low extremes	High extremes	Both extremes
puma32H	8,192	57 (0.7%)	68 (0.83%)	125 (1.52%)
cpusum	8,192	469 (5.73%)	61 (0.74%)	530 (6.47%)
elevator	16,599	13 (0.08%)	1,762 (10.62%)	1,775 (10.70%)
bike	17,379	158 (0.91%)	509 (2.93%)	667 (3.84%)
energy	19,735	9 (0.05%)	2,208 (11.19%)	2,217 (11.24%)
calhousing	20,639	4 (0.02%)	1,072 (5.19%)	1,076 (5.21%)
gasemission	36,733	5 (0.01%)	933 (2.54%)	938 (2.55%)
mv	40,768	2 (0.00%)	1 (0.00%)	3 (0.00%)
fried	40,768	7 (0.02%)	20 (0.05%)	27 (0.07%)
polution	41,757	2 (0.00%)	1,797 (4.30%)	1,799 (4.30%)
car price	122,144	31 (0.03%)	8,405 (6.88%)	8,436 (6.91%)
query	199,843	868 (0.43%)	147 (0.07%)	1,015 (0.50%)
GPU	241,600	1 (0.00%)	26,857 (11.12%)	26,858 (11.12%)
3d	434,873	1 (0.00%)	11,666 (2.68%)	11,667 (2.68%)

TABLE A.4: $RMSE$ and $RMSE_\phi$ estimates considering all the observations: $thr_\phi = 0$.

$RMSE_\phi$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.02 ± 0.0	0.02 ± 0.0	0.01 ± 0.0	0.01 ± 0.0	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.0
cpusum	8.5 ± 0.59 ▷	$\mathbf{6.92 \pm 0.32}$	7.45 ± 0.6 ▷	$\mathbf{6.11 \pm 0.4}$	10.72 ± 1.24 ▷	$\mathbf{9.96 \pm 1.31}$	11.13 ± 0.34 ▷	$\mathbf{10.48 \pm 0.3}$
elevator	$\mathbf{0.02 \pm 0.02}$	0.03 ± 0.03	$\mathbf{0.08 \pm 0.05}$ ◁	$21 \times 10^3 \pm 23 \times 10^3$	$\mathbf{0.09 \pm 0.01}$ ◁	$33 \times 10^3 \pm 19 \times 10^2$	$\mathbf{0.08 \pm 0.0}$ ◁	$28 \times 10^3 \pm 12 \times 10^2$
bike	$\mathbf{4.92 \pm 0.41}$ ◁	5.01 ± 0.43	$\mathbf{23.34 \pm 4.16}$ ◁	23.82 ± 3.59	$\mathbf{80.9 \pm 14.22}$ ◁	81.97 ± 14.47	$\mathbf{86.61 \pm 3.7}$ ◁	87.8 ± 3.75
energy	86.05 ± 1.45 ▷	$\mathbf{79.03 \pm 1.38}$	87.26 ± 0.35 ▷	$\mathbf{80.28 \pm 0.34}$	89.74 ± 0.66 ▷	$\mathbf{82.5 \pm 0.57}$	89.81 ± 0.27 ▷	$\mathbf{82.86 \pm 0.18}$
calhousing	$49 \times 10^3 \pm 823.83$ ▷	$46 \times 10^3 \pm \mathbf{765.91}$	$48 \times 10^3 \pm 598.05$ ▷	$45 \times 10^3 \pm \mathbf{638.8}$	$61 \times 10^3 \pm 47 \times 10^2$ ▷	$59 \times 10^3 \pm 48 \times 10^2$	$63 \times 10^3 \pm 13 \times 10^2$ ▷	$60 \times 10^3 \pm 14 \times 10^2$
gasemission	5.16 ± 0.06 ▷	$\mathbf{4.67 \pm 0.07}$	4.91 ± 0.13 ▷	$\mathbf{4.41 \pm 0.14}$	5.76 ± 0.5 ▷	$\mathbf{5.37 \pm 0.55}$	5.98 ± 0.17 ▷	$\mathbf{5.65 \pm 0.17}$
mv	$\mathbf{3.34 \pm 0.02}$ ◁	3.36 ± 0.02	$\mathbf{2.6 \pm 0.25}$ ◁	2.61 ± 0.26	$\mathbf{4.25 \pm 0.58}$ ◁	4.35 ± 0.61	$\mathbf{4.44 \pm 0.18}$ ◁	4.54 ± 0.18
fried	1.1 ± 0.01 ▷	$\mathbf{1.03 \pm 0.01}$	0.99 ± 0.03 ▷	$\mathbf{0.9 \pm 0.04}$	1.68 ± 0.22 ▷	$\mathbf{1.64 \pm 0.23}$	1.78 ± 0.05 ▷	$\mathbf{1.74 \pm 0.06}$
polution	65.56 ± 0.94 ▷	$\mathbf{61.7 \pm 0.79}$	60.35 ± 1.75 ▷	$\mathbf{56.44 \pm 1.86}$	64.56 ± 2.17 ▷	$\mathbf{61.8 \pm 2.58}$	65.46 ± 0.63 ▷	$\mathbf{62.73 \pm 0.79}$
car price	$40 \times 10^7 \pm 37 \times 10^7$ ◁	$83 \times 10^7 \pm 55 \times 10^7$	$36 \times 10^8 \pm 52 \times 10^8$	$16 \times 10^8 \pm 10 \times 10^8$	$83 \times 10^8 \pm 50 \times 10^7$ ▷	$24 \times 10^8 \pm 14 \times 10^7$	$71 \times 10^8 \pm 30 \times 10^7$ ▷	$20 \times 10^8 \pm 87 \times 10^6$
query	97.21 ± 0.67 ▷	$\mathbf{94.47 \pm 0.68}$	79.05 ± 5.95 ▷	$\mathbf{75.81 \pm 6.12}$	103.93 ± 10.09 ▷	$\mathbf{102.05 \pm 10.5}$	109.6 ± 2.11 ▷	$\mathbf{107.37 \pm 2.34}$
GPU	242.33 ± 0.59 ▷	$\mathbf{204.3 \pm 0.46}$	195.8 ± 23.27 ▷	$\mathbf{165.28 \pm 19.56}$	221.29 ± 25.58 ▷	$\mathbf{198.7 \pm 27.2}$	234.98 ± 7.27 ▷	$\mathbf{214.45 \pm 6.95}$
3d	13.75 ± 0.03 ▷	$\mathbf{13.24 \pm 0.02}$	11.4 ± 1.17 ▷	$\mathbf{10.86 \pm 1.18}$	11.17 ± 0.54 ▷	$\mathbf{10.66 \pm 0.56}$	10.36 ± 1.1 ▷	$\mathbf{9.35 \pm 1.43}$
Avg. Rank	4.32	3.04	3.14	2.11	6.11	4.89	6.71	5.68

$RMSE_\phi$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.02 ± 0.0	0.02 ± 0.0	$\mathbf{0.01 \pm 0.0}$ ◁	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.0
cpusum	$\mathbf{8.5 \pm 0.59}$ ◁	10.75 ± 1.65	$\mathbf{7.45 \pm 0.6}$ ◁	14.56 ± 1.03	$\mathbf{10.72 \pm 1.24}$ ◁	17.37 ± 0.68	$\mathbf{11.13 \pm 0.34}$ ◁	17.03 ± 0.48
elevator	0.02 ± 0.02 ▷	$\mathbf{0.01 \pm 0.0}$	0.08 ± 0.05 ▷	$\mathbf{0.01 \pm 0.0}$	0.09 ± 0.01 ▷	$\mathbf{0.01 \pm 0.0}$	0.08 ± 0.0 ▷	$\mathbf{0.01 \pm 0.0}$
bike	$\mathbf{4.92 \pm 0.41}$ ◁	8.52 ± 0.7	$\mathbf{23.34 \pm 4.16}$ ◁	54.58 ± 8.76	$\mathbf{80.9 \pm 14.22}$ ◁	109.98 ± 14.3	$\mathbf{86.61 \pm 3.7}$ ◁	113.29 ± 4.83
energy	86.05 ± 1.45 ▷	$\mathbf{78.12 \pm 1.41}$	87.26 ± 0.35 ▷	$\mathbf{84.97 \pm 1.96}$	89.74 ± 0.66 ▷	$\mathbf{84.65 \pm 0.93}$	89.81 ± 0.27 ▷	$\mathbf{83.04 \pm 0.35}$
calhousing	$49 \times 10^3 \pm 823.83$ ▷	$45 \times 10^3 \pm 13 \times 10^2$	$48 \times 10^3 \pm 598.05$ ▷	$47 \times 10^3 \pm \mathbf{307.21}$	$61 \times 10^3 \pm 47 \times 10^2$ ▷	$59 \times 10^3 \pm 41 \times 10^2$	$63 \times 10^3 \pm 13 \times 10^2$ ▷	$60 \times 10^3 \pm 14 \times 10^2$
gasemission	5.16 ± 0.06 ▷	$\mathbf{4.54 \pm 0.06}$	4.91 ± 0.13 ▷	$\mathbf{4.7 \pm 0.06}$	5.76 ± 0.5 ▷	$\mathbf{5.59 \pm 0.43}$	5.98 ± 0.17 ▷	$\mathbf{5.76 \pm 0.16}$
mv	$\mathbf{3.34 \pm 0.02}$ ◁	3.54 ± 0.03	$\mathbf{2.6 \pm 0.25}$ ◁	2.85 ± 0.24	$\mathbf{4.25 \pm 0.58}$ ◁	5.17 ± 0.77	$\mathbf{4.44 \pm 0.18}$ ◁	5.44 ± 0.22
fried	1.1 ± 0.01 ▷	$\mathbf{0.95 \pm 0.01}$	0.99 ± 0.03 ▷	$\mathbf{0.9 \pm 0.02}$	1.68 ± 0.22 ▷	$\mathbf{1.65 \pm 0.23}$	1.78 ± 0.05 ▷	$\mathbf{1.75 \pm 0.06}$
polution	65.56 ± 0.94 ▷	$\mathbf{64.24 \pm 0.64}$	$\mathbf{60.35 \pm 1.75}$ ◁	61.55 ± 0.95	$\mathbf{64.56 \pm 2.17}$ ◁	69.86 ± 3.27	$\mathbf{65.46 \pm 0.63}$ ◁	71.64 ± 0.76
car price	$40 \times 10^7 \pm 37 \times 10^7$ ◁	$16 \times 10^8 \pm 11 \times 10^8$	$36 \times 10^8 \pm 52 \times 10^8$	$18 \times 10^8 \pm 15 \times 10^7$	$83 \times 10^8 \pm 50 \times 10^7$ ▷	$14 \times 10^8 \pm 89 \times 10^6$	$71 \times 10^8 \pm 30 \times 10^7$ ▷	$12 \times 10^8 \pm 54 \times 10^6$
query	97.21 ± 0.67 ▷	$\mathbf{91.43 \pm 0.68}$	79.05 ± 5.95 ▷	$\mathbf{74.99 \pm 5.21}$	103.93 ± 10.09 ▷	$\mathbf{102.1 \pm 10.46}$	109.6 ± 2.11 ▷	$\mathbf{107.06 \pm 2.45}$
GPU	242.33 ± 0.59 ▷	$\mathbf{213.2 \pm 0.32}$	195.8 ± 23.27 ▷	$\mathbf{181.94 \pm 15.63}$	$\mathbf{221.29 \pm 25.58}$ ◁	224.38 ± 29.1	$\mathbf{234.98 \pm 7.27}$ ◁	241.54 ± 7.49
3d	13.75 ± 0.03 ▷	$\mathbf{12.65 \pm 0.02}$	11.4 ± 1.17 ▷	$\mathbf{10.62 \pm 1.0}$	11.17 ± 0.54 ▷	$\mathbf{10.45 \pm 0.47}$	10.36 ± 1.1 ▷	$\mathbf{9.43 \pm 1.21}$
Avg. Rank	4.29	3.04	3.18	2.96	5.68	5.04	6.32	5.50

$RMSE$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.03 ± 0.0	0.03 ± 0.0	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.01	0.02 ± 0.0	0.02 ± 0.0	0.02 ± 0.0
cpusum	$\mathbf{10.34 \pm 0.52}$ ◁	11.33 ± 0.23	$\mathbf{9.22 \pm 0.56}$ ◁	9.86 ± 0.57	$\mathbf{12.04 \pm 1.13}$ ◁	12.76 ± 1.15	$\mathbf{12.46 \pm 0.27}$ ◁	13.05 ± 0.33
elevator	$\mathbf{0.02 \pm 0.02}$	0.03 ± 0.03	$\mathbf{0.08 \pm 0.05}$ ◁	$21 \times 10^3 \pm 23 \times 10^3$	$\mathbf{0.09 \pm 0.01}$ ◁	$33 \times 10^3 \pm 19 \times 10^2$	$\mathbf{0.08 \pm 0.0}$ ◁	$28 \times 10^3 \pm 12 \times 10^2$
bike	$\mathbf{8.14 \pm 0.85}$ ◁	8.28 ± 0.87	$\mathbf{37.12 \pm 6.28}$	37.92 ± 5.06	$\mathbf{96.57 \pm 15.29}$ ◁	97.62 ± 15.52	$\mathbf{101.94 \pm 4.33}$ ◁	103.09 ± 4.39
energy	$\mathbf{94.91 \pm 1.5}$ ◁	103.69 ± 1.93	$\mathbf{97.82 \pm 0.98}$ ◁	107.21 ± 1.54	$\mathbf{99.71 \pm 0.27}$ ◁	108.52 ± 0.27	$\mathbf{99.17 \pm 0.3}$ ◁	107.96 ± 0.2
calhousing	$69 \times 10^3 \pm \mathbf{559.84}$ ◁	$70 \times 10^3 \pm 480.48$	$71 \times 10^3 \pm \mathbf{428.16}$ ◁	$72 \times 10^3 \pm 404.2$	$84 \times 10^3 \pm 44 \times 10^2$ ◁	$85 \times 10^3 \pm 45 \times 10^2$	$85 \times 10^3 \pm 14 \times 10^2$ ◁	$86 \times 10^3 \pm 14 \times 10^2$
gasemission	$\mathbf{7.9 \pm 0.04}$ ◁	8.01 ± 0.05	7.78 ± 0.06	$\mathbf{7.74 \pm 0.14}$	8.71 ± 0.51 ▷	$\mathbf{8.65 \pm 0.54}$	8.91 ± 0.19 ▷	$\mathbf{8.88 \pm 0.19}$
mv	$\mathbf{4.49 \pm 0.02}$ ◁	4.53 ± 0.02	$\mathbf{3.51 \pm 0.33}$	3.52 ± 0.34	$\mathbf{5.79 \pm 0.79}$ ◁	5.84 ± 0.81	$\mathbf{6.06 \pm 0.24}$ ◁	6.1 ± 0.24
fried	$\mathbf{2.63 \pm 0.01}$ ◁	2.65 ± 0.01	2.43 ± 0.07 ▷	$\mathbf{2.4 \pm 0.08}$	3.2 ± 0.28 ▷	$\mathbf{3.18 \pm 0.28}$	3.33 ± 0.07 ▷	$\mathbf{3.31 \pm 0.07}$
polution	$\mathbf{79.18 \pm 0.85}$ ◁	82.76 ± 0.9	$\mathbf{74.88 \pm 1.43}$ ◁	78.08 ± 1.64	$\mathbf{78.19 \pm 1.77}$ ◁	81.15 ± 1.71	$\mathbf{79.15 \pm 0.45}$ ◁	82.23 ± 0.35
car price	$14 \times 10^8 \pm 13 \times 10^8$ ◁	$29 \times 10^8 \pm 19 \times 10^8$	$12 \times 10^9 \pm 18 \times 10^9$	$57 \times 10^8 \pm 37 \times 10^8$	$29 \times 10^9 \pm 17 \times 10^8$ ▷	$84 \times 10^8 \pm 50 \times 10^7$	$24 \times 10^9 \pm 10 \times 10^8$ ▷	$71 \times 10^8 \pm 30 \times 10^7$
query	$\mathbf{134.05 \pm 0.72}$ ◁	135.11 ± 0.74	110.6 ± 7.67	$\mathbf{110.19 \pm 8.19}$	159.43 ± 18.0 ▷	$\mathbf{158.94 \pm 18.12}$	168.95 ± 3.96 ▷	$\mathbf{168.21 \pm 4.02}$
GPU	$\mathbf{284.72 \pm 0.66}$ ◁	322.59 ± 1.11	$\mathbf{230.3 \pm 27.1}$ ◁	260.79 ± 30.85	$\mathbf{246.51 \pm 23.32}$ ◁	269.93 ± 22.0	$\mathbf{259.18 \pm 6.53}$ ◁	279.39 ± 7.41
3d	$\mathbf{18.29 \pm 0.02}$ ◁	18.51 ± 0.03	15.45 ± 1.41	$\mathbf{15.44 \pm 1.52}$	15.11 ± 0.62	$\mathbf{15.08 \pm 0.67}$	14.78 ± 0.78 ▷	$\mathbf{14.69 \pm 0.8}$
Avg. Rank	3.32	4.61	2.64	3.18	4.82	5.61	5.50	6.32

$RMSE$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.03 ± 0.0	0.03 ± 0.0	$\mathbf{0.02 \pm 0.0}$ ◁	0.04 ± 0.0	$\mathbf{0.02 \pm 0.01}$ ◁	0.04 ± 0.01	$\mathbf{0.02 \pm 0.0}$ ◁	0.03 ± 0.0
cpusum	$\mathbf{10.34 \pm 0.52}$ ◁	19.77 ± 1.8	$\mathbf{9.22 \pm 0.56}$ ◁	25.84 ± 1.7	$\mathbf{12.04 \pm 1.13}$ ◁	30.72 ± 1.15	$\mathbf{12.46 \pm 0.27}$ ◁	30.16 ± 0.76
elevator	0.02 ± 0.02 ▷	$\mathbf{0.01 \pm 0.01}$	0.08 ± 0.05 ▷	$\mathbf{0.01 \pm 0.0}$	0.09 ± 0.01 ▷	$\mathbf{0.01 \pm 0.0}$	0.08 ± 0.0 ▷	$\mathbf{0.01 \pm 0.0}$
bike	$\mathbf{8.14 \pm 0.85}$ ◁	13.99 ± 1.38	$\mathbf{37.12 \pm 6.28}$ ◁	91.29 ± 14.46	$\mathbf{96.57 \pm 15.29}$ ◁	151.29 ± 14.89	$\mathbf{101.94 \pm 4.33}$ ◁	151.49 ± 6.44
energy	$\mathbf{94.91 \pm 1.5}$ ◁	157.93 ± 2.88	$\mathbf{97.82 \pm 0.98}$ ◁	174.81 ± 5.55	$\mathbf{99.71 \pm 0.27}$ ◁	177.95 ± 0.72	$\mathbf{99.17 \pm 0.3}$ ◁	174.13 ± 1.27
calhousing	$69 \times 10^3 \pm \mathbf{559.84}$ ◁	$80 \times 10^3 \pm 23 \times 10^2$	$71 \times 10^3 \pm \mathbf{428.16}$ ◁	$91 \times 10^3 \pm 926.37$	$84 \times 10^3 \pm 44 \times 10^2$ ◁	$10 \times 10^4 \pm 37 \times 10^2$	$85 \times 10^3 \pm 14 \times 10^2$ ◁	$10 \times 10^4 \pm 17 \times 10^2$
gasemission	$\mathbf{7.9 \pm 0.04}$ ◁	9.08 ± 0.07	$\mathbf{7.78 \pm 0.06}$ ◁	9.94 ± 0.4	$\mathbf{8.71 \pm 0.51}$ ◁	10.73 ± 0.21	$\mathbf{8.91 \pm 0.19}$ ◁	10.55 ± 0.22
mv	$\mathbf{4.49 \pm 0.02}$ ◁	4.8 ± 0.02	$\mathbf{3.51 \pm 0.33}$ ◁	4.13 ± 0.23	$\mathbf{5.79 \pm 0.79}$ ◁	7.13 ± 0.99	$\mathbf{6.06 \pm 0.24}$ ◁	7.43 ± 0.3
fried	$\mathbf{2.63 \pm 0.01}$ ◁	2.75 ± 0.01	$\mathbf{2.43 \pm 0.07}$ ◁	2.76 ± 0.01	$\mathbf{3.2 \pm 0.28}$ ◁	3.44 ± 0.23	$\mathbf{3.33 \pm 0.07}$ ◁	3.52 ± 0.07
polution	$\mathbf{79.18 \pm 0.85}$ ◁	109.51 ± 0.7	$\mathbf{74.88 \pm 1.43}$ ◁	106.17 ± 1.14	$\mathbf{78.19 \pm 1.77}$ ◁	112.41 ± 2.67	$\mathbf{79.15 \pm 0.45}$ ◁	114.22 ± 0.46
car price	$14 \times 10^8 \pm 13 \times 10^8$ ◁	$56 \times 10^8 \pm 39 \times 10^8$	$12 \times 10^9 \pm 18 \times 10^9$	$65 \times 10^8 \pm 54 \times 10^7$	$29 \times 10^9 \pm 17 \times 10^8$ ▷	$51 \times 10^8 \pm 31 \times 10^7$	$24 \times 10^9 \pm 10 \times 10^8$ ▷	$43 \times 10^8 \pm 18 \times 10^7$
query	$\mathbf{134.05 \pm 0.72}$ ◁	142.16 ± 0.78	$\mathbf{110.6 \pm 7.67}$ ◁	121.04 ± 6.89	$\mathbf{159.43 \pm 18.0}$ ◁	165.3 ± 16.64	$\mathbf{168.95 \pm 3.96}$ ◁	173.51 ± 3.83
GPU	$\mathbf{284.72 \pm 0.66}$ ◁	424.37 ± 1.91	$\mathbf{230.3 \pm 27.1}$ ◁	355.23 ± 34.63	$\mathbf{246.51 \pm 23.32}$ ◁	408.21 ± 44.73	$\mathbf{259.18 \pm 6.53}$ ◁	432.28 ± 12.71
3d	$\mathbf{18.29 \pm 0.02}$ ◁	22.26 ± 0.02	$\mathbf{15.45 \pm 1.41}$ ◁	19.21 ± 1.49	$\mathbf{15.11 \pm 0.62}$ ◁	18.97 ± 0.69	$\mathbf{14.78 \pm 0.78}$ ◁	18.52 ± 0.98
Avg. Rank	2.71	4.54	2.25	4.93	4.14	6.50	4.54	6.39

TABLE A.5: $RMSE$ and $RMSE_\phi$ estimates considering `all` (double-side) rare cases: $thr_\phi = 0.8$.

$RMSE_\phi$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.01 ▷	$\mathbf{0.03 \pm 0.0}$	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.0
cpusum	18.74 ± 1.46 ▷	$\mathbf{13.53 \pm 0.9}$	16.37 ± 1.44 ▷	$\mathbf{12.07 \pm 0.87}$	23.81 ± 2.83 ▷	$\mathbf{21.32 \pm 3.04}$	24.73 ± 0.83 ▷	$\mathbf{22.58 \pm 0.71}$
elevator	$\mathbf{0.04 \pm 0.03}$ ◁	0.07 ± 0.07	$\mathbf{0.19 \pm 0.14}$ ◁	$55 \times 10^3 \pm 61 \times 10^3$	$\mathbf{0.24 \pm 0.01}$ ◁	$86 \times 10^3 \pm 50 \times 10^2$	$\mathbf{0.21 \pm 0.01}$ ◁	$73 \times 10^3 \pm 31 \times 10^2$
bike	$\mathbf{7.17 \pm 0.57}$ ◁	7.34 ± 0.58	$\mathbf{35.72 \pm 6.24}$	36.33 ± 5.42	$\mathbf{131.31 \pm 23.34}$ ◁	133.59 ± 23.9	$\mathbf{140.9 \pm 6.02}$ ◁	143.46 ± 6.13
energy	218.02 ± 4.84 ▷	$\mathbf{196.09 \pm 4.25}$	221.51 ± 1.27 ▷	$\mathbf{199.15 \pm 1.06}$	227.81 ± 1.55 ▷	$\mathbf{205.2 \pm 1.51}$	227.52 ± 0.9 ▷	$\mathbf{205.87 \pm 0.65}$
calhousing	$12 \times 10^4 \pm 20 \times 10^2$ ▷	$11 \times 10^4 \pm 19 \times 10^2$	$11 \times 10^4 \pm 17 \times 10^2$ ▷	$11 \times 10^4 \pm 19 \times 10^2$	$15 \times 10^4 \pm 11 \times 10^3$ ▷	$14 \times 10^4 \pm 11 \times 10^3$	$15 \times 10^4 \pm 35 \times 10^2$ ▷	$14 \times 10^4 \pm 36 \times 10^2$
gasemission	18.56 ± 0.32 ▷	$\mathbf{16.24 \pm 0.38}$	17.41 ± 0.59 ▷	$\mathbf{15.13 \pm 0.57}$	20.3 ± 1.79 ▷	$\mathbf{18.55 \pm 2.01}$	21.07 ± 0.62 ▷	$\mathbf{19.55 \pm 0.61}$
mv	$\mathbf{5.56 \pm 0.06}$ ◁	5.65 ± 0.06	$\mathbf{4.32 \pm 0.43}$ ◁	4.38 ± 0.44	$\mathbf{6.78 \pm 0.89}$ ◁	7.1 ± 0.97	$\mathbf{7.06 \pm 0.29}$ ◁	7.39 ± 0.3
fried	3.52 ± 0.04 ▷	$\mathbf{3.14 \pm 0.04}$	3.15 ± 0.14 ▷	$\mathbf{2.77 \pm 0.15}$	5.62 ± 0.79 ▷	$\mathbf{5.46 \pm 0.83}$	6.0 ± 0.19 ▷	$\mathbf{5.86 \pm 0.19}$
polution	119.66 ± 1.41 ▷	$\mathbf{109.91 \pm 1.05}$	109.33 ± 3.35 ▷	$\mathbf{99.43 \pm 3.57}$	117.34 ± 4.15 ▷	$\mathbf{110.15 \pm 5.07}$	119.07 ± 1.16 ▷	$\mathbf{111.89 \pm 1.54}$
car price	$36 \times 10^3 \pm 14 \times 10^3$ ▷	$27 \times 10^3 \pm 11 \times 10^3$	$52 \times 10^3 \pm 82 \times 10^2$ ▷	$26 \times 10^3 \pm 12 \times 10^2$	$53 \times 10^3 \pm 25 \times 10^2$ ▷	$26 \times 10^3 \pm \mathbf{325.56}$	$47 \times 10^3 \pm 12 \times 10^2$ ▷	$25 \times 10^3 \pm \mathbf{305.06}$
query	389.6 ± 3.21 ▷	$\mathbf{376.7 \pm 3.3}$	318.07 ± 23.87 ▷	$\mathbf{302.87 \pm 24.71}$	383.92 ± 30.26 ▷	$\mathbf{375.1 \pm 32.33}$	403.07 ± 5.37 ▷	$\mathbf{392.74 \pm 6.53}$
GPU	498.2 ± 1.71 ▷	$\mathbf{391.79 \pm 1.2}$	402.21 ± 48.04 ▷	$\mathbf{316.86 \pm 37.69}$	456.83 ± 53.67 ▷	$\mathbf{392.01 \pm 57.39}$	485.04 ± 15.43 ▷	$\mathbf{425.76 \pm 14.3}$
3d	43.97 ± 0.08 ▷	$\mathbf{40.83 \pm 0.08}$	36.43 ± 3.79 ▷	$\mathbf{33.46 \pm 3.71}$	35.77 ± 1.81 ▷	$\mathbf{33.03 \pm 1.88}$	33.33 ± 3.54 ▷	$\mathbf{28.8 \pm 4.72}$
Avg. Rank	4.79	3.0	3.64	2.07	6.07	4.43	6.57	5.43

$RMSE_\phi$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.04 ± 0.0 ▷	$\mathbf{0.03 \pm 0.0}$	0.04 ± 0.01 ▷	$\mathbf{0.03 \pm 0.0}$	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.0
cpusum	18.74 ± 1.46	$\mathbf{18.73 \pm 3.64}$	$\mathbf{16.37 \pm 1.44}$ ◁	26.93 ± 2.12	$\mathbf{23.81 \pm 2.83}$ ◁	31.99 ± 1.16	$\mathbf{24.73 \pm 0.83}$ ◁	31.11 ± 1.05
elevator	0.04 ± 0.03 ▷	$\mathbf{0.01 \pm 0.01}$	0.19 ± 0.14 ▷	$\mathbf{0.02 \pm 0.0}$	0.24 ± 0.01 ▷	$\mathbf{0.02 \pm 0.0}$	0.21 ± 0.01 ▷	$\mathbf{0.02 \pm 0.0}$
bike	$\mathbf{7.17 \pm 0.57}$ ◁	12.45 ± 0.82	$\mathbf{35.72 \pm 6.24}$ ◁	82.95 ± 13.3	$\mathbf{131.31 \pm 23.34}$ ◁	176.8 ± 24.26	$\mathbf{140.9 \pm 6.02}$ ◁	183.35 ± 7.82
energy	218.02 ± 4.84 ▷	$\mathbf{173.36 \pm 4.02}$	221.51 ± 1.27 ▷	$\mathbf{187.82 \pm 4.17}$	227.81 ± 1.55 ▷	$\mathbf{186.3 \pm 2.46}$	227.52 ± 0.9 ▷	$\mathbf{182.49 \pm 0.82}$
calhousing	$12 \times 10^4 \pm 20 \times 10^2$ ▷	$10 \times 10^4 \pm 29 \times 10^2$	$11 \times 10^4 \pm 17 \times 10^2$ ▷	$10 \times 10^4 \pm 11 \times 10^2$	$15 \times 10^4 \pm 11 \times 10^3$ ▷	$13 \times 10^4 \pm 98 \times 10^2$	$15 \times 10^4 \pm 35 \times 10^2$ ▷	$13 \times 10^4 \pm 33 \times 10^2$
gasemission	18.56 ± 0.32 ▷	$\mathbf{15.0 \pm 0.38}$	17.41 ± 0.59 ▷	$\mathbf{15.25 \pm 0.11}$	20.3 ± 1.79 ▷	$\mathbf{18.5 \pm 1.65}$	21.07 ± 0.62 ▷	$\mathbf{19.22 \pm 0.57}$
mv	$\mathbf{5.56 \pm 0.06}$ ◁	5.99 ± 0.06	$\mathbf{4.32 \pm 0.43}$ ◁	4.77 ± 0.41	$\mathbf{6.78 \pm 0.89}$ ◁	8.8 ± 1.33	$\mathbf{7.06 \pm 0.29}$ ◁	9.29 ± 0.38
fried	3.52 ± 0.04 ▷	$\mathbf{2.63 \pm 0.05}$	3.15 ± 0.14 ▷	$\mathbf{2.54 \pm 0.05}$	5.62 ± 0.79 ▷	$\mathbf{5.42 \pm 0.85}$	6.0 ± 0.19 ▷	$\mathbf{5.81 \pm 0.2}$
polution	119.66 ± 1.41 ▷	$\mathbf{104.89 \pm 1.11}$	109.33 ± 3.35 ▷	$\mathbf{100.0 \pm 1.65}$	117.34 ± 4.15	$\mathbf{116.49 \pm 6.37}$	$\mathbf{119.07 \pm 1.16}$ ◁	119.78 ± 1.47
car price	$36 \times 10^3 \pm 14 \times 10^3$ ◁	$33 \times 10^5 \pm 10 \times 10^6$	$52 \times 10^3 \pm 82 \times 10^2$ ◁	$22 \times 10^6 \pm 35 \times 10^5$	$53 \times 10^3 \pm 25 \times 10^2$ ◁	$22 \times 10^6 \pm 13 \times 10^5$	$47 \times 10^3 \pm 12 \times 10^2$ ◁	$19 \times 10^6 \pm 81 \times 10^4$
query	389.6 ± 3.21 ▷	$\mathbf{360.87 \pm 3.32}$	318.07 ± 23.87 ▷	$\mathbf{295.66 \pm 21.13}$	383.92 ± 30.26 ▷	$\mathbf{373.72 \pm 32.88}$	403.07 ± 5.37 ▷	$\mathbf{390.23 \pm 7.23}$
GPU	498.2 ± 1.71 ▷	$\mathbf{362.99 \pm 0.88}$	402.21 ± 48.04 ▷	$\mathbf{311.67 \pm 25.81}$	456.83 ± 53.67 ▷	$\mathbf{374.46 \pm 44.81}$	485.04 ± 15.43 ▷	$\mathbf{400.76 \pm 11.66}$
3d	43.97 ± 0.08 ▷	$\mathbf{30.68 \pm 0.11}$	36.43 ± 3.79 ▷	$\mathbf{25.83 \pm 2.42}$	35.77 ± 1.81 ▷	$\mathbf{25.88 \pm 1.37}$	33.33 ± 3.54 ▷	$\mathbf{23.18 \pm 3.29}$
Avg. Rank	4.68	2.25	3.68	2.68	5.82	4.96	6.32	5.61

$RMSE$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.05 ± 0.0 ▷	$\mathbf{0.04 \pm 0.0}$	0.04 ± 0.0 ▷	$\mathbf{0.03 \pm 0.01}$	0.04 ± 0.01	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.0
cpusum	18.81 ± 1.45 ▷	$\mathbf{13.74 \pm 0.89}$	16.44 ± 1.44 ▷	$\mathbf{12.27 \pm 0.87}$	23.94 ± 2.85 ▷	$\mathbf{21.55 \pm 3.05}$	24.87 ± 0.83 ▷	$\mathbf{22.82 \pm 0.71}$
elevator	$\mathbf{0.04 \pm 0.03}$ ◁	0.07 ± 0.07	$\mathbf{0.19 \pm 0.14}$ ◁	$55 \times 10^3 \pm 61 \times 10^3$	$\mathbf{0.24 \pm 0.01}$ ◁	$86 \times 10^3 \pm 50 \times 10^2$	$\mathbf{0.21 \pm 0.01}$ ◁	$73 \times 10^3 \pm 31 \times 10^2$
bike	$\mathbf{7.47 \pm 0.58}$ ◁	7.65 ± 0.59	$\mathbf{37.54 \pm 6.62}$	38.09 ± 5.75	$\mathbf{135.66 \pm 24.03}$ ◁	138.06 ± 24.63	$\mathbf{145.51 \pm 6.21}$ ◁	148.21 ± 6.34
energy	218.34 ± 4.85 ▷	$\mathbf{196.97 \pm 4.27}$	221.93 ± 1.28 ▷	$\mathbf{200.21 \pm 1.1}$	228.24 ± 1.55 ▷	$\mathbf{206.19 \pm 1.47}$	227.95 ± 0.9 ▷	$\mathbf{206.83 \pm 0.65}$
calhousing	$12 \times 10^4 \pm 20 \times 10^2$ ▷	$11 \times 10^4 \pm 19 \times 10^2$	$12 \times 10^4 \pm 17 \times 10^2$ ▷	$11 \times 10^4 \pm 19 \times 10^2$	$15 \times 10^4 \pm 12 \times 10^3$ ▷	$14 \times 10^4 \pm 12 \times 10^3$	$16 \times 10^4 \pm 36 \times 10^2$ ▷	$15 \times 10^4 \pm 38 \times 10^2$
gasemission	19.05 ± 0.33 ▷	$\mathbf{16.72 \pm 0.39}$	17.9 ± 0.58 ▷	$\mathbf{15.63 \pm 0.56}$	20.88 ± 1.84 ▷	$\mathbf{19.14 \pm 2.05}$	21.68 ± 0.64 ▷	$\mathbf{20.17 \pm 0.63}$
mv	$\mathbf{6.04 \pm 0.06}$ ◁	6.13 ± 0.06	$\mathbf{4.7 \pm 0.46}$ ◁	4.75 ± 0.47	$\mathbf{7.41 \pm 0.99}$ ◁	7.74 ± 1.06	$\mathbf{7.72 \pm 0.31}$ ◁	8.06 ± 0.33
fried	3.9 ± 0.05 ▷	$\mathbf{3.48 \pm 0.05}$	3.5 ± 0.15 ▷	$\mathbf{3.08 \pm 0.17}$	6.24 ± 0.88 ▷	$\mathbf{6.06 \pm 0.93}$	6.66 ± 0.21 ▷	$\mathbf{6.51 \pm 0.22}$
polution	122.96 ± 1.37 ▷	$\mathbf{113.96 \pm 1.04}$	112.29 ± 3.45 ▷	$\mathbf{103.06 \pm 3.7}$	120.73 ± 4.33 ▷	$\mathbf{114.31 \pm 5.31}$	122.53 ± 1.21 ▷	$\mathbf{116.14 \pm 1.61}$
car price	$38 \times 10^3 \pm 15 \times 10^3$ ▷	$27 \times 10^3 \pm 10 \times 10^3$	$60 \times 10^3 \pm 10 \times 10^3$ ▷	$27 \times 10^3 \pm 15 \times 10^2$	$62 \times 10^3 \pm 31 \times 10^2$ ▷	$27 \times 10^3 \pm \mathbf{396.05}$	$54 \times 10^3 \pm 15 \times 10^2$ ▷	$26 \times 10^3 \pm \mathbf{331.89}$
query	420.41 ± 3.56 ▷	$\mathbf{406.71 \pm 3.67}$	342.38 ± 26.18 ▷	$\mathbf{326.42 \pm 27.01}$	414.02 ± 32.97 ▷	$\mathbf{404.81 \pm 35.14}$	434.35 ± 6.13 ▷	$\mathbf{423.25 \pm 7.4}$
GPU	499.8 ± 1.73 ▷	$\mathbf{396.5 \pm 1.26}$	403.53 ± 48.18 ▷	$\mathbf{320.7 \pm 38.13}$	458.22 ± 53.79 ▷	$\mathbf{395.73 \pm 57.59}$	486.5 ± 15.46 ▷	$\mathbf{429.37 \pm 14.47}$
3d	45.21 ± 0.08 ▷	$\mathbf{41.93 \pm 0.07}$	37.47 ± 3.89 ▷	$\mathbf{34.38 \pm 3.8}$	36.81 ± 1.86 ▷	$\mathbf{33.95 \pm 1.92}$	34.26 ± 3.66 ▷	$\mathbf{29.63 \pm 4.83}$
Avg. Rank	4.93	3.04	3.61	2.0	6.04	4.46	6.54	5.39

$RMSE$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.05 ± 0.0 ▷	$\mathbf{0.04 \pm 0.0}$	0.04 ± 0.0	0.04 ± 0.0	0.04 ± 0.01	0.04 ± 0.01	0.04 ± 0.0	0.04 ± 0.0
cpusum	$\mathbf{18.81 \pm 1.45}$	19.66 ± 3.75	$\mathbf{16.44 \pm 1.44}$ ◁	27.91 ± 2.18	$\mathbf{23.94 \pm 2.85}$ ◁	33.27 ± 1.25	$\mathbf{24.87 \pm 0.83}$ ◁	32.44 ± 1.08
elevator	0.04 ± 0.03 ▷	$\mathbf{0.01 \pm 0.01}$	0.19 ± 0.14 ▷	$\mathbf{0.02 \pm 0.0}$	0.24 ± 0.01 ▷	$\mathbf{0.02 \pm 0.0}$	0.21 ± 0.01 ▷	$\mathbf{0.02 \pm 0.0}$
bike	$\mathbf{7.47 \pm 0.58}$ ◁	12.99 ± 0.83	$\mathbf{37.54 \pm 6.62}$ ◁	87.18 ± 13.96	$\mathbf{135.66 \pm 24.03}$ ◁	183.91 ± 25.02	$\mathbf{145.51 \pm 6.21}$ ◁	190.52 ± 8.12
energy	218.34 ± 4.85 ▷	$\mathbf{177.34 \pm 4.06}$	221.93 ± 1.28 ▷	$\mathbf{192.53 \pm 4.26}$	228.24 ± 1.55 ▷	$\mathbf{190.88 \pm 2.52}$	227.95 ± 0.9 ▷	$\mathbf{187.06 \pm 0.84}$
calhousing	$12 \times 10^4 \pm 20 \times 10^2$ ▷	$10 \times 10^4 \pm 29 \times 10^2$	$12 \times 10^4 \pm 17 \times 10^2$ ▷	$10 \times 10^4 \pm 11 \times 10^2$	$15 \times 10^4 \pm 12 \times 10^3$ ▷	$14 \times 10^4 \pm 10 \times 10^3$	$16 \times 10^4 \pm 36 \times 10^2$ ▷	$14 \times 10^4 \pm 35 \times 10^2$
gasemission	19.05 ± 0.33 ▷	$\mathbf{15.53 \pm 0.4}$	17.9 ± 0.58 ▷	$\mathbf{15.81 \pm 0.12}$	20.88 ± 1.84 ▷	$\mathbf{19.12 \pm 1.68}$	21.68 ± 0.64 ▷	$\mathbf{19.85 \pm 0.58}$
mv	$\mathbf{6.04 \pm 0.06}$ ◁	6.49 ± 0.07	$\mathbf{4.7 \pm 0.46}$ ◁	5.16 ± 0.45	$\mathbf{7.41 \pm 0.99}$ ◁	9.53 ± 1.44	$\mathbf{7.72 \pm 0.31}$ ◁	10.06 ± 0.41
fried	3.9 ± 0.05 ▷	$\mathbf{2.93 \pm 0.06}$	3.5 ± 0.15 ▷	$\mathbf{2.83 \pm 0.06}$	6.24 ± 0.88 ▷	$\mathbf{6.01 \pm 0.94}$	6.66 ± 0.21 ▷	$\mathbf{6.45 \pm 0.23}$
polution	122.96 ± 1.37 ▷	$\mathbf{112.1 \pm 1.19}$	112.29 ± 3.45 ▷	$\mathbf{106.62 \pm 1.84}$	$\mathbf{120.73 \pm 4.33}$ ◁	124.48 ± 6.91	$\mathbf{122.53 \pm 1.21}$ ◁	128.05 ± 1.6
car price	$38 \times 10^3 \pm 15 \times 10^3$ ◁	$36 \times 10^5 \pm 11 \times 10^6$	$60 \times 10^3 \pm 10 \times 10^3$ ◁	$25 \times 10^6 \pm 49 \times 10^5$	$62 \times 10^3 \pm 31 \times 10^2$ ◁	$26 \times 10^6 \pm 15 \times 10^5$	$54 \times 10^3 \pm 15 \times 10^2$ ◁	$22 \times 10^6 \pm 95 \times 10^4$
query	420.41 ± 3.56 ▷	$\mathbf{388.97 \pm 3.71}$	342.38 ± 26.18 ▷	$\mathbf{317.99 \pm 23.08}$	414.02 ± 32.97 ▷	$\mathbf{402.95 \pm 35.79}$	434.35 ± 6.13 ▷	$\mathbf{420.24 \pm 8.14}$
GPU	499.8 ± 1.73 ▷	$\mathbf{372.93 \pm 0.85}$	403.53 ± 48.18 ▷	$\mathbf{320.29 \pm 26.52}$	458.22 ± 53.79 ▷	$\mathbf{389.01 \pm 47.97}$	486.5 ± 15.46 ▷	$\mathbf{416.94 \pm 12.61}$
3d	45.21 ± 0.08 ▷	$\mathbf{31.34 \pm 0.11}$	37.47 ± 3.89 ▷	$\mathbf{26.43 \pm 2.46}$	36.81 ± 1.86 ▷	$\mathbf{26.48 \pm 1.39}$	34.26 ± 3.66 ▷	$\mathbf{23.81 \pm 3.28}$
Avg. Rank	4.71	2.5	3.5	2.86	5.64	5.14	6.14	5.5

TABLE A.6: $RMSE$ and $RMSE_\phi$ estimates considering `all` (double-side) rare cases: $thr_\phi = 0.9$.

$RMSE_\phi$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.04 ± 0.0 ▷	**0.03 ± 0.01**	0.05 ± 0.01	**0.04 ± 0.0**	0.05 ± 0.0 ▷	**0.04 ± 0.0**
cpusum	22.43 ± 1.88 ▷	**15.6 ± 1.18**	19.73 ± 1.77 ▷	**14.04 ± 1.05**	28.61 ± 3.41 ▷	**25.15 ± 3.63**	29.78 ± 0.97 ▷	**26.74 ± 0.81**
elevator	**0.04 ± 0.03** ◁	0.08 ± 0.07	**0.21 ± 0.14** ◁	$60 \times 10^3 \pm 66 \times 10^3$	**0.26 ± 0.02** ◁	$93 \times 10^3 \pm 55 \times 10^2$	**0.22 ± 0.01** ◁	$79 \times 10^3 \pm 33 \times 10^2$
bike	**7.43 ± 0.62** ◁	7.6 ± 0.63	**36.61 ± 6.13**	37.33 ± 5.26	**138.42 ± 24.74** ◁	140.84 ± 25.34	**148.62 ± 6.32** ◁	151.34 ± 6.45
energy	239.37 ± 4.68 ▷	**213.54 ± 4.15**	244.58 ± 1.4 ▷	**217.93 ± 1.09**	251.27 ± 1.76 ▷	**224.47 ± 1.75**	251.26 ± 0.98 ▷	**225.51 ± 0.69**
calhousing	$15 \times 10^4 \pm 19 \times 10^2$ ▷	$14 \times 10^4 \pm 18 \times 10^2$	$14 \times 10^4 \pm 20 \times 10^2$ ▷	$13 \times 10^4 \pm 23 \times 10^2$	$18 \times 10^4 \pm 13 \times 10^3$ ▷	$17 \times 10^4 \pm 13 \times 10^3$	$18 \times 10^4 \pm 41 \times 10^2$ ▷	$17 \times 10^4 \pm 41 \times 10^2$
gasemission	21.23 ± 0.36 ▷	**18.35 ± 0.42**	19.92 ± 0.75 ▷	**17.08 ± 0.73**	23.15 ± 2.07 ▷	**21.01 ± 2.35**	24.01 ± 0.73 ▷	**22.15 ± 0.72**
mv	**5.92 ± 0.06** ◁	6.05 ± 0.05	**4.6 ± 0.46** ◁	4.69 ± 0.47	**7.12 ± 0.93** ◁	7.53 ± 1.02	**7.42 ± 0.3** ◁	7.84 ± 0.32
fried	4.12 ± 0.07 ▷	**3.65 ± 0.08**	3.66 ± 0.16 ▷	**3.19 ± 0.18**	6.5 ± 0.9 ▷	**6.3 ± 0.95**	6.92 ± 0.21 ▷	**6.75 ± 0.22**
polution	139.68 ± 1.72 ▷	**125.88 ± 1.22**	127.6 ± 4.01 ▷	**113.86 ± 4.17**	136.48 ± 4.72 ▷	**125.9 ± 5.75**	138.61 ± 1.31 ▷	**128.01 ± 1.76**
car price	$38 \times 10^3 \pm 17 \times 10^3$ ▷	$31 \times 10^3 \pm 13 \times 10^3$	$33 \times 10^3 \pm 15 \times 10^2$ ▷	$29 \times 10^3 \pm$ **902.23**	$31 \times 10^3 \pm 283.4$ ▷	$29 \times 10^3 \pm$ **215.5**	$31 \times 10^3 \pm 11 \times 10^2$ ▷	$28 \times 10^3 \pm$ **311.23**
query	449.25 ± 3.33 ▷	**433.49 ± 3.52**	370.92 ± 25.73 ▷	**351.64 ± 26.97**	445.02 ± 33.79 ▷	**433.64 ± 36.44**	468.72 ± 5.02 ▷	**456.54 ± 6.27**
GPU	550.28 ± 1.74 ▷	**425.73 ± 1.12**	444.22 ± 53.09 ▷	**344.25 ± 40.99**	504.91 ± 59.5 ▷	**427.95 ± 63.38**	536.3 ± 17.06 ▷	**465.79 ± 15.48**
3d	48.05 ± 0.09 ▷	**44.74 ± 0.09**	39.8 ± 4.15 ▷	**36.65 ± 4.07**	39.08 ± 1.98 ▷	**36.17 ± 2.06**	36.51 ± 3.79 ▷	**31.49 ± 5.21**
Avg. Rank	5.14	2.82	3.68	2.07	5.93	4.32	6.71	5.32

$RMSE_\phi$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.05 ± 0.0 ▷	**0.03 ± 0.01**	0.04 ± 0.0 ▷	**0.03 ± 0.0**	0.05 ± 0.01	**0.04 ± 0.01**	0.05 ± 0.0	**0.04 ± 0.01**
cpusum	22.43 ± 1.88	**19.54 ± 4.43**	**19.73 ± 1.77** ◁	29.92 ± 2.5	**28.61 ± 3.41** ◁	34.51 ± 0.89	**29.78 ± 0.97** ◁	33.12 ± 1.13
elevator	0.04 ± 0.03 ▷	**0.02 ± 0.01**	0.21 ± 0.14 ▷	**0.02 ± 0.0**	0.26 ± 0.02 ▷	**0.02 ± 0.0**	0.22 ± 0.01 ▷	**0.02 ± 0.0**
bike	**7.43 ± 0.62** ◁	12.89 ± 0.91	**36.61 ± 6.13** ◁	84.74 ± 13.44	**138.42 ± 24.74** ◁	184.45 ± 25.85	**148.62 ± 6.32** ◁	191.76 ± 8.14
energy	239.37 ± 4.68 ▷	**178.13 ± 3.74**	244.58 ± 1.4 ▷	**193.92 ± 4.53**	251.27 ± 1.76 ▷	**191.74 ± 2.92**	251.26 ± 0.98 ▷	**187.38 ± 0.85**
calhousing	$15 \times 10^4 \pm 19 \times 10^2$ ▷	$12 \times 10^4 \pm 33 \times 10^2$	$14 \times 10^4 \pm 20 \times 10^2$ ▷	$12 \times 10^4 \pm 13 \times 10^2$	$18 \times 10^4 \pm 13 \times 10^3$ ▷	$15 \times 10^4 \pm 96 \times 10^2$	$18 \times 10^4 \pm 41 \times 10^2$ ▷	$15 \times 10^4 \pm 34 \times 10^2$
gasemission	21.23 ± 0.36 ▷	**16.7 ± 0.41**	19.92 ± 0.75 ▷	**17.03 ± 0.12**	23.15 ± 2.07 ▷	**20.83 ± 1.95**	24.01 ± 0.73 ▷	**21.7 ± 0.65**
mv	**5.92 ± 0.06** ◁	6.44 ± 0.06	**4.6 ± 0.46** ◁	5.11 ± 0.46	**7.12 ± 0.93** ◁	9.5 ± 1.46	**7.42 ± 0.3** ◁	10.05 ± 0.41
fried	4.12 ± 0.07 ▷	**3.0 ± 0.08**	3.66 ± 0.16 ▷	**2.93 ± 0.04**	6.5 ± 0.9 ▷	**6.26 ± 0.96**	6.92 ± 0.21 ▷	**6.69 ± 0.24**
polution	139.68 ± 1.72 ▷	**111.94 ± 1.27**	127.6 ± 4.01 ▷	**107.32 ± 1.61**	136.48 ± 4.72 ▷	**124.76 ± 6.71**	138.61 ± 1.31 ▷	**128.29 ± 1.57**
car price	$38 \times 10^3 \pm 17 \times 10^3$	$31 \times 10^5 \pm 96 \times 10^5$	$33 \times 10^3 \pm 15 \times 10^2$ ◁	$17 \times 10^6 \pm 17 \times 10^5$	$31 \times 10^3 \pm$ **283.4** ◁	$13 \times 10^6 \pm 81 \times 10^4$	$31 \times 10^3 \pm 11 \times 10^2$ ◁	$11 \times 10^6 \pm 49 \times 10^4$
query	449.25 ± 3.33 ▷	**417.57 ± 3.57**	370.92 ± 25.73 ▷	**345.49 ± 23.15**	445.02 ± 33.79 ▷	**433.22 ± 36.86**	468.72 ± 5.02 ▷	**454.6 ± 7.11**
GPU	550.28 ± 1.74 ▷	**382.53 ± 0.9**	444.22 ± 53.09 ▷	**328.37 ± 27.09**	504.91 ± 59.5 ▷	**384.59 ± 42.61**	536.3 ± 17.06 ▷	**410.13 ± 10.76**
3d	48.05 ± 0.09 ▷	**34.0 ± 0.11**	39.8 ± 4.15 ▷	**28.56 ± 2.73**	39.08 ± 1.98 ▷	**28.63 ± 1.55**	36.51 ± 3.79 ▷	**25.42 ± 3.86**
Avg. Rank	5.07	2.29	3.71	2.71	5.79	4.68	6.5	5.25

$RMSE$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.04 ± 0.0 ▷	0.04 ± 0.01	0.05 ± 0.0	**0.04 ± 0.01**	0.05 ± 0.0 ▷	**0.04 ± 0.0**
cpusum	22.45 ± 1.88 ▷	**15.64 ± 1.18**	19.75 ± 1.77 ▷	**14.08 ± 1.05**	28.65 ± 3.42 ▷	**25.22 ± 3.64**	29.83 ± 0.97 ▷	**26.81 ± 0.81**
elevator	**0.04 ± 0.03** ◁	0.08 ± 0.07	**0.21 ± 0.14** ◁	$60 \times 10^3 \pm 66 \times 10^3$	**0.26 ± 0.02** ◁	$93 \times 10^3 \pm 55 \times 10^2$	**0.22 ± 0.01** ◁	$79 \times 10^3 \pm 33 \times 10^2$
bike	**7.58 ± 0.62** ◁	7.75 ± 0.64	**37.59 ± 6.29**	38.25 ± 5.38	**140.52 ± 25.07** ◁	143.05 ± 25.7	**150.84 ± 6.42** ◁	153.7 ± 6.54
energy	239.43 ± 4.68 ▷	**213.67 ± 4.15**	244.66 ± 1.41 ▷	**218.12 ± 1.11**	251.36 ± 1.76 ▷	**224.64 ± 1.73**	251.34 ± 0.98 ▷	**225.66 ± 0.69**
calhousing	$15 \times 10^4 \pm 19 \times 10^2$ ▷	$14 \times 10^4 \pm 18 \times 10^2$	$14 \times 10^4 \pm 20 \times 10^2$ ▷	$13 \times 10^4 \pm 23 \times 10^2$	$18 \times 10^4 \pm 13 \times 10^3$ ▷	$17 \times 10^4 \pm 13 \times 10^3$	$18 \times 10^4 \pm 41 \times 10^2$ ▷	$17 \times 10^4 \pm 42 \times 10^2$
gasemission	21.37 ± 0.36 ▷	**18.48 ± 0.42**	20.05 ± 0.75 ▷	**17.21 ± 0.73**	23.32 ± 2.08 ▷	**21.17 ± 2.36**	24.18 ± 0.73 ▷	**22.32 ± 0.73**
mv	**6.14 ± 0.06** ◁	6.26 ± 0.05	**4.77 ± 0.47** ◁	4.86 ± 0.49	**7.4 ± 0.97** ◁	7.82 ± 1.07	**7.71 ± 0.32** ◁	8.14 ± 0.33
fried	4.3 ± 0.08 ▷	**3.81 ± 0.09**	3.83 ± 0.17 ▷	**3.34 ± 0.19**	6.82 ± 0.94 ▷	**6.61 ± 0.99**	7.25 ± 0.23 ▷	**7.08 ± 0.24**
polution	141.18 ± 1.73 ▷	**127.74 ± 1.24**	128.99 ± 4.06 ▷	**115.56 ± 4.24**	138.14 ± 4.83 ▷	**127.99 ± 5.91**	140.31 ± 1.35 ▷	**130.15 ± 1.81**
car price	$39 \times 10^3 \pm 17 \times 10^3$ ▷	$31 \times 10^3 \pm 13 \times 10^3$	$33 \times 10^3 \pm 15 \times 10^2$ ▷	$29 \times 10^3 \pm$ **954.34**	$31 \times 10^3 \pm 299.99$ ▷	$29 \times 10^3 \pm$ **230.53**	$31 \times 10^3 \pm 11 \times 10^2$ ▷	$28 \times 10^3 \pm$ **315.68**
query	463.1 ± 3.52 ▷	**446.86 ± 3.75**	381.83 ± 26.77 ▷	**362.14 ± 27.99**	458.54 ± 35.03 ▷	**446.96 ± 37.74**	482.83 ± 5.34 ▷	**470.27 ± 6.65**
GPU	550.75 ± 1.75 ▷	**427.28 ± 1.15**	444.61 ± 53.13 ▷	**345.51 ± 41.13**	505.37 ± 59.56 ▷	**429.13 ± 63.42**	536.81 ± 17.06 ▷	**466.93 ± 15.53**
3d	48.6 ± 0.08 ▷	**45.24 ± 0.08**	40.26 ± 4.2 ▷	**37.06 ± 4.12**	39.53 ± 2.01 ▷	**36.58 ± 2.08**	36.91 ± 3.85 ▷	**31.83 ± 5.27**
Avg. Rank	5.14	2.79	3.64	2.21	5.93	4.29	6.71	5.29

$RMSE$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.04 ± 0.0	0.04 ± 0.0	0.05 ± 0.0	0.05 ± 0.01	0.05 ± 0.0	0.05 ± 0.0
cpusum	22.45 ± 1.88	**19.83 ± 4.49**	**19.75 ± 1.77** ◁	30.3 ± 2.52	**28.65 ± 3.42** ◁	34.96 ± 0.9	**29.83 ± 0.97** ◁	33.56 ± 1.15
elevator	0.04 ± 0.03 ▷	**0.02 ± 0.01**	0.21 ± 0.14 ▷	**0.02 ± 0.0**	0.26 ± 0.02 ▷	**0.02 ± 0.0**	0.22 ± 0.01 ▷	**0.02 ± 0.0**
bike	**7.58 ± 0.62** ◁	13.16 ± 0.91	**37.59 ± 6.29** ◁	86.95 ± 13.73	**140.52 ± 25.07** ◁	188.17 ± 26.27	**150.84 ± 6.42** ◁	195.55 ± 8.3
energy	239.43 ± 4.68 ▷	**178.72 ± 3.74**	244.66 ± 1.41 ▷	**194.69 ± 4.58**	251.36 ± 1.76 ▷	**192.45 ± 2.97**	251.34 ± 0.98 ▷	**188.03 ± 0.85**
calhousing	$15 \times 10^4 \pm 19 \times 10^2$ ▷	$12 \times 10^4 \pm 33 \times 10^2$	$14 \times 10^4 \pm 20 \times 10^2$ ▷	$12 \times 10^4 \pm 13 \times 10^2$	$18 \times 10^4 \pm 13 \times 10^3$ ▷	$15 \times 10^4 \pm 97 \times 10^2$	$18 \times 10^4 \pm 41 \times 10^2$ ▷	$15 \times 10^4 \pm 35 \times 10^2$
gasemission	21.37 ± 0.36 ▷	**16.83 ± 0.42**	20.05 ± 0.75 ▷	**17.17 ± 0.12**	23.32 ± 2.08 ▷	**21.0 ± 1.96**	24.18 ± 0.73 ▷	**21.87 ± 0.66**
mv	**6.14 ± 0.06** ◁	6.67 ± 0.06	**4.77 ± 0.47** ◁	5.29 ± 0.48	**7.4 ± 0.97** ◁	9.84 ± 1.51	**7.71 ± 0.32** ◁	10.4 ± 0.43
fried	4.3 ± 0.08 ▷	**3.14 ± 0.09**	3.83 ± 0.17 ▷	**3.08 ± 0.04**	6.82 ± 0.94 ▷	**6.57 ± 1.0**	7.25 ± 0.23 ▷	**7.02 ± 0.25**
polution	141.18 ± 1.73 ▷	**115.52 ± 1.34**	128.99 ± 4.06 ▷	**110.67 ± 1.69**	138.14 ± 4.83 ▷	**129.1 ± 7.07**	140.31 ± 1.35 ▷	**132.79 ± 1.66**
car price	$39 \times 10^3 \pm 17 \times 10^3$ ◁	$31 \times 10^5 \pm 96 \times 10^5$	$33 \times 10^3 \pm 15 \times 10^2$ ◁	$17 \times 10^6 \pm 17 \times 10^5$	$31 \times 10^3 \pm$ **299.99** ◁	$13 \times 10^6 \pm 81 \times 10^4$	$31 \times 10^3 \pm 11 \times 10^2$ ◁	$11 \times 10^6 \pm 49 \times 10^4$
query	463.1 ± 3.52 ▷	**429.92 ± 3.8**	381.83 ± 26.77 ▷	**355.28 ± 24.03**	458.54 ± 35.03 ▷	**446.22 ± 38.23**	482.83 ± 5.34 ▷	**468.06 ± 7.5**
GPU	550.75 ± 1.75 ▷	**385.82 ± 0.91**	444.61 ± 53.13 ▷	**331.26 ± 27.31**	505.37 ± 59.56 ▷	**389.37 ± 43.62**	536.81 ± 17.06 ▷	**415.41 ± 11.07**
3d	48.6 ± 0.08 ▷	**34.32 ± 0.11**	40.26 ± 4.2 ▷	**28.84 ± 2.75**	39.53 ± 2.01 ▷	**28.9 ± 1.56**	36.91 ± 3.85 ▷	**25.68 ± 3.87**
Avg. Rank	5.0	2.39	3.43	2.75	5.71	4.89	6.43	5.39

TABLE A.7: *RMSE* and $RMSE_\phi$ estimates considering `all` (double-side) rare cases: $thr_\phi = 1.0$.

$RMSE_\phi$ estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.05 ± 0.0	0.05 ± 0.0	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.05 ± 0.01 ▷	0.05 ± 0.01	0.05 ± 0.0	0.05 ± 0.0
cpusum	31.57 ± 2.34 ▷	**21.19 ± 1.53**	27.17 ± 2.52 ▷	**18.65 ± 1.52**	39.16 ± 4.66 ▷	**33.71 ± 4.95**	40.8 ± 1.31 ▷	**35.96 ± 1.09**
elevator	**0.03 ± 0.03**	0.07 ± 0.08	**0.22 ± 0.16** ◁	$65 \times 10^3 \pm 71 \times 10^3$	**0.28 ± 0.02** ◁	$10 \times 10^4 \pm 59 \times 10^2$	**0.24 ± 0.01** ◁	$85 \times 10^3 \pm 36 \times 10^2$
bike	**9.94 ± 1.86**	10.04 ± 1.88	**45.71 ± 7.27** ◁	48.07 ± 7.61	248.65 ± 47.57 ▷	**238.4 ± 45.12**	270.58 ± 11.54 ▷	**258.88 ± 11.04**
energy	250.94 ± 4.73 ▷	**223.35 ± 4.24**	256.78 ± 1.46 ▷	**227.96 ± 1.04**	263.81 ± 1.86 ▷	**235.07 ± 1.96**	264.04 ± 1.02 ▷	**236.47 ± 0.72**
calhousing	$15 \times 10^4 \pm 17 \times 10^2$ ▷	$14 \times 10^4 \pm 18 \times 10^2$	$15 \times 10^4 \pm 21 \times 10^2$ ▷	$14 \times 10^4 \pm 26 \times 10^2$	$19 \times 10^4 \pm 14 \times 10^3$ ▷	$18 \times 10^4 \pm 14 \times 10^3$	$19 \times 10^4 \pm 44 \times 10^2$ ▷	$18 \times 10^4 \pm 44 \times 10^2$
gasemission	24.51 ± 0.28 ▷	**21.05 ± 0.35**	22.81 ± 0.94 ▷	**19.32 ± 0.96**	26.4 ± 2.37 ▷	**23.81 ± 2.76**	27.29 ± 0.87 ▷	**25.04 ± 0.89**
mv	7.36 ± 0.46 ▷	**6.43 ± 0.31**	6.47 ± 0.39 ▷	**5.52 ± 0.34**	15.0 ± 1.9 ▷	**14.19 ± 1.87**	15.79 ± 0.57 ▷	**15.0 ± 0.55**
fried	5.43 ± 0.26 ▷	**4.81 ± 0.26**	4.84 ± 0.23 ▷	**4.24 ± 0.25**	7.85 ± 1.23 ▷	**7.56 ± 1.32**	8.6 ± 0.24 ▷	**8.38 ± 0.25**
polution	245.93 ± 1.83 ▷	**213.4 ± 1.74**	224.6 ± 6.66 ▷	**192.58 ± 6.92**	238.28 ± 7.56 ▷	**210.17 ± 8.81**	241.77 ± 2.29 ▷	**213.18 ± 3.01**
car price	$45 \times 10^3 \pm 357.43$ ▷	$41 \times 10^3 \pm$ **313.93**	$42 \times 10^3 \pm 905.62$ ▷	$41 \times 10^3 \pm 10 \times 10^2$	$42 \times 10^3 \pm 414.93$ ▷	$41 \times 10^3 \pm$ **131.28**	$42 \times 10^3 \pm 223.05$ ▷	$40 \times 10^3 \pm$ **433.43**
query	585.88 ± 6.89 ▷	**568.86 ± 7.49**	500.05 ± 29.78 ▷	**473.3 ± 32.74**	573.43 ± 34.97 ▷	**556.23 ± 38.74**	607.78 ± 1.96 ▷	**591.29 ± 3.11**
GPU	699.71 ± 2.98 ▷	**520.64 ± 1.75**	564.33 ± 67.33 ▷	**420.58 ± 50.05**	638.98 ± 74.45 ▷	**531.86 ± 81.64**	676.48 ± 22.13 ▷	**579.73 ± 20.23**
3d	55.99 ± 0.13 ▷	**52.46 ± 0.14**	46.34 ± 4.86 ▷	**42.89 ± 4.83**	45.48 ± 2.31 ▷	**42.3 ± 2.43**	43.07 ± 3.96 ▷	**37.18 ± 5.81**
Avg. Rank	5.07	2.93	3.93	1.86	6.11	4.14	6.86	5.11

$RMSE_\phi$ estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.05 ± 0.01	0.05 ± 0.01	0.05 ± 0.0	0.05 ± 0.0
cpusum	31.57 ± 2.34 ▷	**19.96 ± 3.82**	**27.17 ± 2.52**	29.97 ± 3.19	39.16 ± 4.66 ▷	**35.19 ± 0.94**	40.8 ± 1.31 ▷	**33.72 ± 1.09**
elevator	0.03 ± 0.03 ▷	**0.01 ± 0.01**	0.22 ± 0.16 ▷	**0.02 ± 0.0**	0.28 ± 0.02 ▷	**0.02 ± 0.0**	0.24 ± 0.01 ▷	**0.02 ± 0.0**
bike	**9.94 ± 1.86** ◁	16.7 ± 3.03	**45.71 ± 7.27** ◁	84.64 ± 12.18	248.65 ± 47.57 ▷	**208.77 ± 32.36**	270.58 ± 11.54 ▷	**220.47 ± 9.28**
energy	250.94 ± 4.73 ▷	**182.61 ± 4.05**	256.78 ± 1.46 ▷	**198.17 ± 4.33**	263.81 ± 1.86 ▷	**196.33 ± 2.7**	264.04 ± 1.02 ▷	**192.32 ± 0.87**
calhousing	$15 \times 10^4 \pm 17 \times 10^2$ ▷	$13 \times 10^4 \pm 36 \times 10^2$	$15 \times 10^4 \pm 21 \times 10^2$ ▷	$13 \times 10^4 \pm 15 \times 10^2$	$19 \times 10^4 \pm 14 \times 10^3$ ▷	$16 \times 10^4 \pm 10 \times 10^3$	$19 \times 10^4 \pm 44 \times 10^2$ ▷	$16 \times 10^4 \pm 35 \times 10^2$
gasemission	24.51 ± 0.28 ▷	**18.96 ± 0.37**	22.81 ± 0.94 ▷	**19.14 ± 0.15**	26.4 ± 2.37 ▷	**23.53 ± 2.34**	27.29 ± 0.87 ▷	**24.52 ± 0.8**
mv	7.36 ± 0.46 ▷	**5.44 ± 1.55**	6.47 ± 0.39 ▷	**4.82 ± 0.22**	15.0 ± 1.9 ▷	**12.16 ± 1.62**	15.79 ± 0.57 ▷	**12.87 ± 0.47**
fried	5.43 ± 0.26 ▷	**3.88 ± 0.25**	4.84 ± 0.23 ▷	**3.96 ± 0.07**	7.85 ± 1.23 ▷	**7.55 ± 1.32**	8.6 ± 0.24 ▷	**8.31 ± 0.27**
polution	245.93 ± 1.83 ▷	**152.71 ± 2.16**	224.6 ± 6.66 ▷	**147.37 ± 1.97**	238.28 ± 7.56 ▷	**161.52 ± 6.01**	241.77 ± 2.29 ▷	**165.37 ± 1.24**
car price	$45 \times 10^3 \pm$ **357.43** ◁	$57 \times 10^3 \pm 15 \times 10^3$	$42 \times 10^3 \pm$ **905.62** ◁	$55 \times 10^3 \pm 73 \times 10^2$	$42 \times 10^3 \pm$ **414.93** ◁	$58 \times 10^3 \pm 23 \times 10^2$	$42 \times 10^3 \pm$ **223.05** ◁	$51 \times 10^3 \pm 14 \times 10^2$
query	585.88 ± 6.89 ▷	**564.32 ± 7.63**	500.05 ± 29.78 ▷	**479.39 ± 29.33**	573.43 ± 34.97 ▷	**563.4 ± 37.91**	607.78 ± 1.96 ▷	**594.3 ± 5.03**
GPU	699.71 ± 2.98 ▷	**429.97 ± 1.09**	564.33 ± 67.33 ▷	**366.01 ± 31.7**	638.98 ± 74.45 ▷	**424.75 ± 46.78**	676.48 ± 22.13 ▷	**454.02 ± 11.02**
3d	55.99 ± 0.13 ▷	**41.06 ± 0.15**	46.34 ± 4.86 ▷	**34.38 ± 3.36**	45.48 ± 2.31 ▷	**34.46 ± 1.91**	43.07 ± 3.96 ▷	**30.62 ± 4.65**
Avg. Rank	5.11	2.32	3.96	2.32	6.25	4.32	6.89	4.82

RMSE estimates for the `chebyOS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.05 ± 0.0	0.05 ± 0.0	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.05 ± 0.01 ▷	0.05 ± 0.01	0.05 ± 0.0	0.05 ± 0.0
cpusum	31.57 ± 2.34 ▷	**21.19 ± 1.53**	27.17 ± 2.52 ▷	**18.65 ± 1.52**	39.16 ± 4.66 ▷	**33.71 ± 4.95**	40.8 ± 1.31 ▷	**35.96 ± 1.09**
elevator	**0.03 ± 0.03**	0.07 ± 0.08	**0.22 ± 0.16** ◁	$65 \times 10^3 \pm 71 \times 10^3$	**0.28 ± 0.02** ◁	$10 \times 10^4 \pm 59 \times 10^2$	**0.24 ± 0.01** ◁	$85 \times 10^3 \pm 36 \times 10^2$
bike	**9.94 ± 1.86**	10.04 ± 1.88	**45.71 ± 7.27** ◁	48.07 ± 7.61	248.65 ± 47.57 ▷	**238.4 ± 45.12**	270.58 ± 11.54 ▷	**258.88 ± 11.04**
energy	250.94 ± 4.73 ▷	**223.35 ± 4.24**	256.78 ± 1.46 ▷	**227.96 ± 1.04**	263.81 ± 1.86 ▷	**235.07 ± 1.96**	264.04 ± 1.02 ▷	**236.47 ± 0.72**
calhousing	$15 \times 10^4 \pm 17 \times 10^2$ ▷	$14 \times 10^4 \pm 18 \times 10^2$	$15 \times 10^4 \pm 21 \times 10^2$ ▷	$14 \times 10^4 \pm 26 \times 10^2$	$19 \times 10^4 \pm 14 \times 10^3$ ▷	$18 \times 10^4 \pm 14 \times 10^3$	$19 \times 10^4 \pm 44 \times 10^2$ ▷	$18 \times 10^4 \pm 44 \times 10^2$
gasemission	24.51 ± 0.28 ▷	**21.05 ± 0.35**	22.81 ± 0.94 ▷	**19.32 ± 0.96**	26.4 ± 2.37 ▷	**23.81 ± 2.76**	27.29 ± 0.87 ▷	**25.04 ± 0.89**
mv	7.36 ± 0.46 ▷	**6.43 ± 0.31**	6.47 ± 0.39 ▷	**5.52 ± 0.34**	15.0 ± 1.9 ▷	**14.19 ± 1.87**	15.79 ± 0.57 ▷	**15.0 ± 0.55**
fried	5.43 ± 0.26 ▷	**4.81 ± 0.26**	4.84 ± 0.23 ▷	**4.24 ± 0.25**	7.85 ± 1.23 ▷	**7.56 ± 1.32**	8.6 ± 0.24 ▷	**8.38 ± 0.25**
polution	245.93 ± 1.83 ▷	**213.4 ± 1.74**	224.6 ± 6.66 ▷	**192.58 ± 6.92**	238.28 ± 7.56 ▷	**210.17 ± 8.81**	241.77 ± 2.29 ▷	**213.18 ± 3.01**
car price	$45 \times 10^3 \pm 357.43$ ▷	$41 \times 10^3 \pm$ **313.93**	$42 \times 10^3 \pm 905.62$ ▷	$41 \times 10^3 \pm 10 \times 10^2$	$42 \times 10^3 \pm 414.93$ ▷	$41 \times 10^3 \pm$ **131.28**	$42 \times 10^3 \pm 223.05$ ▷	$40 \times 10^3 \pm$ **433.43**
query	585.88 ± 6.89 ▷	**568.86 ± 7.49**	500.05 ± 29.78 ▷	**473.3 ± 32.74**	573.43 ± 34.97 ▷	**556.23 ± 38.74**	607.78 ± 1.96 ▷	**591.29 ± 3.11**
GPU	699.71 ± 2.98 ▷	**520.64 ± 1.75**	564.33 ± 67.33 ▷	**420.58 ± 50.05**	638.98 ± 74.45 ▷	**531.86 ± 81.64**	676.48 ± 22.13 ▷	**579.73 ± 20.23**
3d	55.99 ± 0.13 ▷	**52.46 ± 0.14**	46.34 ± 4.86 ▷	**42.89 ± 4.83**	45.48 ± 2.31 ▷	**42.3 ± 2.43**	43.07 ± 3.96 ▷	**37.18 ± 5.81**
Avg. Rank	5.07	2.93	3.93	1.86	6.11	4.14	6.86	5.11

RMSE estimates for the `chebyUS` method.

data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.05 ± 0.0 ▷	**0.04 ± 0.0**	0.05 ± 0.01	0.05 ± 0.01	0.05 ± 0.0	0.05 ± 0.0
cpusum	31.57 ± 2.34 ▷	**19.96 ± 3.82**	**27.17 ± 2.52**	29.97 ± 3.19	39.16 ± 4.66 ▷	**35.19 ± 0.94**	40.8 ± 1.31 ▷	**33.72 ± 1.09**
elevator	0.03 ± 0.03 ▷	**0.01 ± 0.01**	0.22 ± 0.16 ▷	**0.02 ± 0.0**	0.28 ± 0.02 ▷	**0.02 ± 0.0**	0.24 ± 0.01 ▷	**0.02 ± 0.0**
bike	**9.94 ± 1.86** ◁	16.7 ± 3.03	**45.71 ± 7.27** ◁	84.64 ± 12.18	248.65 ± 47.57 ▷	**208.77 ± 32.36**	270.58 ± 11.54 ▷	**220.47 ± 9.28**
energy	250.94 ± 4.73 ▷	**182.61 ± 4.05**	256.78 ± 1.46 ▷	**198.17 ± 4.33**	263.81 ± 1.86 ▷	**196.33 ± 2.7**	264.04 ± 1.02 ▷	**192.32 ± 0.87**
calhousing	$15 \times 10^4 \pm 17 \times 10^2$ ▷	$13 \times 10^4 \pm 36 \times 10^2$	$15 \times 10^4 \pm 21 \times 10^2$ ▷	$13 \times 10^4 \pm 15 \times 10^2$	$19 \times 10^4 \pm 14 \times 10^3$ ▷	$16 \times 10^4 \pm 10 \times 10^3$	$19 \times 10^4 \pm 44 \times 10^2$ ▷	$16 \times 10^4 \pm 35 \times 10^2$
gasemission	24.51 ± 0.28 ▷	**18.96 ± 0.37**	22.81 ± 0.94 ▷	**19.14 ± 0.15**	26.4 ± 2.37 ▷	**23.53 ± 2.34**	27.29 ± 0.87 ▷	**24.52 ± 0.8**
mv	7.36 ± 0.46 ▷	**5.44 ± 1.55**	6.47 ± 0.39 ▷	**4.82 ± 0.22**	15.0 ± 1.9 ▷	**12.16 ± 1.62**	15.79 ± 0.57 ▷	**12.87 ± 0.47**
fried	5.43 ± 0.26 ▷	**3.88 ± 0.25**	4.84 ± 0.23 ▷	**3.96 ± 0.07**	7.85 ± 1.23 ▷	**7.55 ± 1.32**	8.6 ± 0.24 ▷	**8.31 ± 0.27**
polution	245.93 ± 1.83 ▷	**152.71 ± 2.16**	224.6 ± 6.66 ▷	**147.37 ± 1.97**	238.28 ± 7.56 ▷	**161.52 ± 6.01**	241.77 ± 2.29 ▷	**165.37 ± 1.24**
car price	$45 \times 10^3 \pm$ **357.43** ◁	$57 \times 10^3 \pm 15 \times 10^3$	$42 \times 10^3 \pm$ **905.62** ◁	$55 \times 10^3 \pm 73 \times 10^2$	$42 \times 10^3 \pm$ **414.93** ◁	$58 \times 10^3 \pm 23 \times 10^2$	$42 \times 10^3 \pm$ **223.05** ◁	$51 \times 10^3 \pm 14 \times 10^2$
query	585.88 ± 6.89 ▷	**564.32 ± 7.63**	500.05 ± 29.78 ▷	**479.39 ± 29.33**	573.43 ± 34.97 ▷	**563.4 ± 37.91**	607.78 ± 1.96 ▷	**594.3 ± 5.03**
GPU	699.71 ± 2.98 ▷	**429.97 ± 1.09**	564.33 ± 67.33 ▷	**366.01 ± 31.7**	638.98 ± 74.45 ▷	**424.75 ± 46.78**	676.48 ± 22.13 ▷	**454.02 ± 11.02**
3d	55.99 ± 0.13 ▷	**41.06 ± 0.15**	46.34 ± 4.86 ▷	**34.38 ± 3.36**	45.48 ± 2.31 ▷	**34.46 ± 1.91**	43.07 ± 3.96 ▷	**30.62 ± 4.65**
Avg. Rank	5.11	2.32	3.96	2.32	6.25	4.32	6.89	4.82

TABLE A.8: $RMSE_\phi$ estimates considering `left-hand side` rare cases ($thr_\phi = 0.8$).

$RMSE_\phi$ estimates for the `chebyOS` method.								
data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.04 ± 0.0 ▷	**0.03 ± 0.0**	0.04 ± 0.01 ▷	**0.02 ± 0.0**	0.04 ± 0.0 ▷	**0.03 ± 0.01**	0.04 ± 0.0 ▷	**0.03 ± 0.01**
cpusum	3.32 ± 0.11 ▷	**1.33 ± 0.02**	2.88 ± 0.17 ▷	**1.3 ± 0.01**	6.77 ± 1.19 ▷	**2.21 ± 0.3**	7.29 ± 0.3 ▷	**2.35 ± 0.07**
elevator	0.13 ± 0.06	**0.11 ± 0.09**	**0.15 ± 0.03** ◁	$16 \times 10^2 \pm 11 \times 10^2$	**0.13 ± 0.01** ◁	$17 \times 10^2 \pm 106.2$	**0.11 ± 0.0** ◁	$14 \times 10^2 \pm 64.83$
bike	9.93 ± 0.9 ▷	**7.15 ± 1.3**	46.33 ± 6.77	**44.92 ± 6.56**	223.01 ± 41.55 ▷	**144.35 ± 24.15**	240.42 ± 10.54 ▷	**153.22 ± 6.71**
energy	230.12 ± 3.38 ▷	**187.25 ± 2.09**	235.91 ± 0.6 ▷	**192.46 ± 0.73**	241.68 ± 2.39 ▷	**194.53 ± 0.96**	241.88 ± 1.27 ▷	**194.71 ± 0.52**
calhousing	$14 \times 10^4 \pm 19 \times 10^2$ ▷	$10 \times 10^4 \pm 18 \times 10^2$	$13 \times 10^4 \pm 22 \times 10^2$ ▷	$10 \times 10^4 \pm 12 \times 10^2$	$17 \times 10^4 \pm 13 \times 10^3$ ▷	$11 \times 10^4 \pm 50 \times 10^2$	$18 \times 10^4 \pm 35 \times 10^2$ ▷	$11 \times 10^4 \pm 11 \times 10^2$
gasemission	19.19 ± 0.17 ▷	**12.94 ± 0.14**	17.91 ± 0.63 ▷	**12.36 ± 0.27**	20.86 ± 1.78 ▷	**14.32 ± 1.09**	21.6 ± 0.64 ▷	**14.9 ± 0.32**
mv	5.62 ± 0.17 ▷	**0.74 ± 0.02**	4.37 ± 0.37 ▷	**0.61 ± 0.05**	5.93 ± 0.48 ▷	**0.93 ± 0.09**	5.77 ± 0.27 ▷	**0.86 ± 0.3**
fried	3.88 ± 0.02 ▷	**3.64 ± 0.02**	3.42 ± 0.16 ▷	**3.15 ± 0.18**	5.82 ± 0.81 ▷	**4.01 ± 0.37**	6.23 ± 0.18 ▷	**4.19 ± 0.09**
polution	194.58 ± 2.49 ▷	**130.29 ± 2.19**	177.34 ± 5.61 ▷	**120.81 ± 2.59**	189.49 ± 6.12 ▷	**128.25 ± 3.32**	191.92 ± 1.5 ▷	**129.39 ± 0.78**
car price	$35 \times 10^3 \pm 384.03$ ▷	$25 \times 10^3 \pm$ **373.09**	$33 \times 10^3 \pm 575.38$	$31 \times 10^3 \pm 44 \times 10^2$	$34 \times 10^3 \pm 393.88$	$33 \times 10^3 \pm$ **654.55**	$33 \times 10^3 \pm 311.73$ ▷	$31 \times 10^3 \pm$ **589.8**
query	451.03 ± 6.02 ▷	**270.98 ± 3.19**	366.3 ± 28.72 ▷	**223.04 ± 16.52**	403.45 ± 23.28 ▷	**246.39 ± 13.93**	420.29 ± 3.09 ▷	**254.43 ± 3.41**
GPU	600.71 ± 3.41 ▷	**406.95 ± 2.31**	484.93 ± 58.26 ▷	**329.3 ± 39.03**	553.89 ± 65.81 ▷	**392.4 ± 52.03**	588.3 ± 18.67 ▷	**423.71 ± 12.65**
3d	44.01 ± 0.06 ▷	**24.93 ± 0.07**	32.49 ± 10.73 ▷	**18.89 ± 5.57**	42.21 ± 1.89 ▷	**26.38 ± 1.18**	40.41 ± 1.72 ▷	**24.97 ± 1.24**
Avg. Rank	5.54	2.64	4.71	1.89	6.68	3.25	7.39	3.89

$RMSE_\phi$ estimates for the `chebyUS` method.								
data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.04 ± 0.0 ▷	**0.02 ± 0.0**	0.04 ± 0.01 ▷	**0.02 ± 0.0**	0.04 ± 0.0 ▷	**0.02 ± 0.0**	0.04 ± 0.0 ▷	**0.02 ± 0.0**
cpusum	**3.32 ± 0.11** ◁	91.71 ± 0.1	**2.88 ± 0.17** ◁	91.61 ± 0.04	**6.77 ± 1.19** ◁	91.58 ± 0.02	**7.29 ± 0.3** ◁	91.56 ± 0.02
elevator	0.13 ± 0.06	**0.08 ± 0.06**	0.15 ± 0.03 ▷	**0.04 ± 0.01**	0.13 ± 0.01 ▷	**0.03 ± 0.0**	0.11 ± 0.0 ▷	**0.04 ± 0.01**
bike	**9.93 ± 0.9** ◁	13.2 ± 1.74	**46.33 ± 6.77** ◁	69.84 ± 8.59	223.01 ± 41.55 ▷	**123.97 ± 13.91**	240.42 ± 10.54 ▷	**125.46 ± 5.46**
energy	230.12 ± 3.38 ▷	**167.48 ± 1.79**	235.91 ± 0.6 ▷	**181.07 ± 3.28**	241.68 ± 2.39 ▷	**179.1 ± 1.94**	241.88 ± 1.27 ▷	**175.09 ± 0.85**
calhousing	$14 \times 10^4 \pm 19 \times 10^2$ ▷	$86 \times 10^3 \pm 10 \times 10^2$	$13 \times 10^4 \pm 22 \times 10^2$ ▷	$86 \times 10^3 \pm$ **467.64**	$17 \times 10^4 \pm 13 \times 10^3$ ▷	$90 \times 10^3 \pm 16 \times 10^2$	$18 \times 10^4 \pm 35 \times 10^2$ ▷	$91 \times 10^3 \pm$ **555.19**
gasemission	19.19 ± 0.17 ▷	**11.5 ± 0.14**	17.91 ± 0.63 ▷	**11.58 ± 0.09**	20.86 ± 1.78 ▷	**12.89 ± 0.63**	21.6 ± 0.64 ▷	**13.22 ± 0.19**
mv	5.62 ± 0.17 ▷	**0.64 ± 0.1**	4.37 ± 0.37 ▷	**1.13 ± 0.1**	5.93 ± 0.48 ▷	**1.11 ± 0.04**	5.77 ± 0.27 ▷	**0.91 ± 0.32**
fried	3.88 ± 0.02 ▷	**2.93 ± 0.03**	3.42 ± 0.16 ▷	**2.73 ± 0.06**	5.82 ± 0.81 ▷	**3.38 ± 0.25**	6.23 ± 0.18 ▷	**3.52 ± 0.05**
polution	194.58 ± 2.49 ▷	**105.82 ± 3.04**	177.34 ± 5.61 ▷	**102.08 ± 0.98**	189.49 ± 6.12 ▷	**106.82 ± 1.78**	191.92 ± 1.5 ▷	**107.96 ± 0.16**
car price	$35 \times 10^3 \pm 384.03$ ▷	$29 \times 10^3 \pm 28 \times 10^2$	$33 \times 10^3 \pm 575.38$ ▷	$30 \times 10^3 \pm 12 \times 10^2$	$34 \times 10^3 \pm 393.88$ ▷	$29 \times 10^3 \pm$ **545.84**	$33 \times 10^3 \pm 311.73$ ▷	$28 \times 10^3 \pm$ **485.23**
query	451.03 ± 6.02 ▷	**228.16 ± 2.82**	366.3 ± 28.72 ▷	**193.52 ± 11.63**	403.45 ± 23.28 ▷	**211.52 ± 10.29**	420.29 ± 3.09 ▷	**217.15 ± 1.99**
GPU	600.71 ± 3.41 ▷	**385.25 ± 1.8**	484.93 ± 58.26 ▷	**327.16 ± 29.18**	553.89 ± 65.81 ▷	**374.1 ± 38.71**	588.3 ± 18.67 ▷	**399.33 ± 8.63**
3d	44.01 ± 0.06 ▷	**19.0 ± 0.07**	32.49 ± 10.73 ▷	**15.86 ± 2.62**	42.21 ± 1.89 ▷	**20.39 ± 0.73**	40.41 ± 1.72 ▷	**19.5 ± 0.7**
Avg. Rank	5.11	2.29	4.32	3.39	6.32	3.96	7.04	3.57

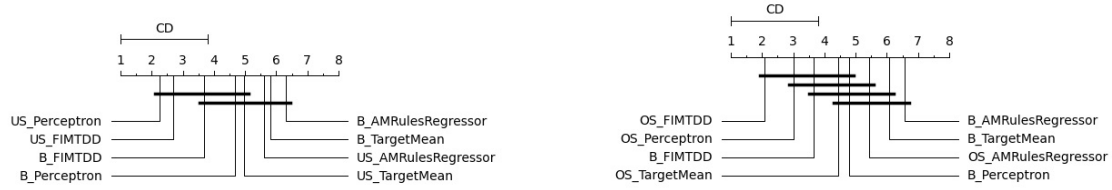
TABLE A.9: $RMSE_\phi$ estimates considering `right-hand side` rare cases ($thr_\phi = 0.8$).

$RMSE_\phi$ estimates for the `chebyOS` method.								
data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS	Baseline	ChebyOS
puma32H	0.04 ± 0.0 ▷	**0.03 ± 0.0**	0.03 ± 0.01 ▷	**0.02 ± 0.0**	0.04 ± 0.0 ▷	**0.02 ± 0.01**	0.04 ± 0.0 ▷	**0.02 ± 0.0**
cpusum	27.37 ± 1.22 ▷	**14.04 ± 0.4**	22.61 ± 1.97 ▷	**13.37 ± 0.36**	33.25 ± 4.23 ▷	**23.97 ± 3.5**	34.99 ± 1.16 ▷	**25.88 ± 0.73**
elevator	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
bike	6.2 ± 2.08 ▷	**1.93 ± 0.14**	32.05 ± 5.37 ▷	**8.89 ± 1.53**	88.24 ± 14.53 ▷	**18.8 ± 2.61**	93.81 ± 4.01 ▷	**19.57 ± 0.83**
energy	55.68 ± 1.68 ▷	**22.73 ± 0.39**	53.84 ± 0.57 ▷	**22.29 ± 0.11**	57.81 ± 1.6 ▷	**23.25 ± 0.36**	58.49 ± 0.56 ▷	**23.33 ± 0.1**
calhousing	$46 \times 10^3 \pm 864.44$ ▷	$31 \times 10^3 \pm$ **424.77**	$46 \times 10^3 \pm 855.59$ ▷	$29 \times 10^3 \pm$ **879.89**	$77 \times 10^3 \pm 93 \times 10^2$ ▷	$35 \times 10^3 \pm 24 \times 10^2$	$81 \times 10^3 \pm 23 \times 10^2$ ▷	$36 \times 10^3 \pm$ **625.17**
gasemission	14.47 ± 0.6 ▷	**10.87 ± 0.18**	14.08 ± 0.19 ▷	**10.57 ± 0.19**	15.71 ± 0.9 ▷	**10.86 ± 0.29**	16.03 ± 0.39 ▷	**10.94 ± 0.14**
mv	5.18 ± 0.16 ▷	**0.04 ± 0.1**	4.41 ± 1.28 ▷	**1.0 ± 1.16**	11.96 ± 1.6 ▷	**6.81 ± 1.0**	12.09 ± 0.48 ▷	**6.95 ± 0.28**
fried	**2.98 ± 0.21**	2.99 ± 0.16	2.88 ± 0.09 ▷	**2.71 ± 0.13**	5.62 ± 0.78 ▷	**3.76 ± 0.37**	5.91 ± 0.22 ▷	**3.88 ± 0.1**
polution	53.97 ± 0.22 ▷	**15.86 ± 0.09**	48.84 ± 1.68 ▷	**15.35 ± 0.13**	56.13 ± 3.15 ▷	**16.14 ± 0.34**	57.0 ± 1.11 ▷	**16.12 ± 0.17**
car price	$22 \times 10^3 \pm 64 \times 10^2$ ▷	$32 \times 10^2 \pm 10 \times 10^2$	$60 \times 10^3 \pm 17 \times 10^3$ ▷	$67 \times 10^2 \pm 18 \times 10^2$	$61 \times 10^3 \pm 36 \times 10^2$ ▷	$58 \times 10^2 \pm$ **308.45**	$53 \times 10^3 \pm 17 \times 10^2$ ▷	$50 \times 10^2 \pm$ **167.69**
query	345.12 ± 1.98 ▷	**207.23 ± 1.39**	285.34 ± 20.14 ▷	**170.98 ± 12.05**	370.06 ± 34.16 ▷	**224.39 ± 21.27**	393.31 ± 5.67 ▷	**234.93 ± 4.91**
GPU	134.41 ± 0.41 ▷	**11.47 ± 0.02**	108.82 ± 12.54 ▷	**9.55 ± 0.96**	116.25 ± 10.56 ▷	**10.72 ± 1.1**	118.66 ± 4.62 ▷	**11.36 ± 0.29**
3d	19.87 ± 0.55 ▷	**8.24 ± 0.69**	16.87 ± 1.56 ▷	**7.19 ± 0.41**	16.3 ± 0.72 ▷	**6.96 ± 0.17**	16.89 ± 0.2 ▷	**7.14 ± 0.05**
Avg. Rank	6.07	2.25	4.89	1.71	6.57	3.64	6.86	4.0

$RMSE_\phi$ estimates for the `chebyUS` method.								
data set	Perceptron		FIMT-DD		TargetMean		AMRules	
	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS	Baseline	ChebyUS
puma32H	0.04 ± 0.0 ▷	**0.02 ± 0.0**	0.03 ± 0.01 ▷	**0.02 ± 0.0**	0.04 ± 0.0 ▷	**0.02 ± 0.0**	0.04 ± 0.0 ▷	**0.02 ± 0.0**
cpusum	27.37 ± 1.22	**19.58 ± 8.93**	**22.61 ± 1.97** ◁	35.28 ± 1.73	**33.25 ± 4.23**	35.1 ± 0.16	34.99 ± 1.16 ▷	**32.2 ± 1.07**
elevator	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
bike	**6.2 ± 2.08** ◁	24.0 ± 3.58	**32.05 ± 5.37** ◁	44.1 ± 5.53	88.24 ± 14.53 ▷	**44.25 ± 2.27**	93.81 ± 4.01 ▷	**38.5 ± 1.36**
energy	55.68 ± 1.68 ▷	**17.73 ± 0.06**	53.84 ± 0.57 ▷	**17.77 ± 0.01**	57.81 ± 1.6 ▷	**17.77 ± 0.01**	58.49 ± 0.56 ▷	**17.79 ± 0.01**
calhousing	$46 \times 10^3 \pm 864.44$ ▷	$24 \times 10^3 \pm$ **703.84**	$46 \times 10^3 \pm 855.59$ ▷	$25 \times 10^3 \pm$ **152.89**	$77 \times 10^3 \pm 93 \times 10^2$ ▷	$28 \times 10^3 \pm 10 \times 10^2$	$81 \times 10^3 \pm 23 \times 10^2$ ▷	$28 \times 10^3 \pm$ **363.34**
gasemission	14.47 ± 0.6 ▷	**9.51 ± 0.18**	14.08 ± 0.19 ▷	**9.65 ± 0.21**	15.71 ± 0.9 ▷	**9.84 ± 0.07**	16.03 ± 0.39 ▷	**9.78 ± 0.09**
mv	5.18 ± 0.16 ▷	**0.07 ± 0.19**	4.41 ± 1.28 ▷	**1.52 ± 0.7**	11.96 ± 1.6 ▷	**5.23 ± 0.71**	12.09 ± 0.48 ▷	**5.29 ± 0.21**
fried	2.98 ± 0.21 ▷	**2.31 ± 0.15**	2.88 ± 0.09 ▷	**2.32 ± 0.03**	5.62 ± 0.78 ▷	**3.12 ± 0.26**	5.91 ± 0.22 ▷	**3.22 ± 0.07**
polution	**53.97 ± 0.22**	65.93 ± 63.68	**48.84 ± 1.68** ◁	71.33 ± 5.3	**56.13 ± 3.15**	57.07 ± 3.33	57.0 ± 1.11 ▷	**48.43 ± 2.03**
car price	$22 \times 10^3 \pm 64 \times 10^2$ ▷	**999.9 ± 2.32**	$60 \times 10^3 \pm 17 \times 10^3$ ▷	$10 \times 10^2 \pm$ **0.52**	$61 \times 10^3 \pm 36 \times 10^2$ ▷	$10 \times 10^2 \pm$ **0.32**	$53 \times 10^3 \pm 17 \times 10^2$ ▷	$10 \times 10^2 \pm$ **0.21**
query	345.12 ± 1.98 ▷	**166.71 ± 1.44**	285.34 ± 20.14 ▷	**146.21 ± 6.89**	370.06 ± 34.16 ▷	**186.07 ± 15.32**	393.31 ± 5.67 ▷	**193.81 ± 3.57**
GPU	134.41 ± 0.41 ▷	**21.91 ± 0.01**	108.82 ± 12.54 ▷	**21.9 ± 0.01**	116.25 ± 10.56 ▷	**21.91 ± 0.0**	118.66 ± 4.62 ▷	**21.91 ± 0.0**
3d	19.87 ± 0.55 ▷	**6.33 ± 0.67**	16.87 ± 1.56 ▷	**5.92 ± 0.17**	16.3 ± 0.72 ▷	**5.84 ± 0.07**	16.89 ± 0.2 ▷	**5.86 ± 0.04**
Avg. Rank	6.07	2.61	4.89	2.57	6.5	3.11	6.82	3.43

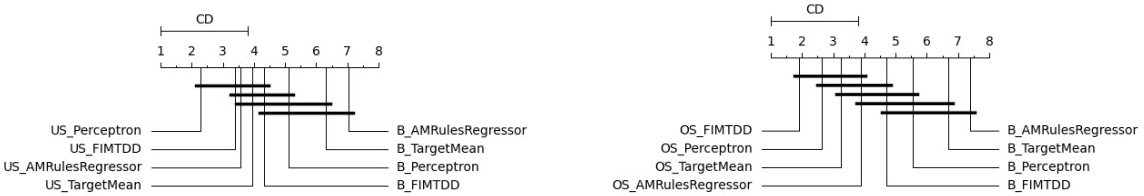
A.3 CD diagramss

CD diagrams for the learners are displayed in Figure A.1. Horizontal bold lines indicate groups of algorithms that are not significantly different (their average ranks differ by less than the CD value). Although learners with sampling strategies rank lower in all cases than their Baseline counterparts, the difference is not significant according to the CD value. This should not mislead us because one item that impacts CD value is the number of data sets contributing to our experiments. Since, in this case, we have 14, which is not big enough, the CD value is not small enough to make a difference in our results.



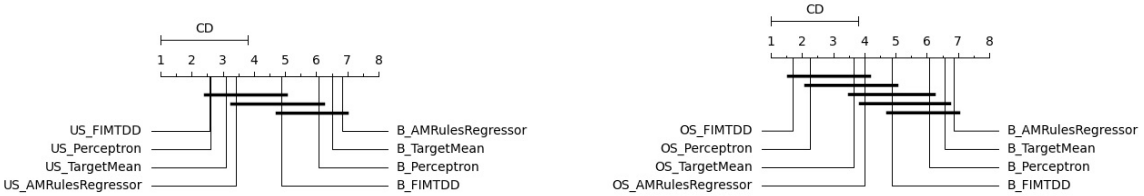
(A) ChebyUS strategy (B) ChebyOS strategy

FIGURE A.1: CD diagrams based on obtained $RMSE_\phi$ results considering both extreme rare cases ($thr_\phi = 0.8$)



(A) ChebyUS strategy (B) ChebyOS strategy

FIGURE A.2: CD diagrams based on obtained $RMSE_\phi$ considering low extreme rare cases ($thr_\phi = 0.8$)



(A) ChebyUS strategy (B) ChebyOS strategy

FIGURE A.3: CD diagrams based on the obtained $RMSE_\phi$ results considering high extreme rare cases ($thr_\phi = 0.8$)

A.4 Target variable information for each dataset

The value of function $\phi()$ to each example of the data sets along with their given probability by Algorithm 3.1, their given K value by Algorithm 3.2 and also the box plot of the data set have been shown in the following figures.

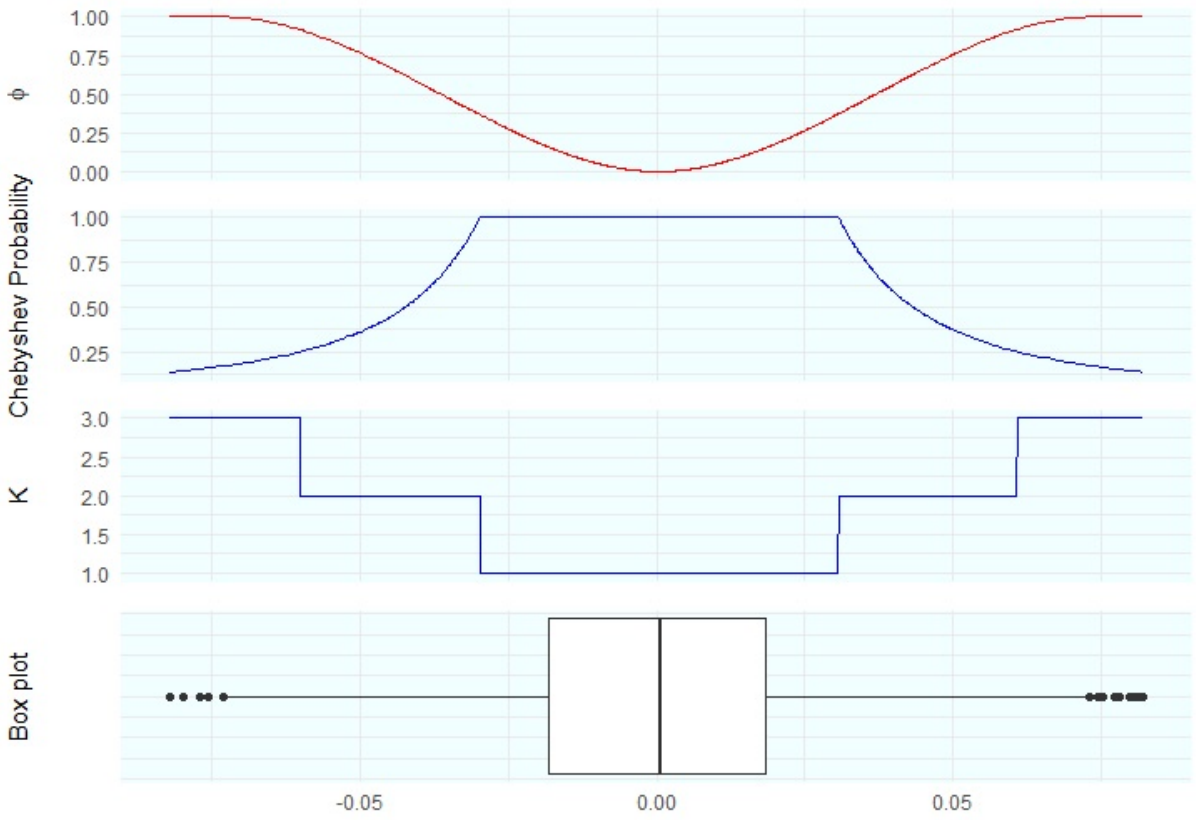


FIGURE A.4: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **puma32H** data set

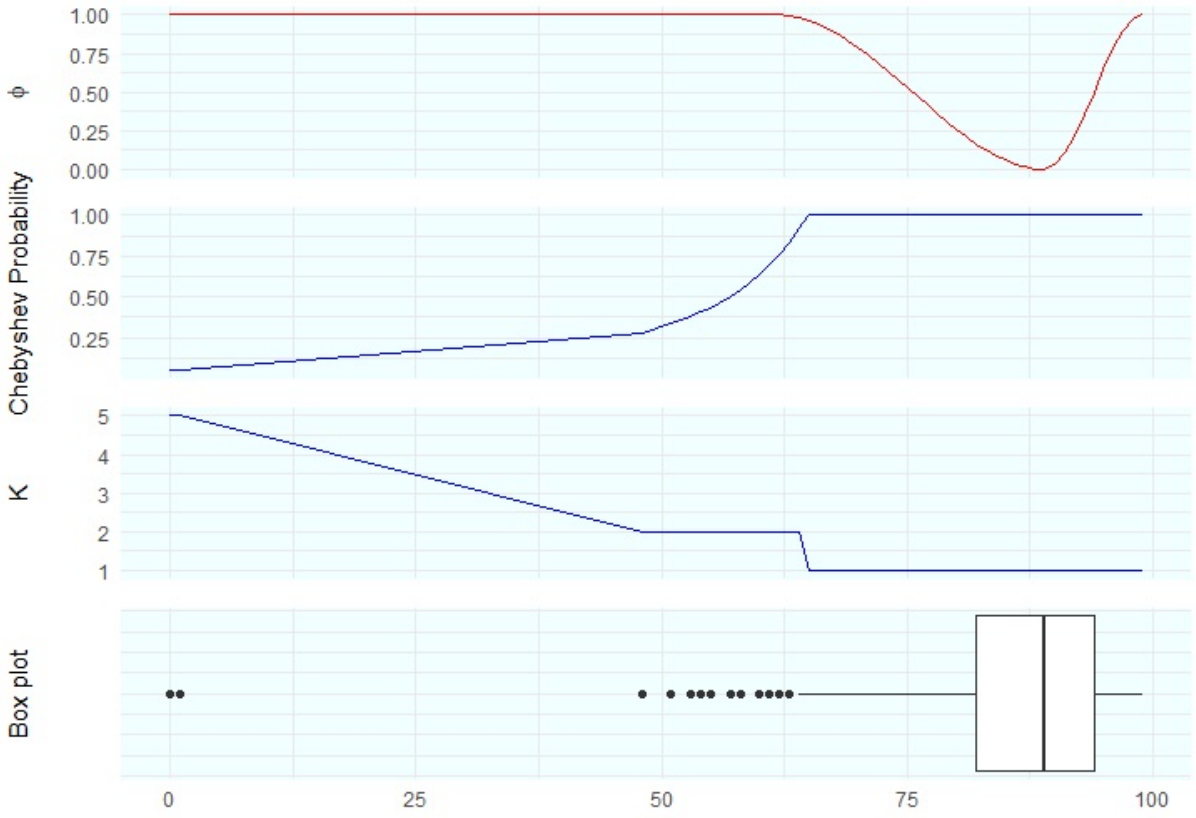


FIGURE A.5: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **cpusum** data set

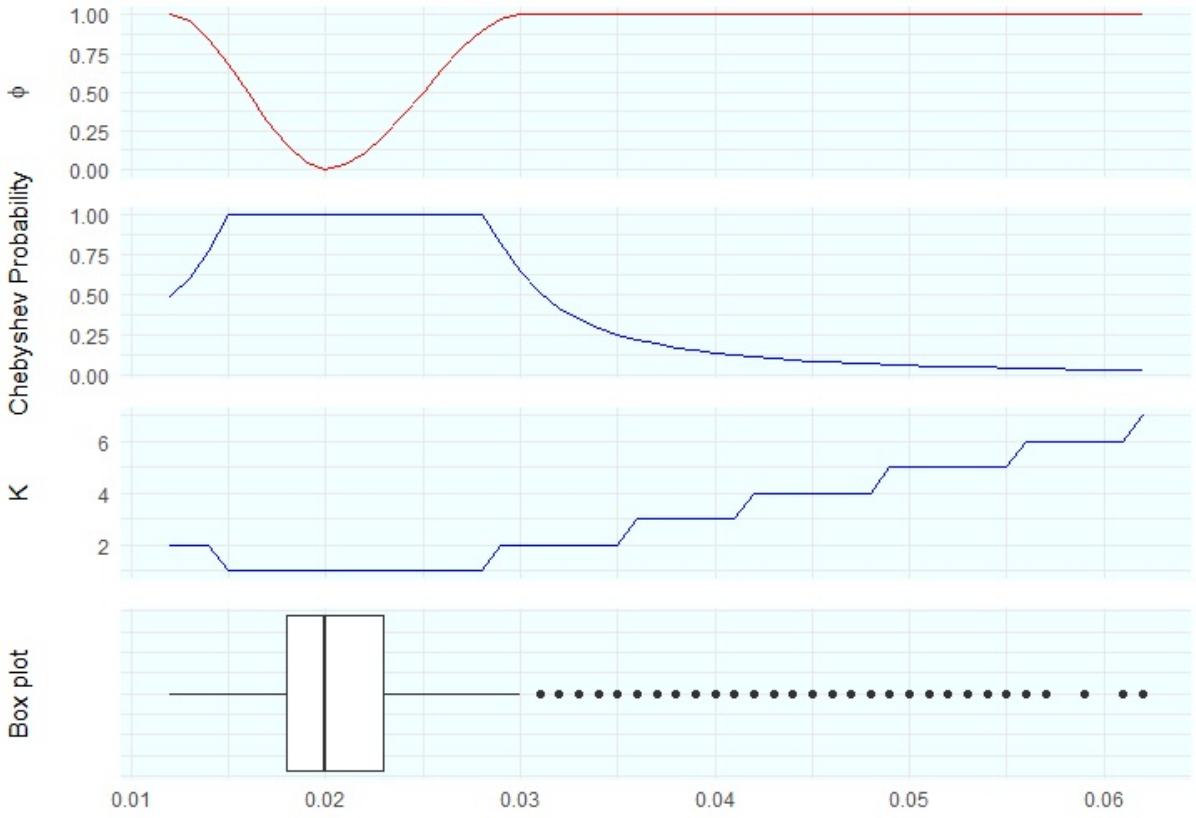


FIGURE A.6: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **elevator** data set

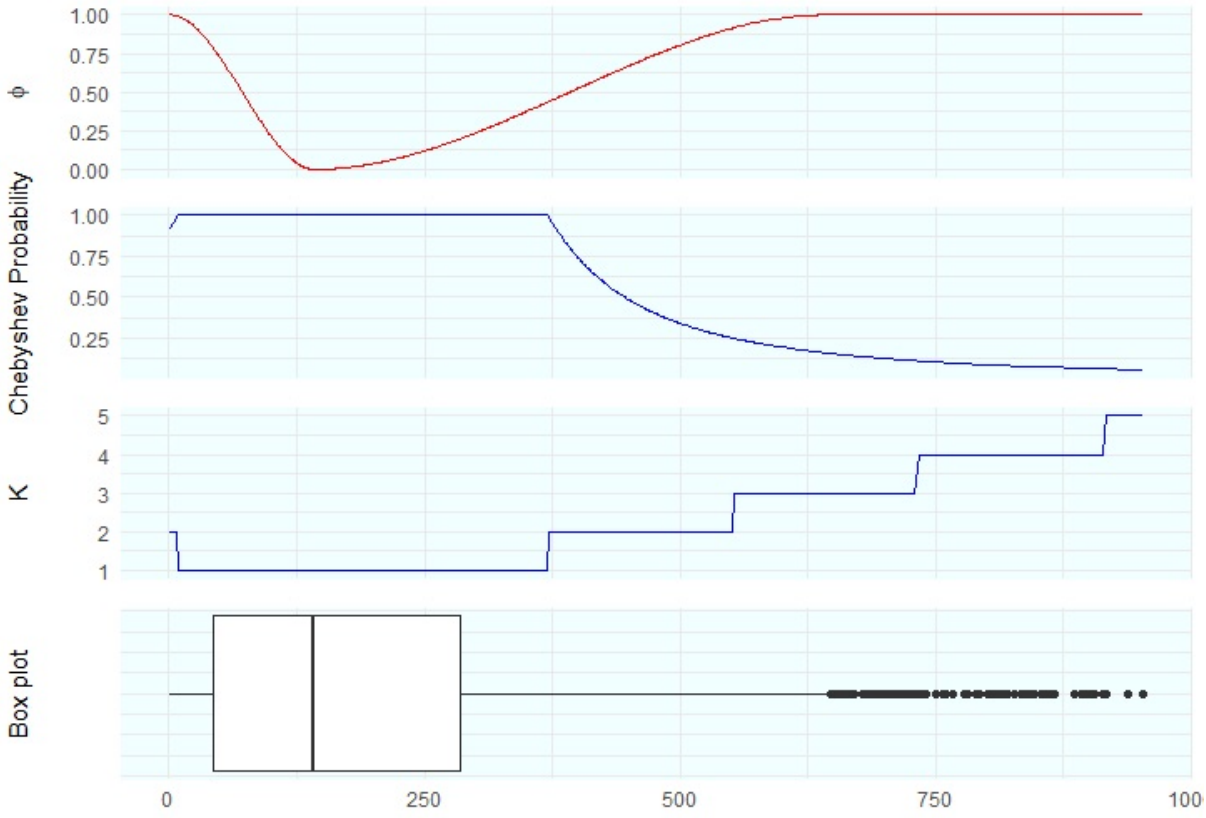


FIGURE A.7: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **bike** data set

FIGURE A.8: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **energy** data set

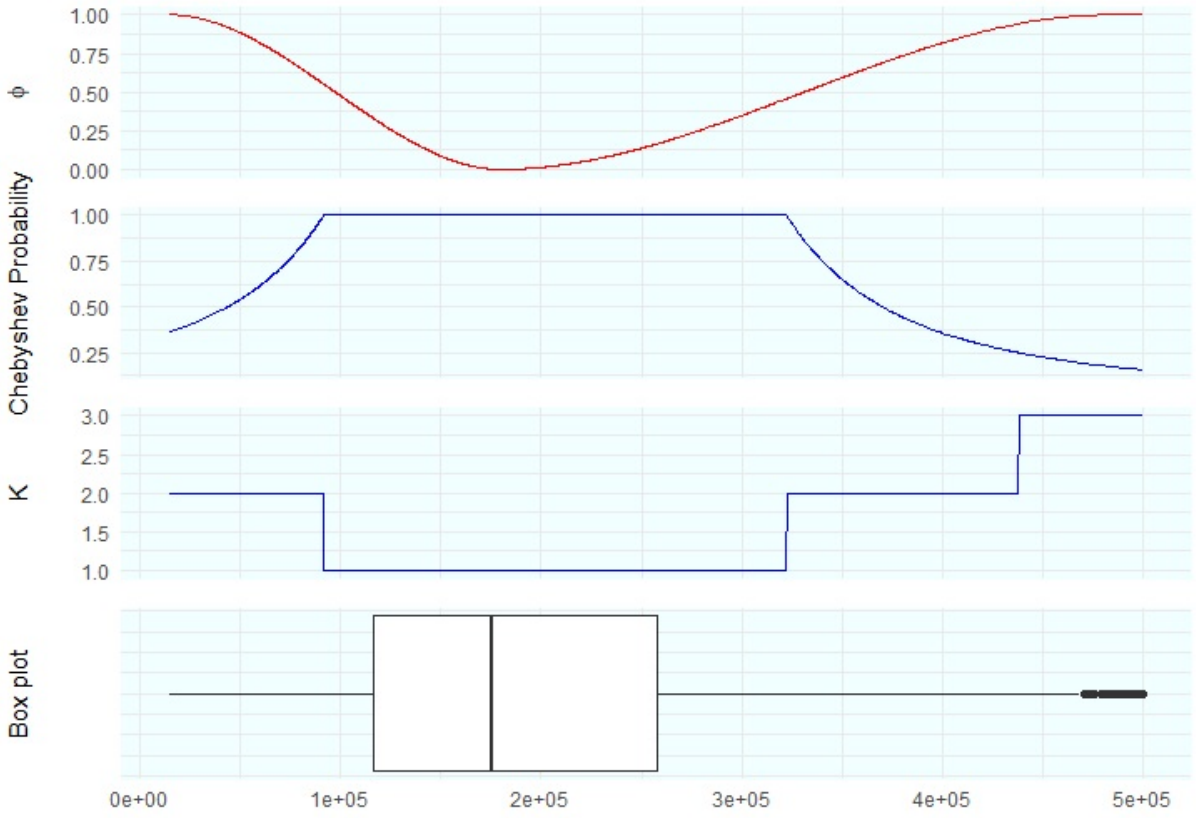


FIGURE A.9: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **calhousing** data set

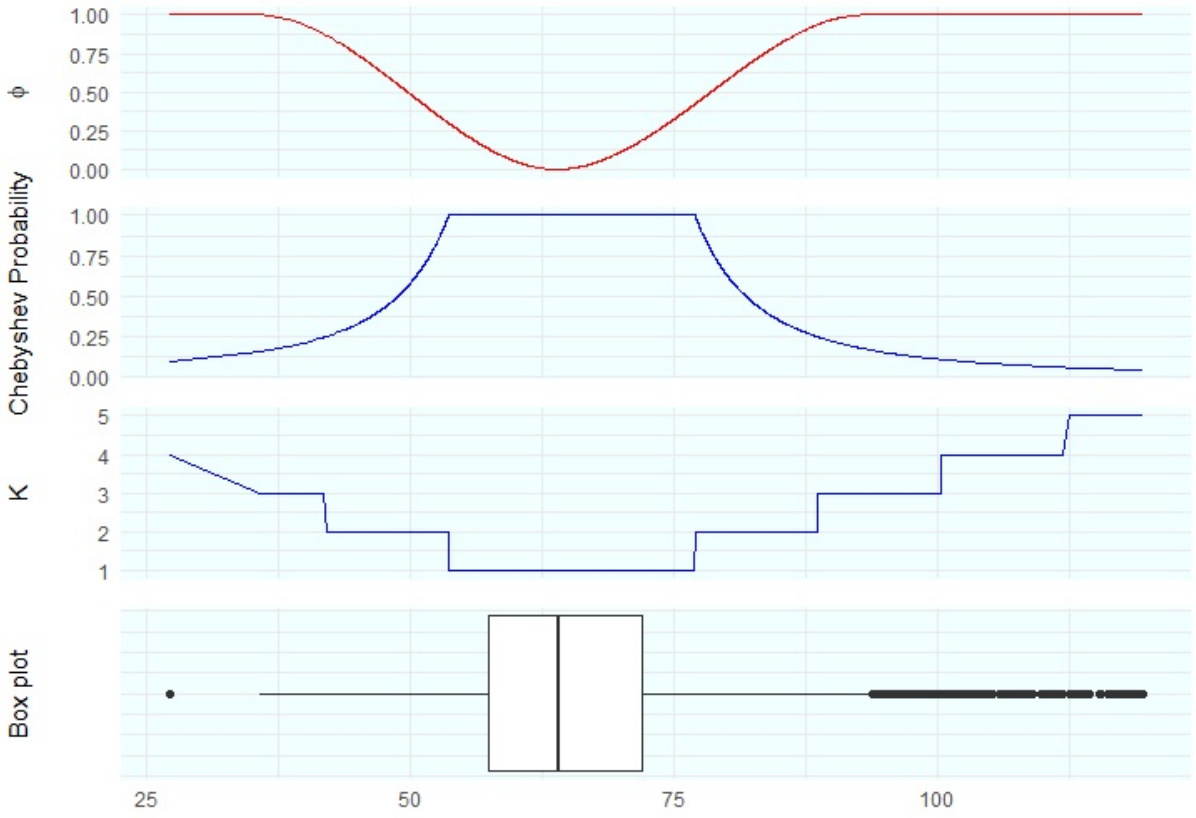


FIGURE A.10: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **gasemission** data set

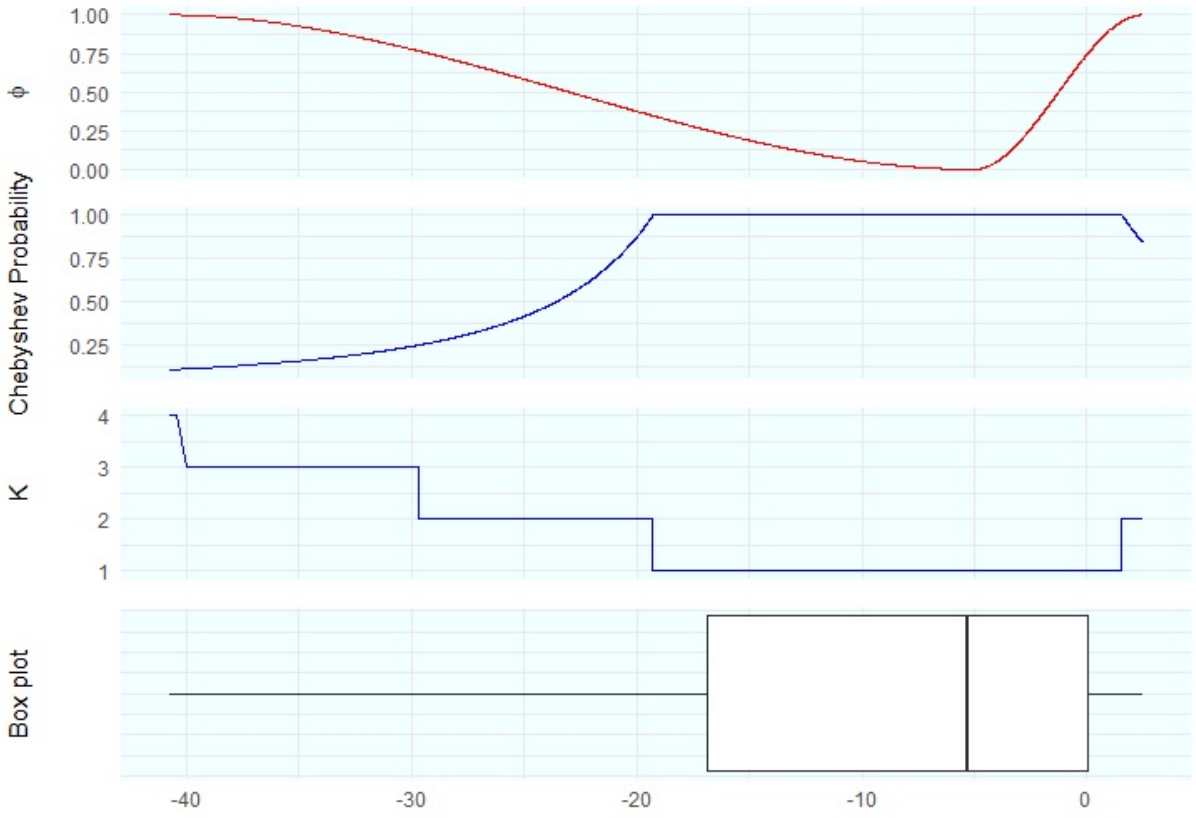


FIGURE A.11: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **mv** data set

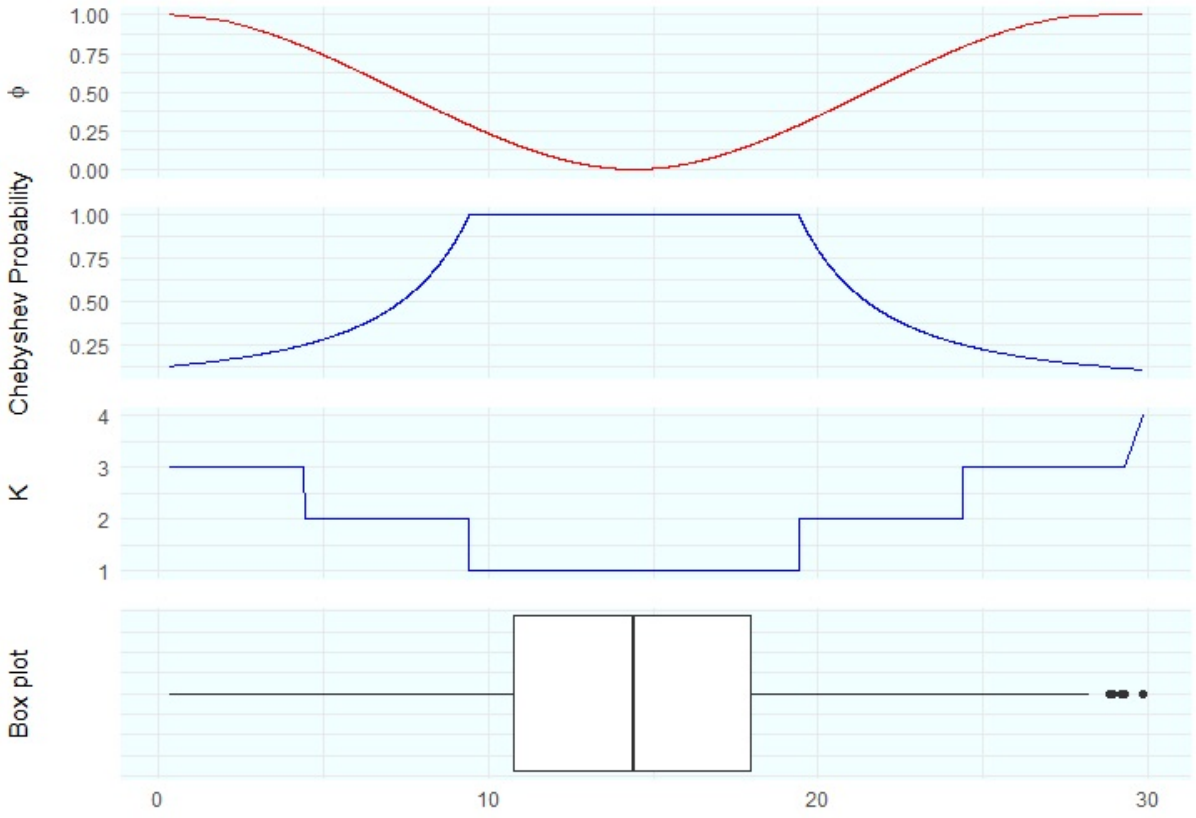


FIGURE A.12: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **fried** data set

FIGURE A.13: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **pollution** data set

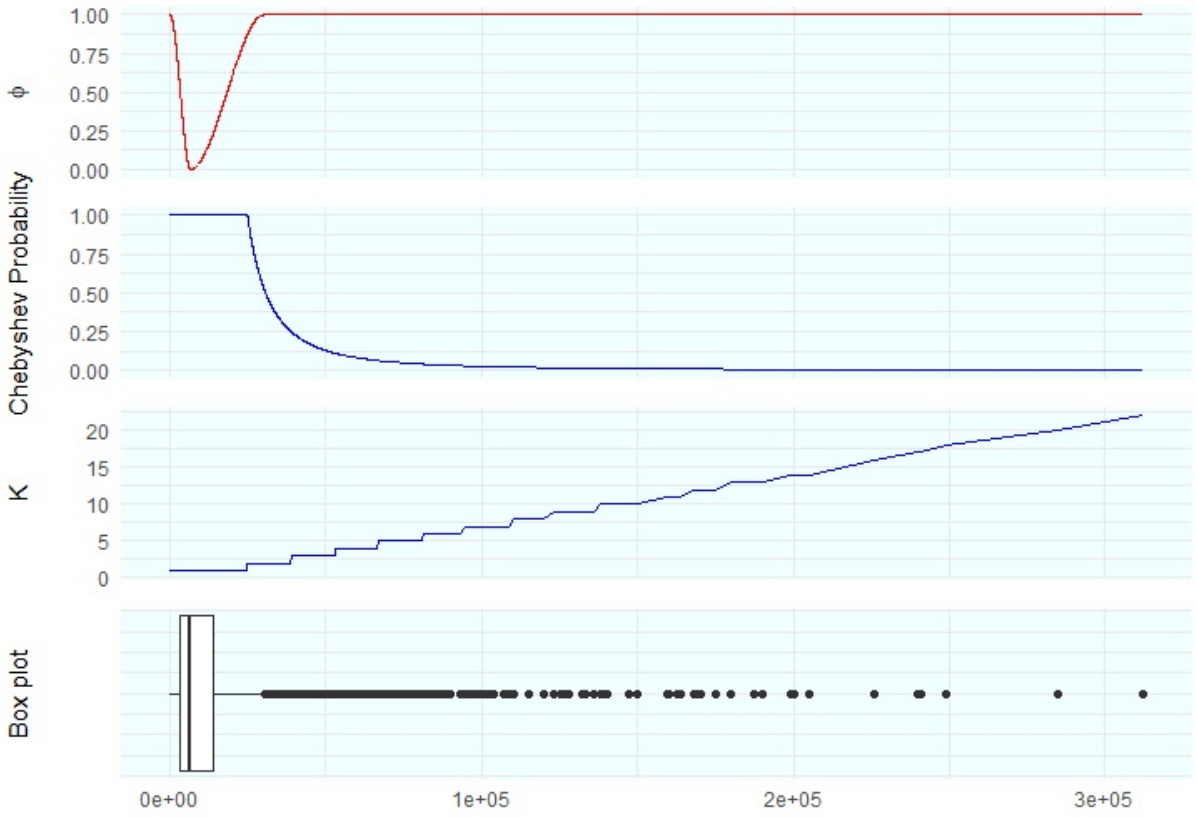


FIGURE A.14: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **car price** data set

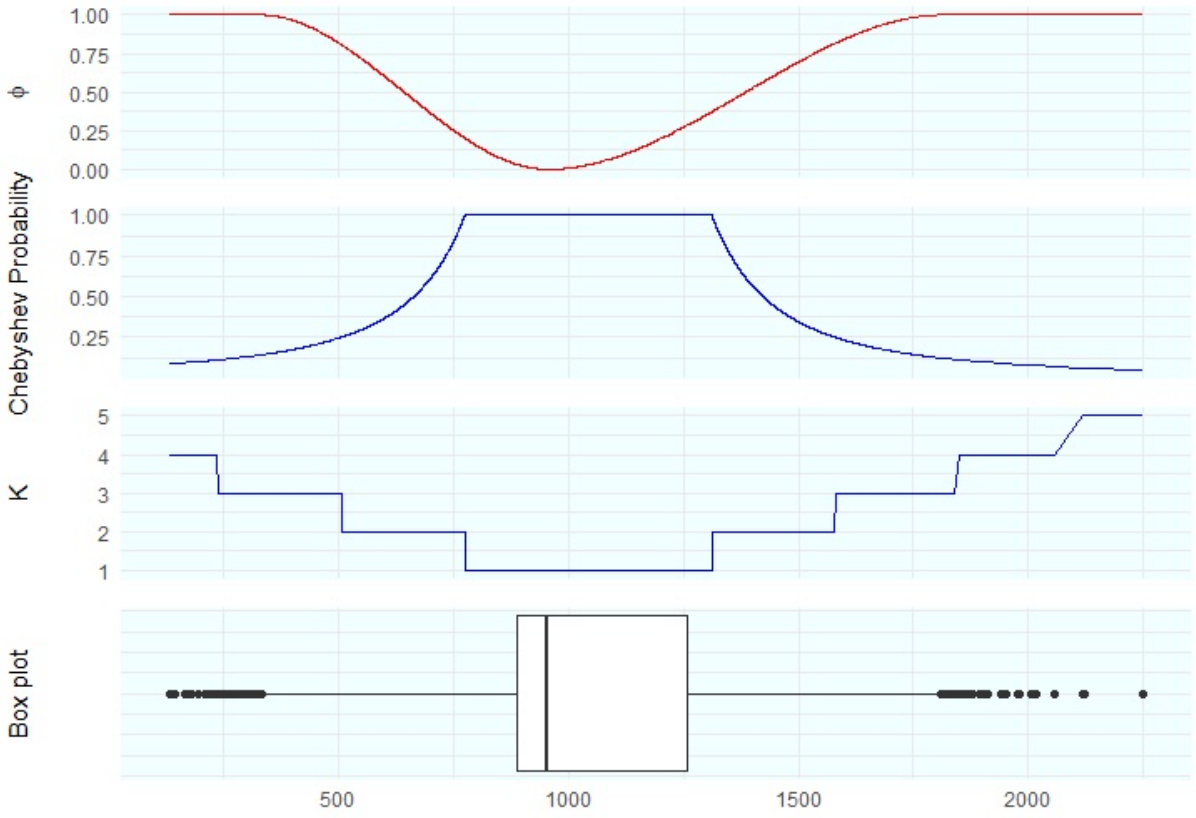


FIGURE A.15: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **query** data set

FIGURE A.16: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **GPU** data set

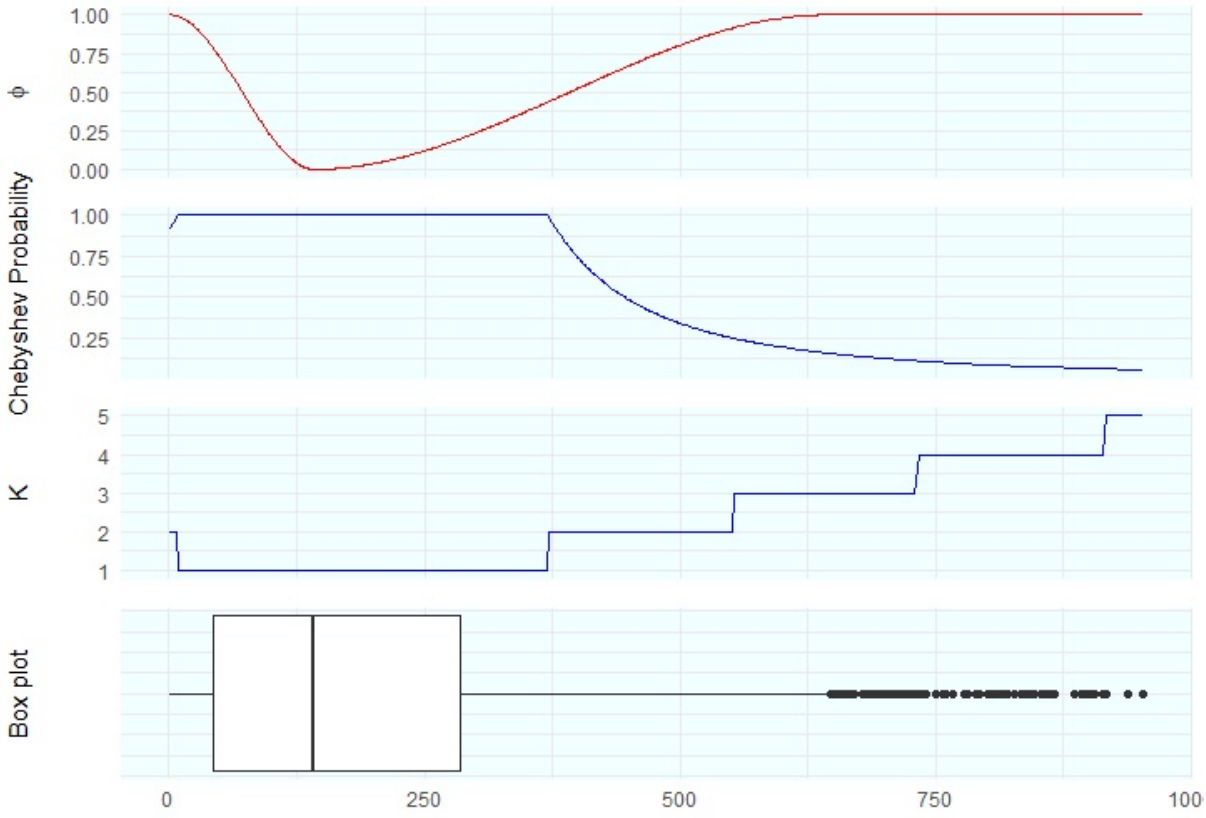


FIGURE A.17: Box plot, K-value, Chebyshev's probability and relevance function $\phi()$ for the target variable of **3d spatial network** data set

A.5 Synthetic Data Generation

The dataset consists of one independent variable x and one dependent variable y, with values of y constrained within the range $[1, 1000]$. To introduce imbalanced data distributions, 95% of the instances were designated as frequent, appearing within two specific intervals, $[150, 350]$ and $[650, 850]$. At the same time, the remaining 5% were classified as rare and distributed across the lower tail $[0, 150]$, the middle range $[350, 650]$, and the upper tail $[850, 1000]$. Different functional relationships were employed for each segment to ensure diverse patterns in the data. The frequent instances followed quadratic relationships, whereas the rare instances were modeled with cubic transformations to introduce non-linearity. The values of x were sampled from uniform distributions tailored to their respective regions, ensuring controlled variability across the dataset.

The functions governing the relationship between the independent variable x and the dependent variable y are defined as follows. For frequent instances, the relationship is modeled using a quadratic function:

$$y = 0.001x^2 + c \tag{A.1}$$

where c is a constant ensuring continuity within the specified frequent intervals.

In contrast, rare instances are modeled using cubic transformations to introduce non-linearity through the function:

$$y = 0.0001x^3 \tag{A.2}$$

for rare instances in the lower tail, and through the function:

$$y = 0.00001(x - 350)^3 + c \tag{A.3}$$

for rare instances in the middle interval, ensuring a clear distinction from frequent cases.

A.6 Range Queries Aggregates Dataset

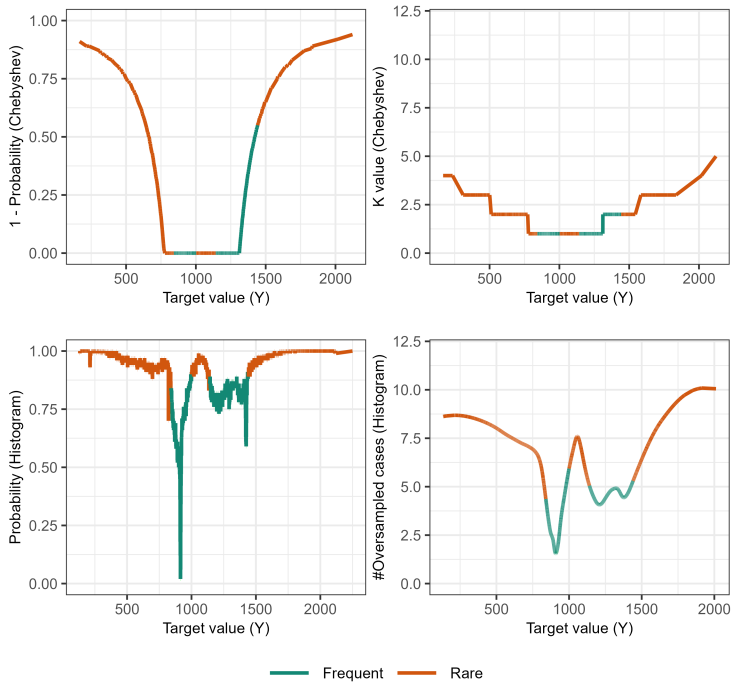
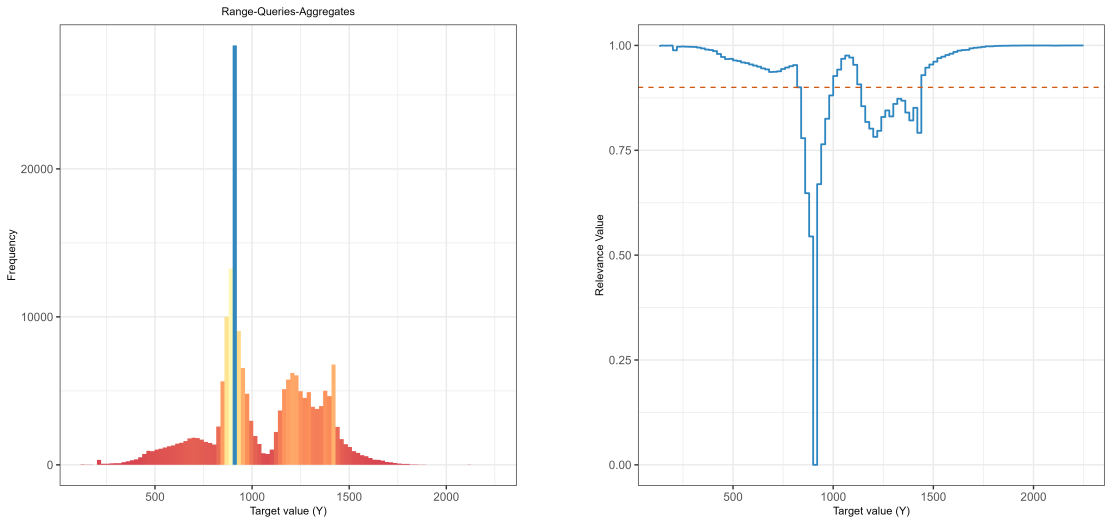


FIGURE A.18: Comparison of Probability and K values between the Chebyshev-based and the histogram-based approaches for the synthetic dataset.

The *Range Queries Aggregates* dataset is a subset of the Query Analytics Workloads Dataset from the UCI Machine Learning Repository. It consists of synthetic range query workloads generated using Gaussian distributions applied to a real dataset. Each query represents a rectangular region in a 2D space, defined by coordinates (X, Y) and corresponding ranges (X-range, Y-range). For each query, the dataset provides aggregate scalar values, including count, sum, and average. To highlight the limitations of Chebyshev-based approaches for this dataset, Figure A.18 presents the derived values. The top two plots show the inverse Chebyshev probability ($1 - \text{Probability}$) and the K values. The inverse probability determines instance selection in ChebyUS, emphasizing rare cases with higher values. Meanwhile, K controls instance replication in ChebyOS, where larger values lead to increased oversampling of rare instances. The bottom plots in the figure depict the probability values and the number of oversampled instances generated by histogram-based approaches. Figures A.19a and A.19b show the histogram and the relevance values assigned to target values, respectively. The dashed orange line indicates the threshold (thr_ϕ) for distinguishing rare cases.



(A) (B)

FIGURE A.19: Histogram of target values (a) and respective relevance values (b) of common and rare cases considering thr_ϕ of 0.9 for "Range Queries Aggregates" dataset

A.7 Benchmark Datasets

Table A.10 provides key dataset statistics, including the number of attributes, examples, and the count and percentage of rare cases for each dataset. Furthermore, To better understand the distribution of target variables and their corresponding relevance values in the datasets, Figures A.20 and A.21 display the histograms of the target value domains and the relevance values assigned to each target value, respectively.

TABLE A.10: Data sets characterization.

Data set	**#Attributes**	**#Examples**	**#Rare cases (%)**
puma32H	34	8,192	211 (%2.58)
cpusum	14	8,192	366 (%4.47)
mv	12	40,768	4,582 (%11.24)
elevator	20	16,599	2,172 (%13.09)
energydata-complete	29	19,735	3,506 (%17.77)
FriedmanArtificialDomain	12	40,768	807 (%1.98)
california	10	20,639	1,272 (%6.16)
bike	18	17,379	3,702 (%21.3)
3d-spatial-network	5	434,873	31,255 (%7.19)
Range-Queries-Aggregates	8	199,843	50,543 (%25.29)
pp-gas-emission	12	36,733	1,823 (%4.96)
GPU-Kernel-Performance	16	241,600	43,019 (%17.81)
pollution	13	41,757	4,693 (%11.24)

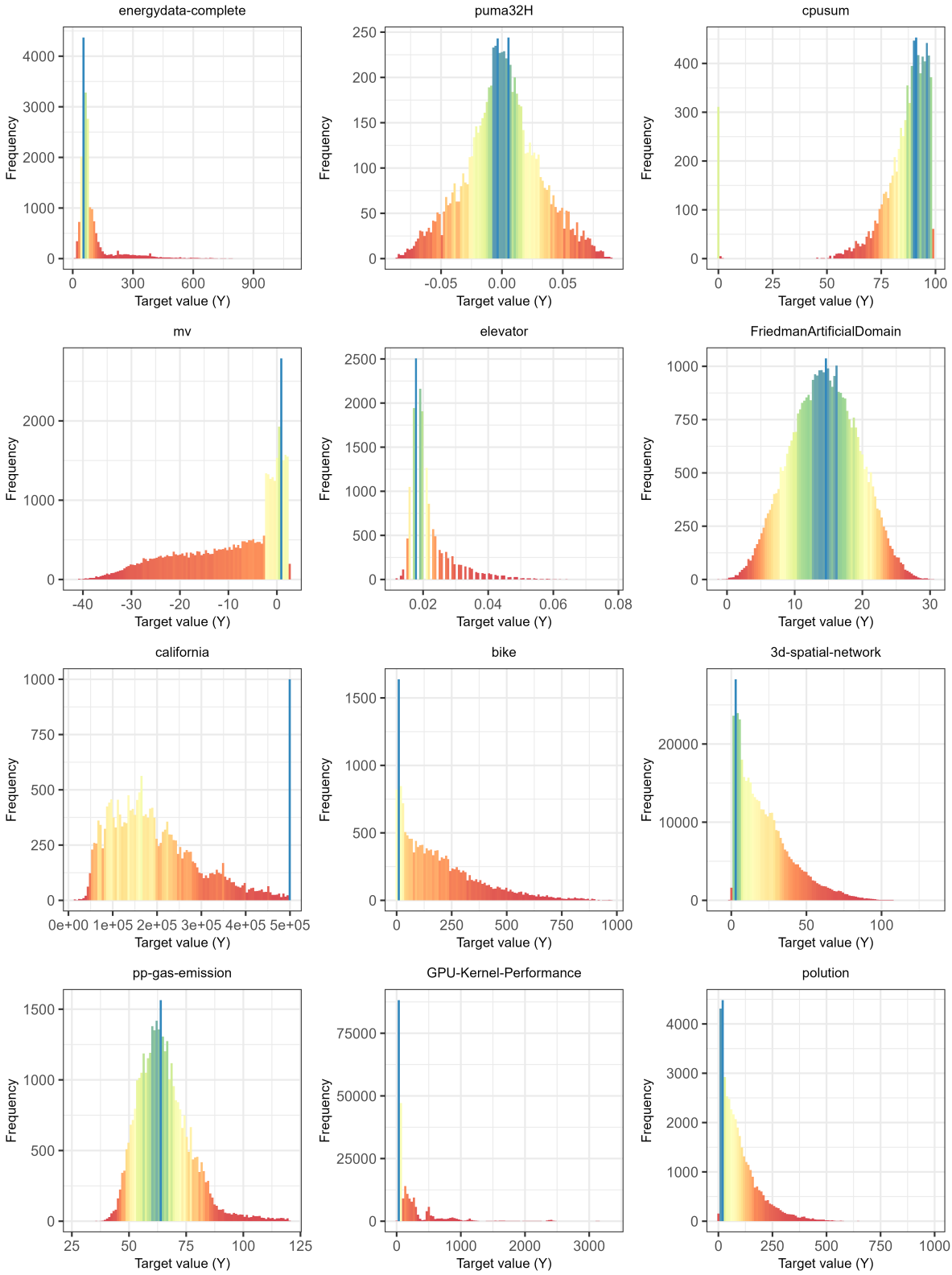


FIGURE A.20: Histograms of the target values for all datasets

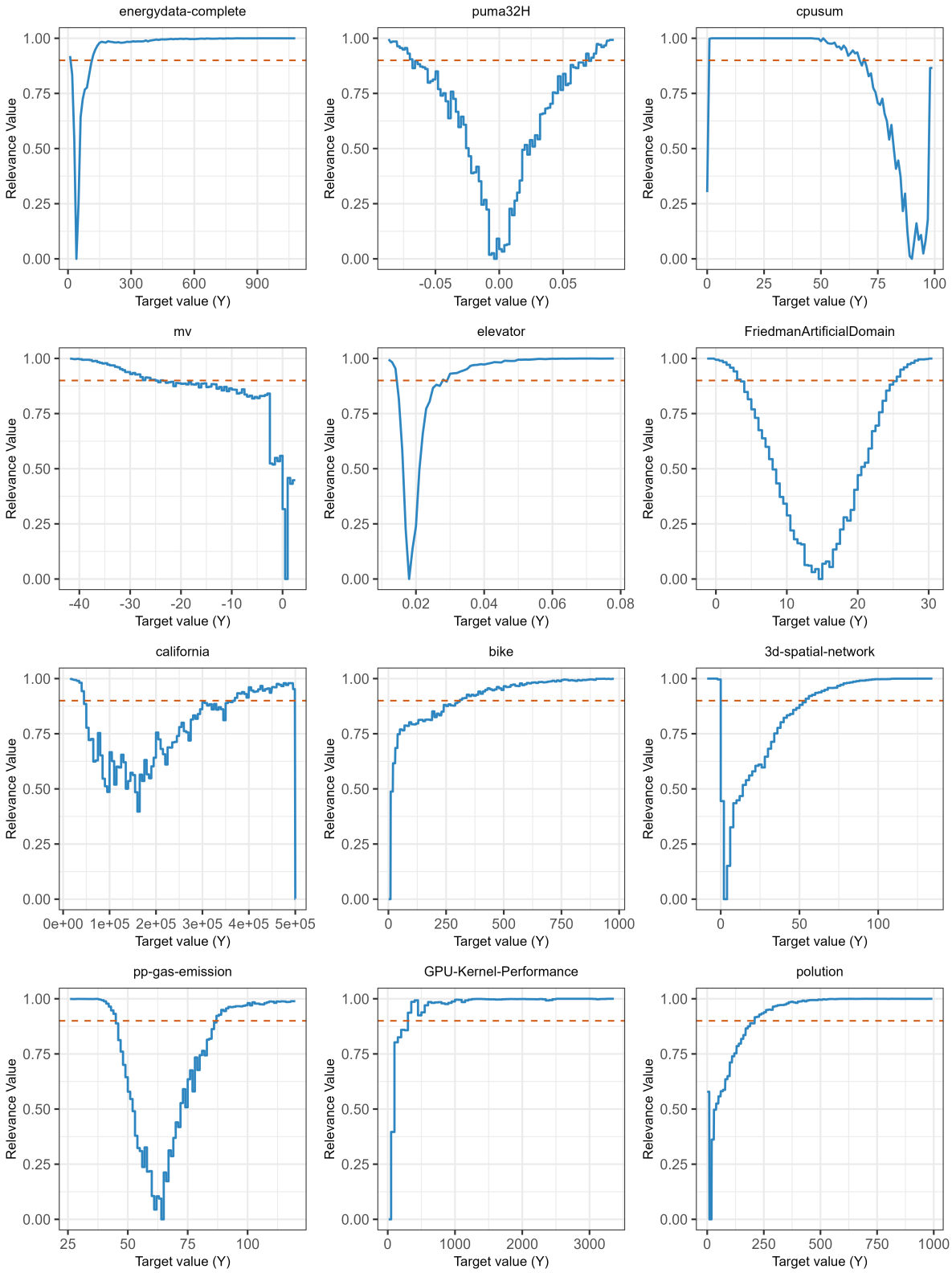


FIGURE A.21: Relevance values of the target variable for all datasets. The dashed orange lines represent the threshold ($thr_\phi = 0.9$) used to distinguish rare cases.

A.8 Additional Results

A comprehensive evaluation of the histogram-based sampling methods, HistUS and HistOS, was performed against their baseline models across 13 datasets over 10 repetitions. To ensure fair comparisons across datasets, the average values were *normalized* by dividing each value by the maximum within its respective dataset.

The results are summarized in Tables A.11 to A.16, covering multiple performance metrics. Tables A.11 and A.12 present the $RMSE_\phi$ evaluation for HistUS and HistOS, respectively, demonstrating their effectiveness in reducing predictive error. Similarly, Tables A.13 and A.14 report the $SERA$ values. The standard $RMSE$ results for both undersampling and oversampling strategies are detailed in Tables A.15 and A.16, providing insights into overall predictive accuracy.

Also, in Figures A.22 to A.24, Critical Difference (CD) diagrams, based on the post-hoc Nemenyi, are used to compare HistUS and HistOS across various result metric functions.

Additionally, direct comparisons between HistUS and HistOS are presented in Tables A.17, A.18, and A.19, and Figure A.25 offering a performance analysis of the two approaches.

Furthermore, the complete results for the Chebyshev-based and histogram-based methods are provided in Tables A.20 to A.23 and Figure A.26, allowing for a more comprehensive evaluation of these sampling strategies.

Table A.11: Evaluation: $RMSE_\phi$, Method: HistUS

	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS
puma32H	0.49 ± 0.30	▷**0.39 ± 0.51**	0.71 ± 0.00	▷**0.57 ± 0.00**	**0.53 ± 1.00**	◁0.70 ± 0.36	1.00 ± 0.00	1.00 ± 0.00
cpusum	**0.55 ± 0.41**	0.56 ± 0.12	**0.42 ± 0.02**	◁0.66 ± 1.00	0.62 ± 0.02	**0.61 ± 0.05**	1.00 ± 0.01	▷**0.73 ± 0.03**
mv	0.05 ± 0.14	▷**0.03 ± 0.04**	0.23 ± 0.01	▷**0.12 ± 0.03**	0.10 ± 1.00	▷**0.06 ± 0.37**	1.00 ± 0.02	▷**0.83 ± 0.02**
elevator	**0.00 ± 0.00**	0.00 ± 0.00	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	▷**0.00 ± 0.00**
energydata-complete	0.96 ± 0.63	▷**0.79 ± 0.42**	0.90 ± 0.23	▷**0.78 ± 0.19**	0.97 ± 0.25	▷**0.85 ± 0.45**	1.00 ± 0.35	▷**0.78 ± 0.71**
FriedmanArtificialDomain	0.30 ± 0.33	▷**0.22 ± 0.88**	0.34 ± 0.21	▷**0.22 ± 0.48**	0.28 ± 1.00	▷**0.23 ± 0.87**	**1.00 ± 0.18**	1.00 ± 0.19
california	0.56 ± 0.31	**0.56 ± 0.48**	0.59 ± 0.22	▷**0.57 ± 0.78**	**0.55 ± 0.37**	0.55 ± 1.00	1.00 ± 0.15	▷**0.98 ± 0.23**
bike	0.03 ± 0.08	**0.03 ± 0.11**	**0.03 ± 0.01**	0.03 ± 0.02	**0.28 ± 0.25**	0.29 ± 0.36	1.00 ± 0.11	▷**0.79 ± 0.13**
3d-spatial-network	0.72 ± 0.81	▷**0.51 ± 0.25**	0.96 ± 0.01	▷**0.66 ± 0.10**	0.35 ± 0.51	▷**0.30 ± 1.00**	1.00 ± 0.00	▷**0.74 ± 0.13**
pp-gas-emission	0.46 ± 0.26	▷**0.40 ± 0.18**	0.62 ± 0.11	▷**0.48 ± 0.08**	0.53 ± 0.27	▷**0.51 ± 1.00**	1.00 ± 0.14	▷**0.88 ± 0.24**
GPU-Kernel-Performance	0.47 ± 0.98	▷**0.42 ± 0.38**	0.68 ± 0.02	▷**0.58 ± 0.03**	0.12 ± 0.25	▷**0.08 ± 0.03**	1.00 ± 0.02	▷**0.86 ± 0.02**
pollution	0.77 ± 0.27	▷**0.59 ± 0.24**	0.83 ± 0.04	▷**0.63 ± 0.03**	0.67 ± 0.98	▷**0.55 ± 0.65**	1.00 ± 0.10	▷**0.76 ± 0.16**
Avg.Rank	3.83	2.42	5.58	3.83	3.92	2.96	7.46	6.0

TABLE A.12: Evaluation: $RMSE_{\phi}$, Method: HistOS

	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS
puma32H	0.49 ± 0.30	**0.43 ± 0.99**	0.71 ± 0.00	▷**0.57 ± 0.00**	0.53 ± 1.00	▷**0.39 ± 0.51**	1.00 ± 0.00	1.00 ± 0.00
cpusum	0.55 ± 0.41	**0.43 ± 0.02**	**0.42 ± 0.02**	0.47 ± 0.29	0.62 ± 0.02	▷**0.41 ± 0.01**	1.00 ± 0.01	▷**0.85 ± 0.01**
mv	0.05 ± 0.14	▷**0.03 ± 0.02**	0.23 ± 0.01	▷**0.09 ± 0.01**	0.10 ± 1.00	▷**0.04 ± 0.16**	1.00 ± 0.02	▷**0.77 ± 0.02**
elevator	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	◁1.00 ± 1.00	0.00 ± 0.00	▷**0.00 ± 0.00**
energydata-complete	0.96 ± 0.63	▷**0.80 ± 0.65**	0.90 ± 0.23	▷**0.78 ± 0.23**	0.97 ± 0.25	▷**0.85 ± 1.00**	1.00 ± 0.35	▷**0.76 ± 0.36**
FriedmanArtificialDomain	0.30 ± 0.33	▷**0.23 ± 0.70**	0.34 ± 0.21	▷**0.26 ± 0.14**	0.28 ± 1.00	▷**0.20 ± 0.95**	**1.00 ± 0.18**	1.00 ± 0.16
california	0.56 ± 0.31	▷**0.50 ± 0.47**	0.59 ± 0.22	▷**0.52 ± 0.20**	0.55 ± 0.37	▷**0.50 ± 0.41**	1.00 ± 0.15	▷**0.90 ± 0.62**
bike	**0.03 ± 0.08**	0.04 ± 0.21	0.03 ± 0.01	**0.03 ± 0.12**	0.28 ± 0.25	**0.26 ± 1.00**	1.00 ± 0.11	▷**0.71 ± 0.12**
3d-spatial-network	0.72 ± 0.81	▷**0.46 ± 0.13**	0.96 ± 0.01	▷**0.70 ± 0.01**	0.35 ± 0.51	▷**0.25 ± 0.77**	1.00 ± 0.00	▷**0.78 ± 0.01**
pp-gas-emission	0.46 ± 0.26	▷**0.39 ± 0.22**	0.62 ± 0.11	▷**0.48 ± 0.09**	0.53 ± 0.27	▷**0.44 ± 0.75**	1.00 ± 0.14	▷**0.91 ± 0.14**
GPU-Kernel-Performance	0.47 ± 0.98	▷**0.39 ± 0.09**	0.68 ± 0.02	▷**0.52 ± 0.03**	0.12 ± 0.25	▷**0.07 ± 1.00**	1.00 ± 0.02	▷**0.81 ± 0.02**
pollution	0.77 ± 0.27	▷**0.55 ± 0.11**	0.83 ± 0.04	▷**0.58 ± 0.04**	0.67 ± 0.98	▷**0.51 ± 1.00**	1.00 ± 0.10	▷**0.75 ± 0.15**
Avg.Rank	4.25	2.58	5.5	3.75	4.33	2.33	7.38	5.88

Table A.13: Evaluation: *SERA*, Method: HistUS

	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS
puma32H	0.75 ± 0.22	▷**0.57 ± 0.18**	0.85 ± 0.00	▷**0.66 ± 0.02**	0.65 ± 1.00	▷**0.50 ± 0.02**	1.00 ± 0.00	1.00 ± 0.00
cpusum	1.00 ± 1.00	▷**0.69 ± 0.06**	**0.66 ± 0.00**	0.66 ± 0.03	0.91 ± 0.07	▷**0.77 ± 0.12**	0.73 ± 0.00	▷**0.60 ± 0.01**
mv	0.64 ± 0.23	▷**0.61 ± 0.26**	0.66 ± 0.00	▷**0.58 ± 0.00**	1.00 ± 0.78	▷**0.88 ± 0.60**	0.94 ± 0.00	▷**0.78 ± 0.00**
elevator	0.91 ± 0.24	**0.84 ± 1.00**	1.00 ± 0.05	▷**0.81 ± 0.77**	0.93 ± 0.19	▷**0.65 ± 0.05**	0.98 ± 0.00	▷**0.41 ± 0.00**
energydata-complete	1.00 ± 0.09	▷**0.75 ± 0.39**	0.98 ± 0.02	▷**0.75 ± 0.32**	0.94 ± 0.28	▷**0.74 ± 1.00**	0.98 ± 0.01	▷**0.68 ± 0.73**
FriedmanArtificialDomain	0.59 ± 0.05	▷**0.44 ± 0.11**	0.61 ± 0.03	▷**0.46 ± 0.06**	0.57 ± 0.11	▷**0.40 ± 1.00**	1.00 ± 0.02	**1.00 ± 0.01**
california	0.98 ± 0.40	▷**0.95 ± 1.00**	**0.94 ± 0.58**	◁0.94 ± 0.37	1.00 ± 0.68	▷**0.98 ± 0.74**	**0.90 ± 0.34**	◁0.90 ± 0.34
bike	1.00 ± 1.00	**0.96 ± 0.49**	0.86 ± 0.07	▷**0.86 ± 0.10**	0.86 ± 0.05	▷**0.82 ± 0.22**	0.89 ± 0.01	▷**0.74 ± 0.02**
3d-spatial-network	0.96 ± 0.07	▷**0.66 ± 1.00**	0.98 ± 0.00	▷**0.57 ± 0.06**	0.97 ± 0.14	▷**0.72 ± 0.54**	1.00 ± 0.00	▷**0.61 ± 0.10**
pp-gas-emission	0.82 ± 0.06	▷**0.61 ± 0.82**	0.85 ± 0.04	▷**0.61 ± 0.69**	0.76 ± 0.50	▷**0.55 ± 0.84**	1.00 ± 0.02	▷**0.78 ± 1.00**
GPU-Kernel-Performance	0.97 ± 0.03	▷**0.86 ± 0.08**	0.92 ± 0.00	▷**0.79 ± 0.00**	0.85 ± 0.59	▷**0.50 ± 0.84**	1.00 ± 0.00	▷**0.70 ± 0.01**
pollution	1.00 ± 0.03	▷**0.80 ± 0.51**	1.00 ± 0.03	▷**0.76 ± 0.12**	0.98 ± 1.00	▷**0.80 ± 0.71**	1.00 ± 0.07	▷**0.67 ± 0.39**
Avg.Rank	6.25	3.58	5.5	2.75	5.33	3.0	6.62	2.96

TABLE A.14: Evaluation: *SERA*, Method: HistOS

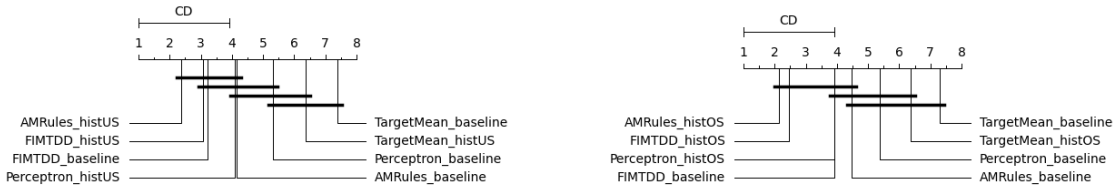
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS
puma32H	0.75 ± 0.22	**0.71 ± 0.07**	0.85 ± 0.00	▷**0.76 ± 0.01**	0.65 ± 1.00	**0.61 ± 0.23**	1.00 ± 0.00	1.00 ± 0.00
cpusum	1.00 ± 1.00	**0.85 ± 0.22**	**0.66 ± 0.00**	0.68 ± 0.11	0.91 ± 0.07	▷**0.88 ± 0.06**	0.73 ± 0.00	▷**0.65 ± 0.00**
mv	0.64 ± 0.23	**0.60 ± 0.35**	0.66 ± 0.00	▷**0.57 ± 0.00**	1.00 ± 0.78	▷**0.88 ± 1.00**	0.94 ± 0.00	▷**0.74 ± 0.00**
elevator	0.91 ± 0.24	**0.88 ± 0.88**	1.00 ± 0.05	**0.92 ± 0.54**	0.93 ± 0.19	**0.86 ± 0.62**	0.98 ± 0.00	▷**0.41 ± 0.00**
energydata-complete	1.00 ± 0.09	▷**0.75 ± 0.37**	0.98 ± 0.02	▷**0.74 ± 0.15**	0.94 ± 0.28	▷**0.74 ± 0.87**	0.98 ± 0.01	▷**0.65 ± 0.19**
FriedmanArtificialDomain	0.59 ± 0.05	▷**0.52 ± 0.06**	0.61 ± 0.03	▷**0.51 ± 0.05**	0.57 ± 0.11	▷**0.51 ± 0.16**	**1.00 ± 0.02**	1.00 ± 0.01
california	0.98 ± 0.40	**0.98 ± 0.81**	**0.94 ± 0.58**	◁0.94 ± 0.54	**1.00 ± 0.68**	1.00 ± 1.00	**0.90 ± 0.34**	◁0.91 ± 0.24
bike	1.00 ± 1.00	**1.00 ± 0.56**	0.86 ± 0.07	▷**0.85 ± 0.05**	0.86 ± 0.05	**0.86 ± 0.12**	0.89 ± 0.01	▷**0.70 ± 0.00**
3d-spatial-network	0.96 ± 0.07	▷**0.74 ± 0.30**	0.98 ± 0.00	▷**0.61 ± 0.00**	0.97 ± 0.14	▷**0.84 ± 0.43**	1.00 ± 0.00	▷**0.65 ± 0.01**
pp-gas-emission	0.82 ± 0.06	▷**0.71 ± 0.47**	0.85 ± 0.04	▷**0.66 ± 0.14**	0.76 ± 0.50	▷**0.64 ± 0.62**	1.00 ± 0.02	▷**0.84 ± 0.29**
GPU-Kernel-Performance	0.97 ± 0.03	▷**0.93 ± 0.15**	0.92 ± 0.00	▷**0.71 ± 0.00**	0.85 ± 0.59	▷**0.55 ± 1.00**	1.00 ± 0.00	▷**0.61 ± 0.01**
pollution	1.00 ± 0.03	▷**0.81 ± 0.38**	1.00 ± 0.03	▷**0.75 ± 0.04**	0.98 ± 1.00	▷**0.82 ± 0.18**	1.00 ± 0.07	▷**0.66 ± 0.13**
Avg.Rank	5.92	3.92	5.42	2.75	5.17	3.25	6.46	3.12

Table A.15: Evaluation: *RMSE*, Method: HistUS

	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS
puma32H	**0.37 ± 0.00**	◁0.54 ± 0.20	**0.56 ± 0.00**	◁0.74 ± 0.00	**0.37 ± 0.00**	◁1.00 ± 1.00	0.56 ± 0.00	0.56 ± 0.00
cpusum	**0.35 ± 0.02**	◁0.62 ± 0.05	**0.48 ± 0.00**	◁1.00 ± 1.00	**0.42 ± 0.00**	◁0.51 ± 0.01	**0.79 ± 0.00**	◁0.84 ± 0.00
mv	**0.15 ± 1.00**	0.15 ± 0.17	**0.39 ± 0.00**	◁0.41 ± 0.01	0.14 ± 0.31	▷**0.12 ± 0.07**	**0.90 ± 0.00**	◁0.96 ± 0.01
elevator	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	0.00 ± 0.00
energydata-complete	**0.70 ± 0.03**	◁0.86 ± 0.09	**0.68 ± 0.03**	◁0.85 ± 0.08	**0.76 ± 0.08**	◁0.98 ± 0.34	**0.73 ± 0.02**	◁0.91 ± 0.07
FriedmanArtificialDomain	**0.47 ± 0.01**	◁0.58 ± 0.04	**0.53 ± 0.00**	◁0.58 ± 0.01	**0.45 ± 0.02**	◁0.69 ± 1.00	**1.00 ± 0.00**	1.00 ± 0.00
california	**0.58 ± 0.03**	◁0.59 ± 0.09	**0.60 ± 0.03**	◁0.60 ± 0.02	**0.64 ± 1.00**	0.64 ± 0.50	**0.97 ± 0.03**	◁0.98 ± 0.04
bike	**0.03 ± 0.02**	0.03 ± 0.04	**0.04 ± 0.01**	0.04 ± 0.01	**0.35 ± 0.47**	◁0.39 ± 1.00	**0.86 ± 0.03**	◁0.94 ± 0.07
3d-spatial-network	**0.66 ± 0.08**	◁0.77 ± 1.00	**0.81 ± 0.00**	◁1.00 ± 0.09	**0.39 ± 0.23**	◁0.48 ± 0.83	**0.82 ± 0.00**	◁1.00 ± 0.15
pp-gas-emission	**0.51 ± 0.05**	◁0.61 ± 0.29	**0.62 ± 0.02**	◁0.75 ± 0.47	**0.63 ± 0.15**	◁0.92 ± 1.00	**0.89 ± 0.02**	◁1.00 ± 0.22
GPU-Kernel-Performance	0.45 ± 0.24	▷**0.45 ± 0.21**	**0.64 ± 0.01**	◁0.67 ± 0.01	**0.12 ± 0.16**	0.13 ± 0.15	**0.83 ± 0.01**	◁0.92 ± 0.01
pollution	**0.68 ± 0.05**	◁0.77 ± 0.15	**0.71 ± 0.01**	◁0.82 ± 0.08	**0.63 ± 0.52**	◁0.74 ± 0.44	**0.83 ± 0.02**	◁0.98 ± 0.14
Avg.Rank	2.38	3.83	3.92	5.5	2.79	4.67	5.83	7.08

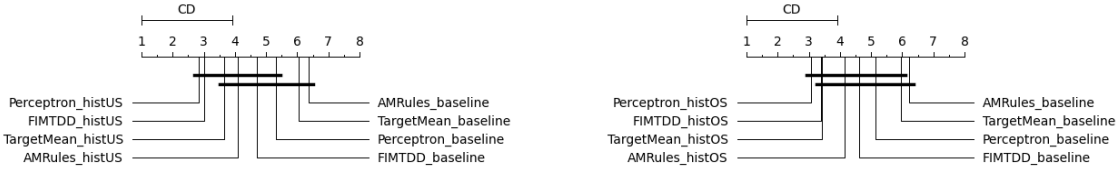
TABLE A.16: Evaluation: *RMSE*, Method: HistOS

	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS	Baseline	HistUS
puma32H	**0.37 ± 0.00**	0.41 ± 0.36	0.56 ± 0.00	0.56 ± 0.00	**0.37 ± 0.00**	◁0.46 ± 0.57	0.56 ± 0.00	0.56 ± 0.00
cpusum	**0.35 ± 0.02**	0.37 ± 0.01	**0.48 ± 0.00**	0.49 ± 0.01	0.42 ± 0.00	▷**0.30 ± 0.00**	**0.79 ± 0.00**	◁0.80 ± 0.00
mv	**0.15 ± 1.00**	0.15 ± 0.51	**0.39 ± 0.00**	◁0.41 ± 0.00	0.14 ± 0.31	▷**0.07 ± 0.03**	**0.90 ± 0.00**	◁1.00 ± 0.01
elevator	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	◁1.00 ± 1.00	0.00 ± 0.00	0.00 ± 0.00
energydata-complete	**0.70 ± 0.03**	◁0.90 ± 0.12	**0.68 ± 0.03**	◁0.89 ± 0.14	**0.76 ± 0.08**	◁1.00 ± 1.00	**0.73 ± 0.02**	◁0.96 ± 0.06
FriedmanArtificialDomain	**0.47 ± 0.01**	◁0.49 ± 0.03	**0.53 ± 0.00**	◁0.54 ± 0.00	**0.45 ± 0.02**	◁0.46 ± 0.03	1.00 ± 0.00	1.00 ± 0.00
california	**0.58 ± 0.03**	◁0.60 ± 0.02	**0.60 ± 0.03**	◁0.61 ± 0.03	**0.64 ± 1.00**	0.65 ± 0.65	**0.97 ± 0.03**	◁1.00 ± 0.07
bike	**0.03 ± 0.02**	0.03 ± 0.07	0.04 ± 0.01	**0.04 ± 0.14**	0.35 ± 0.47	**0.34 ± 0.68**	**0.86 ± 0.03**	◁1.00 ± 0.05
3d-spatial-network	**0.66 ± 0.08**	◁0.71 ± 0.18	**0.81 ± 0.00**	◁0.95 ± 0.01	**0.39 ± 0.23**	0.40 ± 1.00	**0.82 ± 0.00**	◁0.95 ± 0.00
pp-gas-emission	**0.51 ± 0.05**	◁0.53 ± 0.13	**0.62 ± 0.02**	◁0.69 ± 0.07	**0.63 ± 0.15**	◁0.69 ± 0.47	**0.89 ± 0.02**	◁0.96 ± 0.06
GPU-Kernel-Performance	0.45 ± 0.24	▷**0.44 ± 0.09**	**0.64 ± 0.01**	◁0.73 ± 0.02	0.12 ± 0.16	**0.12 ± 1.00**	**0.83 ± 0.01**	◁1.00 ± 0.01
pollution	**0.68 ± 0.05**	◁0.79 ± 0.20	**0.71 ± 0.01**	◁0.86 ± 0.04	**0.63 ± 0.52**	◁0.73 ± 1.00	**0.83 ± 0.02**	◁1.00 ± 0.05
Avg.Rank	2.29	3.67	4.21	5.71	2.88	4.0	6.04	7.21



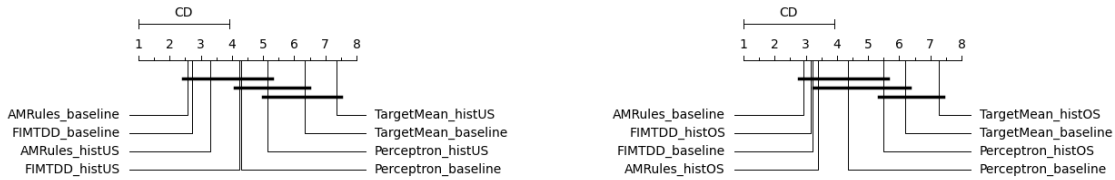
(A) Sampling Strategy: `HistUS` (B) Sampling Strategy: `HistOS`

FIGURE A.22: CD diagrams based on the obtained $RMSE_{\phi}$ results, illustrating the performance comparison between the HistUS (US) and HistOS (OS) sampling strategies and their respective baselines (B).



(A) Sampling Strategy: `HistUS` (B) Sampling Strategy: `HistOS`

FIGURE A.23: CD diagrams based on the obtained *SERA* results, illustrating the performance comparison between the `HistUS` and `HistOS` sampling strategies and their respective baselines.



(A) Sampling Strategy: `HistUS` (B) Sampling Strategy: `HistOS`

FIGURE A.24: CD diagrams based on the obtained *RMSE* results, illustrating the performance comparison between the `HistUS` and `HistOS` sampling strategies and their respective baselines.

TABLE A.17: Evaluation: $RMSE_\phi$, Method: HistUS vs HistOS

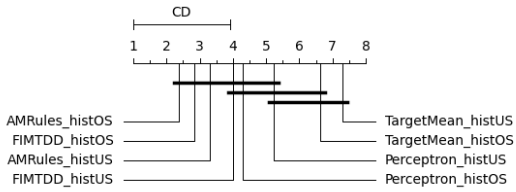
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	HistUS	HistOS	HistUS	HistOS	HistUS	HistOS	HistUS	HistOS
puma32H	**0.39 ± 0.51**	0.43 ± 0.99	0.57 ± 0.00	0.57 ± 0.00	0.70 ± 0.36	▷**0.39 ± 0.51**	1.00 ± 0.00	1.00 ± 0.00
cpusum	0.56 ± 0.12	▷**0.43 ± 0.02**	0.66 ± 1.00	**0.47 ± 0.29**	0.61 ± 0.05	▷**0.41 ± 0.01**	**0.73 ± 0.03**	◁0.85 ± 0.01
mv	**0.03 ± 0.04**	0.03 ± 0.02	0.12 ± 0.03	▷**0.09 ± 0.01**	0.06 ± 0.37	▷**0.04 ± 0.16**	0.83 ± 0.02	▷**0.77 ± 0.02**
elevator	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	◁1.00 ± 1.00	0.00 ± 0.00	0.00 ± 0.00
energydata-complete	**0.79 ± 0.42**	◁0.80 ± 0.65	0.78 ± 0.19	▷**0.78 ± 0.23**	0.85 ± 0.45	**0.85 ± 1.00**	0.78 ± 0.71	▷**0.76 ± 0.36**
FriedmanArtificialDomain	**0.22 ± 0.88**	◁0.23 ± 0.70	**0.22 ± 0.48**	◁0.26 ± 0.14	0.23 ± 0.87	▷**0.20 ± 0.95**	1.00 ± 0.19	1.00 ± 0.16
california	0.56 ± 0.48	▷**0.50 ± 0.47**	0.57 ± 0.78	▷**0.52 ± 0.20**	0.55 ± 1.00	▷**0.50 ± 0.41**	0.98 ± 0.23	▷**0.90 ± 0.62**
bike	**0.03 ± 0.11**	0.04 ± 0.21	0.03 ± 0.02	**0.03 ± 0.12**	0.29 ± 0.36	**0.26 ± 1.00**	0.79 ± 0.13	▷**0.71 ± 0.12**
3d-spatial-network	0.51 ± 0.25	▷**0.46 ± 0.13**	**0.66 ± 0.10**	◁0.70 ± 0.01	0.30 ± 1.00	▷**0.25 ± 0.77**	**0.74 ± 0.13**	◁0.78 ± 0.01
pp-gas-emission	0.40 ± 0.18	**0.39 ± 0.22**	**0.48 ± 0.08**	0.48 ± 0.09	0.51 ± 1.00	▷**0.44 ± 0.75**	**0.88 ± 0.24**	◁0.91 ± 0.14
GPU-Kernel-Performance	0.42 ± 0.38	▷**0.39 ± 0.09**	0.58 ± 0.03	▷**0.52 ± 0.03**	0.08 ± 0.03	**0.07 ± 1.00**	0.86 ± 0.02	▷**0.81 ± 0.02**
pollution	0.59 ± 0.24	▷**0.55 ± 0.11**	0.63 ± 0.03	▷**0.58 ± 0.04**	0.55 ± 0.65	▷**0.51 ± 1.00**	0.76 ± 0.16	▷**0.75 ± 0.15**
Avg.Rank	3.25	3.17	4.71	4.38	4.42	2.83	6.83	6.42

Table A.18: Evaluation: *SERA*, Method: HistUS vs HistOS

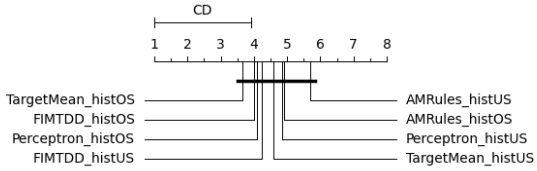
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	HistUS	HistOS	HistUS	HistOS	HistUS	HistOS	HistUS	HistOS
puma32H	**0.57 ± 0.18**	◁0.71 ± 0.07	**0.66 ± 0.02**	◁0.76 ± 0.01	**0.50 ± 0.02**	◁0.61 ± 0.23	1.00 ± 0.00	1.00 ± 0.00
cpusum	**0.69 ± 0.06**	◁0.85 ± 0.22	**0.66 ± 0.03**	0.68 ± 0.11	**0.77 ± 0.12**	◁0.88 ± 0.06	**0.60 ± 0.01**	◁0.65 ± 0.00
mv	0.61 ± 0.26	**0.60 ± 0.35**	0.58 ± 0.00	▷**0.57 ± 0.00**	**0.88 ± 0.60**	0.88 ± 1.00	0.78 ± 0.00	▷**0.74 ± 0.00**
elevator	**0.84 ± 1.00**	0.88 ± 0.88	**0.81 ± 0.77**	◁0.92 ± 0.54	**0.65 ± 0.05**	◁0.86 ± 0.62	0.41 ± 0.00	0.41 ± 0.00
energydata-complete	0.75 ± 0.39	**0.75 ± 0.37**	0.75 ± 0.32	▷**0.74 ± 0.15**	0.74 ± 1.00	**0.74 ± 0.87**	0.68 ± 0.73	▷**0.65 ± 0.19**
FriedmanArtificialDomain	**0.44 ± 0.11**	◁0.52 ± 0.06	**0.46 ± 0.06**	◁0.51 ± 0.05	**0.40 ± 1.00**	◁0.51 ± 0.16	**1.00 ± 0.01**	◁1.00 ± 0.01
california	**0.95 ± 1.00**	◁0.98 ± 0.81	**0.94 ± 0.37**	0.94 ± 0.54	**0.98 ± 0.74**	◁1.00 ± 1.00	**0.90 ± 0.34**	◁0.91 ± 0.24
bike	**0.96 ± 0.49**	1.00 ± 0.56	0.86 ± 0.10	**0.85 ± 0.05**	**0.82 ± 0.22**	◁0.86 ± 0.12	0.74 ± 0.02	▷**0.70 ± 0.00**
3d-spatial-network	**0.66 ± 1.00**	◁0.74 ± 0.30	**0.57 ± 0.06**	◁0.61 ± 0.00	**0.72 ± 0.54**	◁0.84 ± 0.43	**0.61 ± 0.10**	◁0.65 ± 0.01
pp-gas-emission	**0.61 ± 0.82**	◁0.71 ± 0.47	**0.61 ± 0.69**	◁0.66 ± 0.14	**0.55 ± 0.84**	◁0.64 ± 0.62	**0.78 ± 1.00**	◁0.84 ± 0.29
GPU-Kernel-Performance	**0.86 ± 0.08**	◁0.93 ± 0.15	0.79 ± 0.00	▷**0.71 ± 0.00**	**0.50 ± 0.84**	0.55 ± 1.00	0.70 ± 0.01	▷**0.61 ± 0.01**
pollution	**0.80 ± 0.51**	◁0.81 ± 0.38	0.76 ± 0.12	▷**0.75 ± 0.04**	**0.80 ± 0.71**	◁0.82 ± 0.18	0.67 ± 0.39	▷**0.66 ± 0.13**
Avg.Rank	4.75	6.42	3.75	4.33	3.75	5.67	3.67	3.67

TABLE A.19: Evaluation: *RMSE*, Method: HistUS vs HistOS

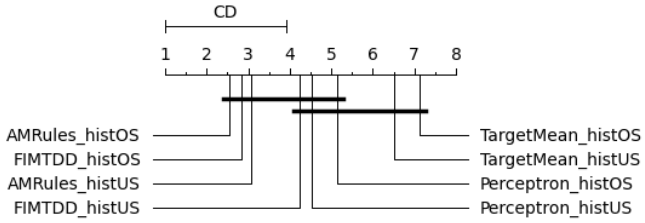
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	HistUS	HistOS	HistUS	HistOS	HistUS	HistOS	HistUS	HistOS
puma32H	0.54 ± 0.20	▷**0.41 ± 0.36**	0.74 ± 0.00	▷**0.56 ± 0.00**	1.00 ± 1.00	▷**0.46 ± 0.57**	0.56 ± 0.00	0.56 ± 0.00
cpusum	0.62 ± 0.05	▷**0.37 ± 0.01**	1.00 ± 1.00	▷**0.49 ± 0.01**	0.51 ± 0.01	▷**0.30 ± 0.00**	0.84 ± 0.00	▷**0.80 ± 0.00**
mv	0.15 ± 0.17	**0.15 ± 0.51**	**0.41 ± 0.01**	◁0.41 ± 0.00	0.12 ± 0.07	▷**0.07 ± 0.03**	**0.96 ± 0.01**	◁1.00 ± 0.01
elevator	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	0.00 ± 0.00	**0.00 ± 0.00**	◁1.00 ± 1.00	0.00 ± 0.00	0.00 ± 0.00
energydata-complete	**0.86 ± 0.09**	◁0.90 ± 0.12	**0.85 ± 0.08**	◁0.89 ± 0.14	**0.98 ± 0.34**	1.00 ± 1.00	**0.91 ± 0.07**	◁0.96 ± 0.06
FriedmanArtificialDomain	0.58 ± 0.04	▷**0.49 ± 0.03**	0.58 ± 0.01	▷**0.54 ± 0.00**	0.69 ± 1.00	▷**0.46 ± 0.03**	1.00 ± 0.00	**1.00 ± 0.00**
california	**0.59 ± 0.09**	◁0.60 ± 0.02	**0.60 ± 0.02**	◁0.61 ± 0.03	**0.64 ± 0.50**	0.65 ± 0.65	**0.98 ± 0.04**	◁1.00 ± 0.07
bike	**0.03 ± 0.04**	0.03 ± 0.07	0.04 ± 0.01	**0.04 ± 0.14**	0.39 ± 1.00	▷**0.34 ± 0.68**	**0.94 ± 0.07**	◁1.00 ± 0.05
3d-spatial-network	0.77 ± 1.00	▷**0.71 ± 0.18**	1.00 ± 0.09	▷**0.95 ± 0.01**	0.48 ± 0.83	▷**0.40 ± 1.00**	1.00 ± 0.15	▷**0.95 ± 0.00**
pp-gas-emission	0.61 ± 0.29	▷**0.53 ± 0.13**	0.75 ± 0.47	▷**0.69 ± 0.07**	0.92 ± 1.00	▷**0.69 ± 0.47**	1.00 ± 0.22	▷**0.96 ± 0.06**
GPU-Kernel-Performance	0.45 ± 0.21	▷**0.44 ± 0.09**	**0.67 ± 0.01**	◁0.73 ± 0.02	0.13 ± 0.15	**0.12 ± 1.00**	**0.92 ± 0.01**	◁1.00 ± 0.01
pollution	**0.77 ± 0.15**	◁0.79 ± 0.20	**0.82 ± 0.08**	◁0.86 ± 0.04	0.74 ± 0.44	**0.73 ± 1.00**	**0.98 ± 0.14**	◁1.00 ± 0.05
Avg.Rank	3.25	2.75	4.83	4.5	4.33	3.25	6.5	6.58



(A) Evaluation metric: $RMSE_{\phi}$



(B) Evaluation metric: $SERA$



(C) Evaluation metric: $RMSE$

FIGURE A.25: CD diagrams comparing the `HistUS` and `HistOS` methods based on $RMSE_{\phi}$, $SERA$, and $RMSE$ evaluation metrics.

TABLE A.20: Evaluation: $RMSE_\phi$, Methods: ChebyUS VS HistUS

	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	ChebyUS	HistUS	ChebyUS	HistUS	ChebyUS	HistUS	ChebyUS	HistUS
puma32H	0.40 ± 0.44	**0.39 ± 0.51**	0.57 ± 0.00	0.57 ± 0.00	**0.66 ± 0.67**	0.70 ± 0.36	1.00 ± 0.00	1.00 ± 0.00
cpusum	**0.29 ± 0.08**	◁0.56 ± 0.12	**0.45 ± 1.00**	◁0.66 ± 1.00	1.00 ± 0.57	▷**0.61 ± 0.05**	**0.26 ± 0.01**	◁0.73 ± 0.03
mv	0.03 ± 0.17	**0.03 ± 0.04**	**0.07 ± 0.01**	◁0.12 ± 0.03	0.15 ± 0.16	▷**0.06 ± 0.37**	**0.65 ± 0.01**	◁0.83 ± 0.02
elevator	0.42 ± 0.16	**0.00 ± 0.00**	0.32 ± 0.06	▷**0.00 ± 0.00**	0.07 ± 0.00	**0.00 ± 0.00**	0.07 ± 0.00	**0.00 ± 0.00**
energydata-complete	**0.78 ± 0.01**	0.79 ± 0.42	**0.77 ± 0.02**	◁0.78 ± 0.19	0.95 ± 0.40	▷**0.85 ± 0.45**	**0.72 ± 1.00**	0.78 ± 0.71
FriedmanArtificialDomain	0.24 ± 0.19	▷**0.22 ± 0.88**	0.24 ± 0.04	▷**0.22 ± 0.48**	0.24 ± 1.00	**0.23 ± 0.87**	**1.00 ± 0.04**	1.00 ± 0.19
california	**0.45 ± 0.49**	◁0.56 ± 0.48	**0.48 ± 0.76**	◁0.57 ± 0.78	**0.50 ± 1.00**	◁0.55 ± 1.00	**0.78 ± 0.19**	◁0.98 ± 0.23
bike	0.03 ± 0.05	**0.03 ± 0.11**	0.06 ± 0.05	▷**0.03 ± 0.02**	0.49 ± 1.00	▷**0.29 ± 0.36**	**0.59 ± 0.02**	◁0.79 ± 0.13
3d-spatial-network	**0.48 ± 0.52**	◁0.51 ± 0.25	0.67 ± 0.00	▷**0.66 ± 0.10**	**0.29 ± 0.15**	0.30 ± 1.00	**0.73 ± 0.01**	◁0.74 ± 0.13
pp-gas-emission	0.42 ± 0.28	▷**0.40 ± 0.18**	0.51 ± 0.40	▷**0.48 ± 0.08**	0.52 ± 1.00	▷**0.51 ± 1.00**	0.93 ± 0.28	▷**0.88 ± 0.24**
GPU-Kernel-Performance	**0.39 ± 0.40**	◁0.42 ± 0.38	**0.47 ± 0.01**	◁0.58 ± 0.03	0.27 ± 0.35	▷**0.08 ± 0.03**	**0.69 ± 0.01**	◁0.86 ± 0.02
pollution	**0.50 ± 0.31**	◁0.59 ± 0.24	**0.50 ± 0.08**	◁0.63 ± 0.03	**0.49 ± 0.30**	◁0.55 ± 0.65	**0.63 ± 0.03**	◁0.76 ± 0.16
Avg.Rank	2.83	3.08	4.21	4.46	4.71	3.79	5.92	7.0

Table A.21: Evaluation: $RMSE_{\phi}$, Methods: ChebyOS VS HistOS

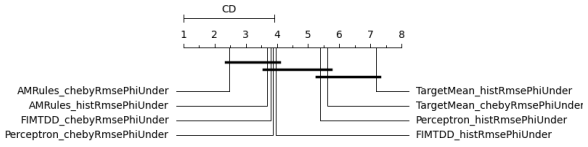
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	ChebyOS	HistOS	ChebyOS	HistOS	ChebyOS	HistOS	ChebyOS	HistOS
puma32H	**0.41 ± 0.81**	0.43 ± 0.99	0.64 ± 0.69	▷**0.57 ± 0.00**	**0.17 ± 0.44**	◁0.39 ± 0.51	1.00 ± 0.00	1.00 ± 0.00
cpusum	**0.16 ± 0.10**	0.43 ± 0.02	**0.18 ± 0.00**	0.47 ± 0.29	**0.20 ± 0.01**	◁0.41 ± 0.01	**0.31 ± 0.03**	◁0.85 ± 0.01
mv	0.04 ± 0.19	▷**0.03 ± 0.02**	0.17 ± 0.01	▷**0.09 ± 0.01**	**0.04 ± 0.18**	0.04 ± 0.16	0.93 ± 0.01	▷**0.77 ± 0.02**
elevator	0.43 ± 0.16	**0.00 ± 0.00**	0.88 ± 0.61	**0.00 ± 0.00**	**0.44 ± 1.00**	1.00 ± 1.00	0.14 ± 0.00	▷**0.00 ± 0.00**
energydata-complete	0.86 ± 0.01	▷**0.80 ± 0.65**	0.83 ± 0.02	▷**0.78 ± 0.23**	0.87 ± 0.02	**0.85 ± 1.00**	0.90 ± 0.03	▷**0.76 ± 0.36**
FriedmanArtificialDomain	0.27 ± 0.12	▷**0.23 ± 0.70**	0.30 ± 0.02	▷**0.26 ± 0.14**	**0.19 ± 0.21**	0.20 ± 0.95	1.00 ± 0.04	1.00 ± 0.16
california	**0.50 ± 0.38**	0.50 ± 0.47	0.54 ± 0.06	▷**0.52 ± 0.20**	**0.49 ± 0.83**	0.50 ± 0.41	0.94 ± 0.08	▷**0.90 ± 0.62**
bike	**0.02 ± 0.04**	0.04 ± 0.21	0.03 ± 0.03	**0.03 ± 0.12**	**0.23 ± 0.31**	◁0.26 ± 1.00	0.92 ± 0.01	▷**0.71 ± 0.12**
3d-spatial-network	0.57 ± 0.09	▷**0.46 ± 0.13**	0.89 ± 0.00	▷**0.70 ± 0.01**	**0.25 ± 0.17**	0.25 ± 0.77	0.95 ± 0.00	▷**0.78 ± 0.01**
pp-gas-emission	0.41 ± 0.07	▷**0.39 ± 0.22**	0.54 ± 0.26	▷**0.48 ± 0.09**	**0.44 ± 0.23**	0.44 ± 0.75	0.97 ± 0.22	▷**0.91 ± 0.14**
GPU-Kernel-Performance	0.42 ± 0.15	▷**0.39 ± 0.09**	0.52 ± 0.01	▷**0.52 ± 0.03**	0.08 ± 1.00	**0.07 ± 1.00**	0.90 ± 0.02	▷**0.81 ± 0.02**
pollution	0.63 ± 0.17	▷**0.55 ± 0.11**	0.71 ± 0.05	▷**0.58 ± 0.04**	0.55 ± 1.00	▷**0.51 ± 1.00**	0.91 ± 0.02	▷**0.75 ± 0.15**
Avg.Rank	3.5	3.0	5.42	4.33	2.92	3.5	7.25	6.08

TABLE A.22: Evaluation: *SERA*, Methods: ChebyUS VS HistUS

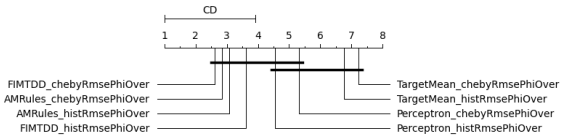
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	ChebyUS	HistUS	ChebyUS	HistUS	ChebyUS	HistUS	ChebyUS	HistUS
puma32H	0.62 ± 0.32	▷**0.57 ± 0.18**	0.70 ± 0.03	▷**0.66 ± 0.02**	0.54 ± 0.61	▷**0.50 ± 0.02**	1.00 ± 0.01	1.00 ± 0.00
cpusum	**0.56 ± 0.12**	◁0.69 ± 0.06	0.74 ± 0.82	**0.66 ± 0.03**	0.89 ± 0.36	▷**0.77 ± 0.12**	**0.52 ± 0.00**	◁0.60 ± 0.01
mv	0.68 ± 0.78	**0.61 ± 0.26**	0.60 ± 0.00	**0.58 ± 0.00**	0.99 ± 0.09	▷**0.88 ± 0.60**	**0.71 ± 0.00**	◁0.78 ± 0.00
elevator	**0.82 ± 1.00**	0.84 ± 1.00	0.89 ± 0.64	**0.81 ± 0.77**	**0.58 ± 0.01**	◁0.65 ± 0.05	**0.41 ± 0.00**	0.41 ± 0.00
energydata-complete	**0.66 ± 0.05**	◁0.75 ± 0.39	**0.67 ± 0.02**	◁0.75 ± 0.32	**0.68 ± 0.03**	◁0.74 ± 1.00	**0.59 ± 1.00**	◁0.68 ± 0.73
FriedmanArtificialDomain	0.47 ± 0.38	▷**0.44 ± 0.11**	0.49 ± 0.02	▷**0.46 ± 0.06**	0.43 ± 1.00	▷**0.40 ± 1.00**	1.00 ± 0.02	**1.00 ± 0.01**
california	0.99 ± 0.40	▷**0.95 ± 1.00**	0.94 ± 0.87	▷**0.94 ± 0.37**	1.00 ± 0.79	▷**0.98 ± 0.74**	0.90 ± 0.10	▷**0.90 ± 0.34**
bike	**0.88 ± 1.00**	◁0.96 ± 0.49	**0.86 ± 0.06**	0.86 ± 0.10	**0.81 ± 0.05**	0.82 ± 0.22	**0.66 ± 0.00**	◁0.74 ± 0.02
3d-spatial-network	0.70 ± 1.00	▷**0.66 ± 1.00**	0.59 ± 0.01	▷**0.57 ± 0.06**	0.80 ± 0.22	▷**0.72 ± 0.54**	**0.60 ± 0.00**	◁0.61 ± 0.10
pp-gas-emission	0.68 ± 1.00	▷**0.61 ± 0.82**	0.65 ± 0.26	▷**0.61 ± 0.69**	0.58 ± 0.46	▷**0.55 ± 0.84**	0.89 ± 0.45	▷**0.78 ± 1.00**
GPU-Kernel-Performance	**0.72 ± 0.13**	◁0.86 ± 0.08	**0.62 ± 0.00**	◁0.79 ± 0.00	0.66 ± 0.27	▷**0.50 ± 0.84**	**0.43 ± 0.00**	◁0.70 ± 0.01
pollution	**0.72 ± 0.38**	◁0.80 ± 0.51	**0.71 ± 0.49**	◁0.76 ± 0.12	**0.74 ± 1.00**	◁0.80 ± 0.71	**0.59 ± 0.02**	◁0.67 ± 0.39
Avg.Rank	4.92	5.42	4.42	4.42	4.83	4.5	3.29	4.21

Table A.23: Evaluation: *SERA*, Methods: ChebyOS VS HistOS

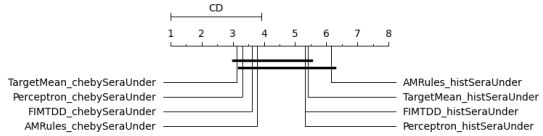
	AMRules		Perceptron		FIMT-DD		TargetMean	
Dataset	ChebyOS	HistOS	ChebyOS	HistOS	ChebyOS	HistOS	ChebyOS	HistOS
puma32H	**0.70 ± 0.32**	0.71 ± 0.07	0.81 ± 0.01	▷**0.76 ± 0.01**	**0.55 ± 0.25**	◁0.61 ± 0.23	1.00 ± 0.01	1.00 ± 0.00
cpusum	0.93 ± 1.00	**0.85 ± 0.22**	**0.60 ± 0.01**	◁0.68 ± 0.11	1.00 ± 0.13	▷**0.88 ± 0.06**	**0.60 ± 0.01**	◁0.65 ± 0.00
mv	0.65 ± 0.55	**0.60 ± 0.35**	0.64 ± 0.00	▷**0.57 ± 0.00**	0.95 ± 0.72	**0.88 ± 1.00**	0.91 ± 0.01	▷**0.74 ± 0.00**
elevator	0.94 ± 0.32	**0.88 ± 0.88**	0.99 ± 0.12	**0.92 ± 0.54**	**0.85 ± 0.16**	0.86 ± 0.62	0.97 ± 0.00	▷**0.41 ± 0.00**
energydata-complete	0.87 ± 0.03	▷**0.75 ± 0.37**	0.86 ± 0.00	▷**0.74 ± 0.15**	0.85 ± 0.09	▷**0.74 ± 0.87**	0.84 ± 0.11	▷**0.65 ± 0.19**
FriedmanArtificialDomain	0.54 ± 0.12	▷**0.52 ± 0.06**	0.55 ± 0.03	▷**0.51 ± 0.05**	**0.50 ± 0.16**	◁0.51 ± 0.16	1.00 ± 0.02	1.00 ± 0.01
california	**0.97 ± 0.35**	◁0.98 ± 0.81	**0.91 ± 0.56**	0.94 ± 0.54	**0.99 ± 1.00**	◁1.00 ± 1.00	**0.87 ± 0.39**	0.91 ± 0.24
bike	**0.97 ± 0.52**	1.00 ± 0.56	0.86 ± 0.05	**0.85 ± 0.05**	0.86 ± 0.20	**0.86 ± 0.12**	0.83 ± 0.01	▷**0.70 ± 0.00**
3d-spatial-network	0.87 ± 0.05	▷**0.74 ± 0.30**	0.87 ± 0.00	▷**0.61 ± 0.00**	0.93 ± 0.13	▷**0.84 ± 0.43**	0.90 ± 0.00	▷**0.65 ± 0.01**
pp-gas-emission	0.76 ± 0.41	▷**0.71 ± 0.47**	0.76 ± 0.23	▷**0.66 ± 0.14**	0.71 ± 0.49	▷**0.64 ± 0.62**	0.96 ± 0.13	▷**0.84 ± 0.29**
GPU-Kernel-Performance	0.96 ± 0.03	▷**0.93 ± 0.15**	0.72 ± 0.00	▷**0.71 ± 0.00**	0.67 ± 1.00	▷**0.55 ± 1.00**	0.80 ± 0.00	▷**0.61 ± 0.01**
pollution	0.90 ± 0.45	▷**0.81 ± 0.38**	0.86 ± 0.04	▷**0.75 ± 0.04**	0.90 ± 0.47	▷**0.82 ± 0.18**	0.86 ± 0.23	▷**0.66 ± 0.13**
Avg.Rank	6.0	4.5	5.08	3.08	5.08	3.58	5.33	3.33



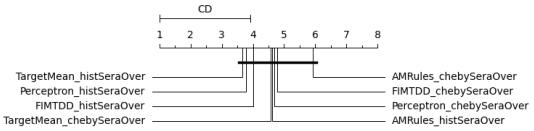
(A) Evaluation: $RMSE_\phi$, Methods: ChebyUS and HistUS



(B) Evaluation: $RMSE_\phi$, Methods: ChebyOS and HistOS



(C) Evaluation: $SERA$, Methods: ChebyUS and HistUS



(D) Evaluation: $SERA$, Methods: ChebyOS and HistOS

FIGURE A.26: CD diagrams comparing ChebyUS, ChebyOS, HistUS, and HistOS methods based on $RMSE_\phi$ and $SERA$ evaluation metrics.