NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

IMBALANCED LEARNING

A comparative study of oversampling and undersampling techniques

Henrique Miguel Pires Marques

Master Thesis

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

IMBALANCED LEARNING

A comparative study of oversampling and undersampling techniques

by

Henrique Miguel Pires Marques

Master Thesis presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialization in Data Science

Supervised by

Professor Fernando José Ferreira Lucas Bação

July, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Lisbon, 15 July 2024

DEDICATION

Para o meu querido avô.

Um homem pelo qual tenho uma enorme admiração, pelo trabalho e sacrifícios que fez durante toda a sua vida.

Ele, que encoraja o neto a estudar e a ir sempre o mais longe possível.

Ele, que transmitiu e continua a transmitir ideais, postura e ensinamentos de vida inigualáveis.

Obrigado por ter herdado tantas das suas características.

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to all those who have supported me throughout this project.

First and foremost, I am deeply grateful to my supervisor, Professor Fernando Bação, for his invaluable guidance, encouragement, and expertise. His insightful feedback and support have been instrumental in completing this work.

I also thank Professor Carlos Soares for his insights and ideas regarding meta-feature analysis, which have significantly improved the conclusions of this work.

I would also like to thank my parents and sister from the bottom of my heart for all the love, encouragement, and sacrifices they have made so that I could reach this milestone in my life.

To my girlfriend, thank you for your patience throughout all the ups and downs over the years. Your love and support have been my constant source of strength.

This work would not have been possible without the support and encouragement of these remarkable individuals. Thank you all for believing in me and for your continued support.

ABSTRACT

Imbalanced data distribution is a recurrent and challenging problem in classification models as most algorithms are designed to assume balanced data. This imbalance often results in poor predictive performance for the minority class, despite an acceptable overall accuracy. A common and easily implementable approach to address this issue is resampling, which can be categorized into oversampling, undersampling, and hybrid methods—a combination of both. However, the effectiveness of these techniques varies based on dataset characteristics such as imbalance ratio, class overlap, and dimensionality. This study evaluates 10 resampling techniques across 35 benchmark datasets from various domains. To mitigate classifier bias, the evaluation employs 4 different classifiers. Unlike many studies focusing on individual resampling types, this research concurrently examines all three categories of resampling methods. Furthermore, the study offers a detailed analysis of average scores and rankings, facilitating a deeper understanding of each technique's relative performance. It also provides specific guidelines for selecting appropriate resampling methods based on the characteristics of each dataset. These findings aim to improve the application of resampling methods, helping practitioners make informed decisions to enhance classification performance in the presence of imbalanced data.

KEYWORDS

Imbalanced Learning; Class Imbalance; Oversampling; Undersampling; Resampling

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

A-SUWO	Adaptive Semi-Unsupervised Weighted Oversampling
AUC	Area Under the Curve
CNN	Condensed Nearest Neighbor
ENN	Edited Nearest Neighbor
FN	False Negatives
FP	False Positives
F1v	Directional-Vector Fisher's Discriminant Ratio
G-SMOTE	Geometric SMOTE
inst2attr	Instances to Attributes Ratio
IR	Imbalance Ratio
KNN	K-Nearest Neighbors
OP	Optimized Precision
OSS	One-Sided Selection
PR	Precision-Recall
RBU	Radial-Based Undersampling
ROC	Receiver Operating Characteristic
SMOTE	Synthetic Minority Oversampling Technique
SOM	Self Organizing Map
SOMO	Self-Organizing Map Oversampling

1. INTRODUCTION

In the world of data science and artificial intelligence, the data deluge has brought an era of exceptional predictive capabilities and intelligent insights. Yet, an important issue keeps challenging experts in the field and happens all too often in real data – the disparity in class distribution. Imbalanced datasets are characterized by a large disproportion in the number of instances in each class. One or more classes – commonly referred to as the minority class(es) – are dimmed by the prevalence of the majority class(es). This skew in class distribution results in a reduction in the predictive power of the different classifiers, resulting in a bias towards the majority class. This problem makes the classifications unreliable and makes it difficult to improve the performance of the classifiers.

In many domains, the use of imbalanced datasets poses a significant challenge, with increasing demands for precise classifications. Healthcare and medical diagnosis stand out as domains where the effective handling of imbalanced datasets has grown in importance. This is especially true in situations when there is a low representation of the class of interest, which makes it difficult for algorithms to produce reliable predictions. For example, the diagnosis of infectious diseases - a domain where an abundance of information about healthy people is compared to a small sample of people who have the disease. In such cases, the imbalance between classes creates substantial hurdles for predictive algorithms.

However, this problem extends beyond the healthcare domain to many other areas. Imbalanced datasets are present in most classification problems across industries, including finance, marketing, and security where incorrect classification could lead to brutal impacts on the business and the security of the organization and its clients. Detecting fraud related to banking transactions, credit cards, virtual payments or even insurance companies (Kraiem et al., 2021) is another major obstacle to improving the quality of the models developed. In this field, and with the widespread use of online transactions, these frauds are becoming a serious threat to financial security (L. Wang et al., 2021). This not only leads to a possible loss of assets by the client but also harms the bank's credibility and causes it to lose assets as well. However, fraud detection, including online fraud, is difficult to predict because legal actions are much more common in this area, which makes the datasets quite imbalanced. Nonetheless, the introduction of machine learning (ML) models in fraud detection has significantly increased the percentage of detection by banking and insurance institutions. In the field of security, which is largely represented by cybersecurity, the same problem exists. It's a critical issue, for both parties, the organization, and its customers, and at the same time, there aren't enough samples in the minority class to train a model capable of successfully classifying these threats.

Minority class is, for the most part, the one that is most important to predict correctly, regarding the domain. This challenge can become even more critical when combined with other difficulty factors in the dataset, such as small disjuncts, the presence of outliers, and an insufficient number of training observations (Koziarski, 2020). Small disjuncts, instances in

which the minority class is sparsely distributed, can make the learning process difficult, so it is hard for models to discern patterns and make accurate predictions. The presence of outliers adds another layer of complexity since they have a significant impact on the model's training phase and make the data sparser, leading to suboptimal generalization. Furthermore, an insufficient number of training observations limits the model's ability to learn robust representations and increases the risk of overfitting to the majority class.

Many standard algorithms perform badly when dealing with imbalanced data because they assume from the outset that the data is balanced and that the misclassification costs are the same. There are, on the other hand, some classification algorithms that tend to be very robust to data imbalance, such as Random Forest, boosting algorithms, and, outside of the traditional machine learning, the neural networks (Tischio & Weiss, n.d.). Still, this usage is more problem-specific and depends on the underlying classifier selected. Assessing classifications in the context of imbalanced data poses challenges, as relying on the broad metric of accuracy can mislead regarding the algorithm's predictive power due to its susceptibility to false positives and negatives. Assuming a basic example in which a classifier obtains an accuracy of 99%, it would seem, at first sight, that it has obtained remarkable results. However, the dataset under study had an imbalance ratio (IR) of 99% and it was predicted zero positives. That's why we will focus on other metrics aside from accuracy, like Area Under the Curve (AUC), G-Mean and F-measure which take class distribution into account (Haixiang et al., 2017).

There are three main approaches to deal with imbalanced learning, each one focuses on applying different techniques, some of them more problem-specific:

- Cost-sensitive learning which is a type of machine learning that considers the costs of misclassification and prioritizes minimizing the overall cost associated with misclassifying instances (Ling & Sheng, 2010). Once again, the proposition arises that we can't just rely on common metrics and the idea that false positives (FP) and false negatives (FN) have the same misclassification error, because they don't (Elkan, 2001). Cost-sensitive learning has three main groups: applications at the level of hyperparameter tuning of algorithms, at the level of data/algorithms such as decision threshold adjustment, and meta-learning (Sterner et al., n.d.).
- Algorithmic modification that reinforces the learning towards the minority class, adapting their structures or incorporating class-weight adjustments to address the class imbalance.
- Data Sampling which is a more general approach, and the one we will focus on, involves resampling techniques, where the training dataset is adjusted to achieve a more balanced class distribution. This can include oversampling techniques, such as SMOTE (Synthetic Minority Over-Sampling Technique), or undersampling methods, such as RUS (Random Under-Sampling), which removes instances from the majority

class (Van Hulse et al., 2007). It is also possible to use a hybrid of these two techniques which will be discussed in a later section.

The importance of this research lies in providing some guidelines on when to use each of the resampling techniques (oversampling, undersampling, or hybrid), considering the problem and the characteristics of the dataset. Nevertheless, given the no-free lunch theorem, it is not to be expected that a given approach will achieve state-of-the-art performance for a given problem (Koziarski, 2020). Experimentation with various techniques is always advisable, although this study allows for a narrow down in the range of appropriate methods for the problem at hand.

At the beginning of this thesis, an extensive literature review is presented in section 2. It will cover the different types of class imbalance, 6 important dataset characteristics that impact the imbalanced classifications and the main evaluation metrics used in these scenarios. Section 2 finishes with a comprehensive overview of multiple resampling strategies, covering oversampling, undersampling and hybrid techniques. Subsequently, section 3 outlines the research methodology with the presentation of the datasets, resampling methods, classifiers, and metrics used. It also explains how the tests were conducted, and all the steps taken to obtain the results. After that, in section 4 the results of application of the 10 resampling strategies over 4 different classifiers are analysed, and some guidelines are designed considering some meta-features of the datasets. This section ends with a discussion of all the results and observations. Finally, the thesis is concluded in section 5 and future work and some limitations are presented in section 6.

2. LITERATURE REVIEW

There is a huge amount of literature focusing on the problem of imbalanced learning, although this is a problem that continues to attract new research and further findings. In this section, we discuss the current state-of-the-art resampling methods which will be applied in a later section. To begin with, we'll discuss the different types of imbalanced learning that can be found, as well as factors that can further complicate the disproportion between classes.

2.1. DIFFERENT TYPES OF CLASS IMBALANCE

Understanding the different kinds of class imbalances is crucial to developing effective strategies for addressing them. This section explores the various dimensions of class imbalance, including between-class and within-class imbalance, relative and absolute imbalance, and intrinsic versus extrinsic class imbalance.

2.1.1. Between-class and within-class imbalance

While frequently overshadowed in discussions about data distribution, a distinct dichotomy in class imbalance emerges. Imbalances in the distribution of the data can occur either between classes or within a single class. According to Japkowicz, (2001), "addressing both problems simultaneously is beneficial and should be done by more sophisticated techniques as well".

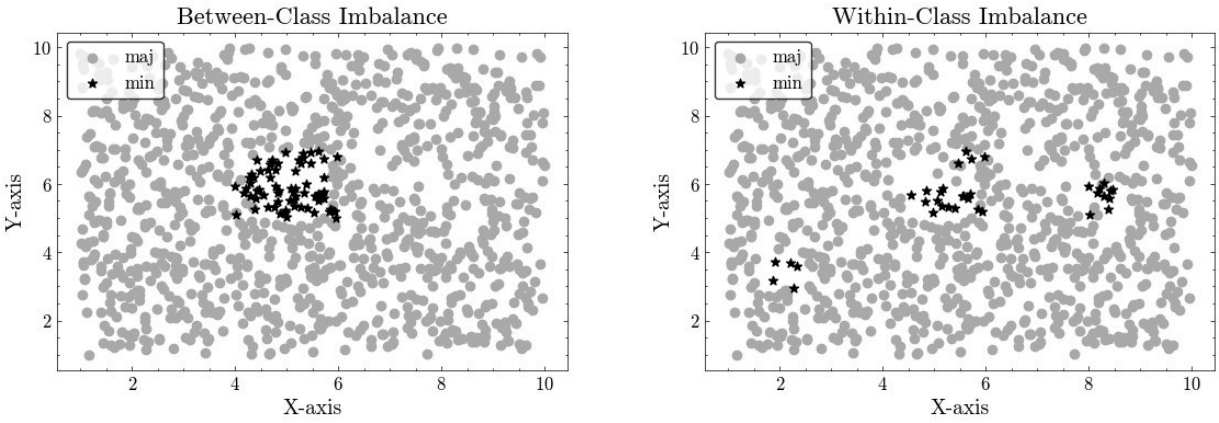
- Between-class imbalance refers to the imbalance that exists between different classes having a significant disparity in the number of instances between each one (Welvaars et al., 2023). Existing studies in the field of imbalance learning primarily focus on proportion imbalance, where the proportion of training samples in each class is not balanced. However, recent research has identified other types of imbalances, including variance, distance, neighborhood, and quality imbalances between classes (Wu, 2023).
- Within-class imbalance is characterized by the uneven distribution of examples within a single class in a dataset, where the class is composed of some different subclusters with different numbers of examples (Japkowicz, 2001). This type of class imbalance commonly exists alongside between-class imbalance, but it can occur independently, and existing work fails to explicitly consider within-class imbalance, leading to suboptimal performances (Z. Liu et al., 2021).

If the two problems mentioned above come together, we are dealing with a problem called small disjuncts which highlights the challenge posed by instances of the minority class that are scattered or isolated in the dataset, making it difficult for the model to effectively learn and generalize (Holte et al., 1989). Rare or exceptional cases correspond to a limited number of training examples in specific regions of the feature space. In the context of acquiring knowledge about a concept, the existence of rare cases in the domain holds significant importance due to their association with small disjuncts, which are known to be more

susceptible to errors compared to large disjuncts. Learning systems commonly generate concept definitions that consist of multiple disjuncts, each representing a conjunction-based definition of a sub concept of the original concept. The coverage of a disjunct is determined by the number of training examples it accurately classifies, and a disjunct is classified as a "small disjunct" if its coverage is low. It must be noted that small disjuncts do not possess inherent error-proneness compared to large disjuncts (Holte et al., 1989). Rather, their heightened proneness to errors is attributed to biases in the classifier, noise in the attributes, absence of attributes, noise in the class, and the size of the training set. These factors collectively impact the rare cases present in the dataset (G. M. Weiss, 1995).

Recent papers by Sun et al. and Al-Maqaleh et al. propose methods to deal with small disjuncts. Sun et al., (2023) introduce a disjuncts-robust oversampling (DROS) method that fills data space with new synthetic samples to the minority class areas, effectively avoiding negative oversampling results. Al-Maqaleh et al., (2012) present a classification algorithm based on Evolutionary Algorithm (EA) that discovers interesting small-disjunct rules.

Though we won't be focussing directly on within-class imbalance in this paper, it's important to understand the concept and the impact it can have in combined with between-class imbalance. Figure 1-a shows an example of between-class imbalance, on the other hand Figure 1-b shows an example of within-class imbalance. We show two examples of artificially generated datasets where the negative samples are underrepresented concerning the positive samples. We must point out that, in both figures, the positive instances or majority class are represented with grey circles whereas negative instances or minority class are represented as dark stars.



(a) (b)

Figure 1 – Between-class and within-class imbalance

2.1.2. Relative and absolute imbalance

When it comes to relative imbalance, the definition is the different proportion of examples between classes. However, we may be considering this imbalance incorrectly, or it may have an additional interpretation if we consider the number of examples from the minority class.

It's important to understand the concept of relative imbalance, which is used when, despite a high degree of imbalance, the minority class has enough examples so that "the minority concept is accurately learned with little disturbance from the imbalance" (He & Garcia, 2009). Consider a dataset with 10.000 examples and an imbalance ratio of 1%. We expect this dataset to contain 100 minority class examples. Suppose we get much more data, and the sample size now consists of 150.000 majority class examples, and the distribution does not change. Now, we have 1.500 from the minority class, which is not necessarily rare but rather relative to the majority class. Real-world applications commonly face relative imbalances, which are the subject of several knowledge discovery and data engineering research projects.

Absolute imbalance focuses on the actual number of instances in the minority class, regardless of the majority class size. Taking the previous example, when we have 1% IR but only 10 instances of the minority class, we can say that this is a case of absolute imbalance, in which case, regardless of the number of instances of the majority class, the model will struggle to learn from such a limited number of instances.

2.1.3. Intrinsic vs extrinsic class imbalance

In this section, we explore the multifaceted nature of imbalance data, distinguishing between intrinsic and extrinsic imbalances. Intrinsic imbalances result directly from the inherent characteristics of the data space, presenting a unique set of challenges. This means that the imbalance is a fundamental feature of the data set due to the underlying properties of the data itself. For example, in a binary classification problem, if one class is significantly more prevalent than another in the original dataset, the resulting imbalance is considered intrinsic. This can occur when certain classes naturally occur more frequently in the real-world scenarios that the data represents.

However, it is essential to recognise that imbalances are not limited to intrinsic origins. External factors such as time, storage limitations, or intermittent interruptions in data transmission also contribute to imbalanced datasets, forming what is known as extrinsic imbalances. These extrinsic imbalances are particularly intriguing as they may occur independently of the nature of the data space. For instance, disruptions in data transmission within an interval of continuous balanced data might result in an extrinsically imbalanced dataset, even though the original dataspace was inherently balanced. This distinction emphasizes the need to address both intrinsic and extrinsic imbalances to comprehensively understand and mitigate challenges associated with imbalanced datasets (He & Garcia, 2009).

2.2. DATASET CHARACTERISTICS THAT IMPACT IMBALANCE

Continuing our study of imbalanced datasets, we now focus on looking more closely at the specific characteristics of datasets that affect imbalance. In section 2.1, we discussed the various types of class imbalance. Building on this foundation, section 2.2 examines the intricate interplay between dataset characteristics and the challenges posed by imbalance.

2.2.1. Small disjuncts

In section 2.1.1 we explored between-class and within-class imbalance and now we're going to take a closer look at one of the most recurrent types of within-class imbalance, which are small disjuncts.

Small disjuncts represent challenging and often overlooked subsets within a class. These subsets, characterized by a limited number of instances, pose a significant hurdle in machine learning tasks, as most of the standard algorithms may struggle to accurately detect and learn from them. To provide a visual aid and enhance understanding, Figure 1-b from the other section illustrates the concept of small disjuncts. The term "disjuncts" refers to disjoint or distinct regions within a class, and when these regions are small, they become particularly challenging to identify (Jo & Japkowicz, 2004).

Weiss extensively explores the issue of small disjuncts in imbalanced datasets and provides various techniques for addressing this issue (G. Weiss, 2010), (Maimon & Rokach, 2010). The strategies include obtaining additional training data through informed sampling, employing a more appropriate inductive bias that distinguishes between large and small disjuncts (Holte et al., 1989) using specific metrics that positively weight minority classes in small disjuncts, avoiding pruning as this tends to eliminate most small disjuncts by a generalization of the obtained rules, and leveraging boosting algorithms like AdaBoost to improve classification performance on difficult-to-predict instances. The author also emphasizes the efficacy of the CBO method, a resampling strategy that simultaneously addresses between-class and within-class imbalances. This method utilizes the k-means algorithm to detect clusters in positive and negative classes, followed by random replication of examples for each cluster (excluding the largest negative cluster) to achieve a balanced distribution. The CBO method is particularly designed to emphasize the significance of small disjuncts in the data (Jo & Japkowicz, 2004).

2.2.2. Class overlapping

Another major issue in imbalance classification has to do with class overlapping, or sometimes referred to as class separability or class complexity. This problem occurs when the distributions of different classes show similarities, resulting in instances where the boundaries between the classes become less distinct. In this scenario, the examples of one class can overlap with those of another, posing challenges for machine learning algorithms in accurately distinguishing between classes (Ali et al., 2013).

Denil and Trappenberg (2010) bring evidence that when the training set is small, high levels of imbalance significantly impact classifiers' performance, although this effect is reduced with larger training sets. The same happens with small disjuncts, where adding more data can mitigate the problem, but in the case of overlapping, adding more data results in a performance drop independently of the training set size. This decline aligns with expectations, as overlapping classes create ambiguous regions in the data space.

In a recent study, researchers conducted empirical analyses on real-world datasets, obtaining interesting insights. The authors evaluated the performance of several real-world datasets and organized the results based on various data complexity metrics, including imbalance ratio (IR). Even though it was not possible to draw notable conclusions based on IR alone, the study successfully identified some patterns of behaviour concerning the other metrics (Luengo et al., 2011).

So far, the most widely used set of metrics for measuring dataset complexity, including overlap between classes, was proposed in 2002 and has been widely used. These metrics are divided into 3 types: overlaps in feature values from different classes; separability of classes and measures of geometry, topology, and density of manifolds (Ho & Basu, 2002). Some of the most interesting measures for class overlapping are summarised in Table 1.

Table 1 – Data complexity measures names and acronyms by Ho and Basu

Measure types	Id.	Description
Overlaps in feature values from different classes	F1	Maximum Fisher's discriminant ratio
	F2	Error rate of linear classifier by linear programming
	F3	Maximum (individual) feature efficiency
Separability of classes	L1	Minimized sum of error distance by linear programming
	L2	Error rate of linear classifier by linear programming
	N1	Fraction of points on class boundary
	N2	Ratio of average intra/inter class NN distance
	N3	Error rate of 1NN classifier

Finally, one of the latest studies on the topic provides a comprehensive discussion of the synergistic relationship between class imbalance and overlap in classification problems. It introduces a new taxonomy of complexity measures for class overlap and explores the challenges faced in data intrinsic characteristics, difficulty factors, and irregularities contributing to these issues. The paper delves into the development of new techniques for addressing class overlap and imbalance, such as feature selection algorithms and feature extraction methods. It also touches on the importance of algorithm design and hyperparameter tuning in effectively addressing these challenges (Santos et al., 2023).

2.2.3. Borderline examples

There are 3 ways of classifying the location of points in space: safe, noisy and borderline examples. Safe examples are situated within relatively homogeneous regions concerning their

assigned class label. In contrast, noisy examples refer to instances from one class found in secure regions associated with the other class. Last, borderline examples are in the vicinity of the decision boundary between classes (Kubat, 2000). Therefore, they are critical because often encapsulate the ambiguity inherent in imbalanced datasets.

Figure 2 represents two examples named "Clover" and "Paw" as shown by Napierała et al. (2010), respectively. In the first, it represented a non-linear setting, where the minority class forms a flower, which makes it difficult to determine the boundaries examples. The latter represents a minority class decomposed into three elliptic subregions, where one is separated and the other two are located close to each other.

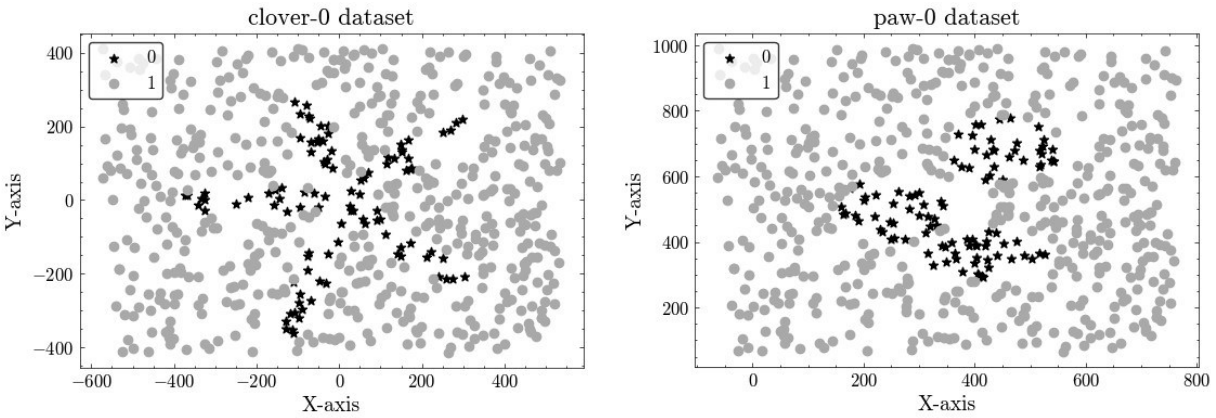


Figure 2 – Examples of data with difficult borderline examples (Alcalá-Fdez et al., 2011)

2.2.4. Noisy data

When certain data points deviate from their typical pattern, we may be dealing with noisy data or outliers, which can also come from errors, but they introduce extra difficulty and complexity for machine learning models. When we are dealing with imbalanced datasets, it takes fewer noisy points to affect the minority class, as opposed to the majority, because it becomes more difficult for the model to understand the concept inherent in it.

Addressing noisy data and outliers in imbalanced datasets is critical for building robust machine learning models. Techniques such as noise filtering, outlier detection, and careful preprocessing become very important to identify and handle instances that could negatively impact the learning process.

A study about the classification performance of learners on software quality data shows three important conclusions:

- Random undersampling and Wilson's edited nearest neighbour (ENN) (Wilson, 1972) performed the best when dealing with noise and imbalanced data. SMOTE and borderline SMOTE performed well but didn't show the same results as those two undersampling techniques.

- When imbalance increases, it is responsible for the bad performance of classifiers, independently of noise. But also, classifiers are more sensitive to noise than imbalance.
- Bayesian classifiers and SVMs proved to be the most robust classifiers (Seiffert et al., 2007).

Although outliers can be included in the noisy points because they cause interference in the data, they are different because they are further away from their class, as can be seen in Figure 3.

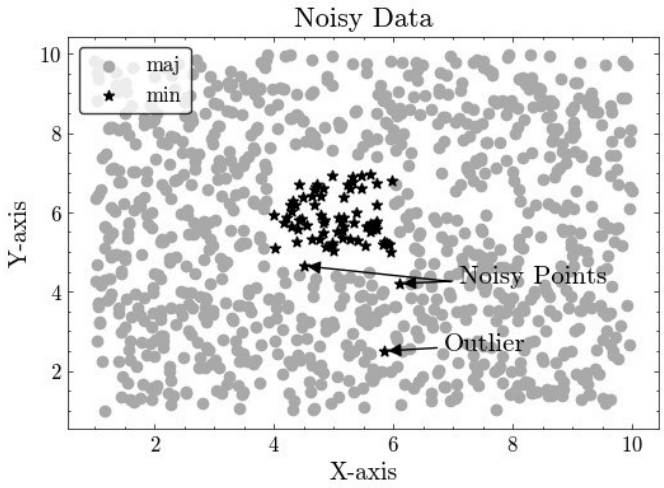


Figure 3 – Representation of noisy points and outliers using synthetic data

2.2.5. Dataset shift

Dataset shift is a common challenge in machine learning, especially when there is limited data available. This challenge occurs when the data distribution used for training the model differs from the distribution of the data used to test the model. To address this challenge, researchers have proposed techniques such as domain adaptation or transfer learning, which leverage data from auxiliary source populations that share some conditional distributions or are linked in other ways with the target domain (Qiu et al., 2023). It is also proposed that, as dataset shift is a crucial avenue for future research in the domain of imbalanced classification, researchers should focus on the development of techniques to detect and quantify dataset shift, with a particular impact on the minority class (Moreno-Torres et al., 2012).

2.2.6. Lack of density

Lack of density is the last impacting characteristic in the problems with imbalanced data that we're going to explore, and it concerns the difficulty of identifying patterns in the examples of the minority class, which are scattered throughout the feature space in such a way that they can be identified as noisy data. As explored in previous sections, this difficulty in defining patterns in the minority class is also common in situations where small disjuncts and very high levels of imbalance occur. However, there is still this problem of a lack of density in the data, further increasing the risk of misclassification and less ability to label the examples of the underrepresented class which could also increase the susceptibility to overfitting on the

majority class (López et al., 2013). In a study by G. Weiss & Provost (2003), the impact of class distribution and training-set size on classifier performance, employing C4.5 as the base learning algorithm, was thoroughly investigated. The research involved varying the available training data and the degree of imbalance across multiple datasets, with a focus on observing differences in the AUC metric under different conditions. The findings revealed two crucial insights. Firstly, a straightforward observation indicated that an increase in the number of training data positively correlated with improved performance, irrespective of the class distribution. Secondly, and more importantly, the study emphasized the dynamic nature of the optimal imbalance ratio (IR) for achieving peak performance, as it occasionally varied with changes in training-set size. This variability suggests the existence of an optimal or near-optimal class distribution for a given learning task. The study further proposed the potential utility of a progressive sampling algorithm to identify the class distribution that maximizes classifier performance under varying training-set sizes.

2.3. IMBALANCE ASSESSMENT METRICS

The choice of metrics to use becomes a critical step in the evaluation of any machine learning model, especially when we are dealing with an imbalanced dataset. Traditional metrics won't provide the necessary information in such scenarios and therefore, as discussed in a previous section, accuracy is one of those that is excluded from the range of metrics to be used, but still presented. As Sun et al. (2009) point out, "evaluation measures play a crucial role in both assessing the classification performance and guiding the classifier modelling".

To structure this section, we will adopt a taxonomy proposed by Ferri et al. (2009), categorizing the evaluation metrics into three distinct groups:

- Threshold metrics
- Ranking metrics
- Probability metrics

This approach ensures an organized and structured presentation of the different metrics and is in harmony with established frameworks also used in more recent articles.

2.3.1. Threshold metrics

Threshold metrics provide a single numerical value to assess the performance of a binary classification model based on specific criteria. They are often computed at a particular threshold or operating point, typically 0.5.

The most common metric for classification problems is accuracy and its complement called classification error or error rate.

Let {P, N} be the true positive and negative class label and {PP, PN} be the predicted positive and negative class labels. Then, a representation of the confusion matrix can be formulated, as illustrated in Table 2. The right column represents negative instances, and the left one

represents the positive instances. Therefore, the proportion of both is representative of the class distribution, and any metric that uses values from both columns in its formula will be sensitive to imbalance, as we will see.

Table 2 – Confusion matrix for performance evaluation

	P **(Positives)**	**N** **(Negatives)**
PP **(Predicted Positives)**	TP (True Positives)	FP (False Positives)
PN **(Predicted Negatives)**	FN (False Negatives)	TN (True Negatives)

Following this, accuracy and error rate are defined as:

$$Accuracy = \frac{TP + TN}{P + N}; \qquad Error\ Rate = 1 - Accuracy$$

These metrics are simple and easy to understand when describing a classifier's performance, however, as stated before, they are not adequate to assess an imbalance problem, as they show a bias towards the majority class. It is possible to have a high accuracy value and still not correctly predict any, or almost any, of the examples in the minority class, because accuracy indicates how many correct classifications were made overall (He & Garcia, 2009). Other metrics are also used in the research community based on confusion matrix values.

Recall, also referred to as true positive rate or sensitivity, indicates the prediction accuracy among minority class examples. It is interpreted as how many minority instances were correctly classified. When assessing a classification model of a problem where it is crucial to obtain the correct classification of the positive class, this is a very useful metric. In the context of cancer classification, a high recall value ensures that the classifier correctly identifies the majority of instances with cancer. This means that, among all the individuals who actually have cancer, the classifier successfully recognizes them, even if it occasionally misidentifies some people without cancer as being ill. In the medical context, prioritizing high recall is crucial as it minimizes the risk of false negatives—instances where the classifier fails to detect actual cases of cancer, which can have more serious consequences than false positives.

$$Recall = \frac{TP}{P}$$

Specificity answers the same question but for the majority class, which is not relevant to this situation.

$$Specificity = \frac{TN}{N}$$

Precision is a measure of exactness i.e., of the predicted positives, how many are actually classified correctly. Although, precision cannot assert how many positive instances were labelled incorrectly.

$$Precision = \frac{TP}{PP}$$

The F-Measure metric combines precision and recall as a measure of the effectiveness of classification. It is a harmonic mean of both completeness and exactness of positive predictions.

$$F - Measure = \frac{(1+\beta^2) \cdot Recall \cdot Precision}{\beta^2 \cdot Recall + Precision} = \frac{(1+\beta^2) \times \left(\frac{TP}{P} \times \frac{TP}{PP}\right)}{\beta^2 \times \frac{TP}{P} + \frac{TP}{PP}},$$

where β is a coefficient to adjust the relative importance of recall versus precision. Usually, this metric is referred to as F-1 Score, where β is set to 1, attributing the same importance to both measures.

The Geometric mean score, also referred to as G-Mean, provides a balanced measure of a classifier's performance across both the majority and minority classes being the geometric mean of recall and specificity. Still, G-Mean and F-Measure are ineffective for more generic classification issues, such as comparing the performance of different classifiers over a range of sampling distributions.

$$G - Mean = \sqrt{recall \times specificity} = \sqrt{\frac{TP}{P} \times \frac{TN}{N}}$$

To get around the G-Mean drawback of not differentiating the contribution of each class to overall accuracy, a new metric called optimized precision (OP) was suggested by (Ranawana & Palade, 2006). It achieves its optimal value when the true negative rate and true positive rate are closely aligned, especially in situations where overall accuracy is a key consideration (Kraiem et al., 2021).

$$OptimizedPrecision = Accuracy - \frac{|Specificity - Recall|}{Specificity + Recall}$$

The metrics mentioned above are the most popular threshold metrics. However, there are many more, due to the ease with which existing metrics can be adapted to specific problems. For example, adjusted geometric mean, balanced accuracy, kappa, macro-average accuracy, and more. Japkowicz (2013) stated that "an important disadvantage of all the threshold metrics … is that they assume full knowledge of the conditions under which the classifier will

be deployed. In particular, they assume that the class imbalance present in the training set is the one that will be encountered throughout the operating life of the classifier".

2.3.2. Ranking metrics

Ranking metrics are crucial tools in assessing the performance of classifiers when dealing with imbalanced datasets because, unlike traditional classification metrics that focus on binary outcomes, ranking metrics evaluate the model's ability to correctly order instances based on their predicted probabilities or scores. These metrics are particularly relevant in situations where the distribution of class labels is imbalanced, and accurately ranking instances becomes more critical than absolute predictions.

The most used ranking metric for assessing classifier performance is the ROC Curve. ROC is an acronym that means Receiver Operating Characteristics and serves as a graphical representation that shows the diagnostic ability of a binary classifier as the discrimination threshold is varied. Positioned on the ROC curve, the X-axis represents the False Positive Rate (FPR), denoting the proportion of incorrectly classified negative examples relative to the total count of negative class observations. Simultaneously, the Y-axis reflects the True Positive Rate. As the classifier's classification threshold is adjusted, different points on the ROC curve emerge, indicating diverse trade-offs. A lower threshold yields more True Positives but also increases False Positives, while a higher threshold reduces both False Positives and True Positives. A classifier that predicts the majority class under all thresholds will be represented by a diagonal line, while a perfect model will be pointed in the plot's top left.

Figure 4 illustrates the ROC curves of different classifiers labelled with the letters A, B, C, and the point P which represents a perfect classifier. Generally speaking, one classifier is better than the other if its corresponding point in ROC space is closer to point P than the other. In this case we can say that the classifier on curve A performed better than the one on curve B because it is closer to point P. The curve labelled C, on the other hand, is a random classifier. Therefore, if any classifier appears in the lower right triangle of ROC space, it performs worse than random guessing. As previously stated, the AUC metric is the area of the ROC space below the curve, until the bottom right corner. It can also happen, for instance, that a certain classifier with a high AUC performs worse in a specific region in ROC space than a low AUC classifier (Fawcett, 2006).

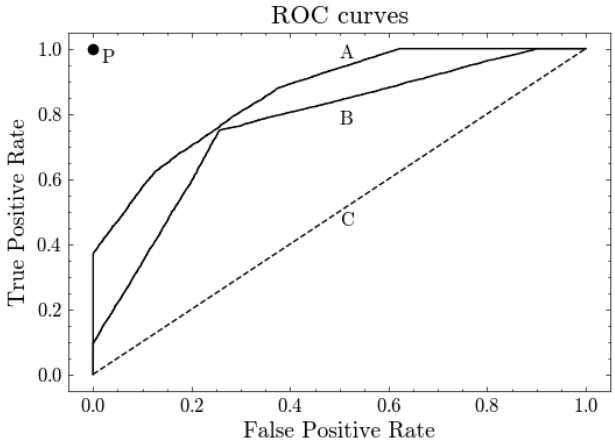


Figure 4 – ROC curves representation

Precision-Recall curve has been used by many of the current researchers in the community due to its ability to assess the model's performance in the presence of highly skewed data (Davis et al., 2005; Singla & Domingos, 2005). This PR curve is defined by plotting precision against recall, and it exhibits a strong correspondence with ROC curves: A curve dominates in ROC space if and only if it dominates in PR space (He & Garcia, 2009). However, optimizing the area under the ROC curve (AUC-ROC) doesn't guarantee optimization of the area under the PR curve (AUC-PR).

Like the ROC curve, the PR curve is a helpful diagnostic tool that can be easily interpreted as it is illustrated in Figure 5. The best models are those near the top right of the graph, where point P represents the perfect model with precision and recall values equal to one. It's easy to see that model A performs better than model B and the letter C is the PR curve of the baseline model, which can also be referred to as the random model.

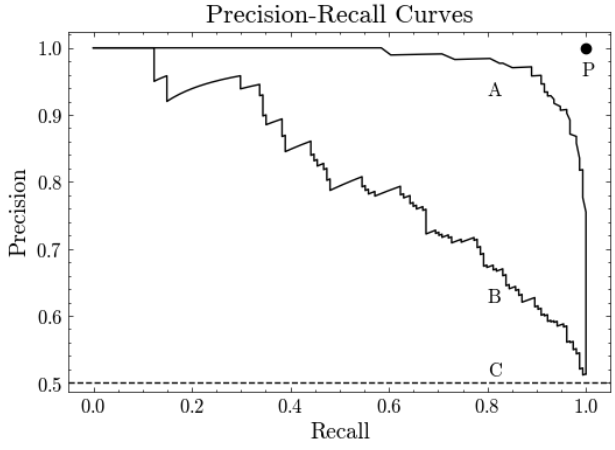


Figure 5 – Precision-Recall curves representation

2.3.3. Probabilistic metrics

In order to evaluate how effectively classification models function, probabilistic metrics are essential since they provide important information about how well-calibrated the probability estimates they provide may be. As opposed to the metrics discussed in the previous two sections, which center on the final predicted labels, they are intended to measure the uncertainty in a classifier's predictions. Certain classifiers, like logistic regression or even certain ensemble techniques like gradient boosting, are dependent on an assumed probability distribution, such as maximum likelihood estimation, which implies that these probabilities are calibrated. However, in order to be compared using probabilistic metrics, models like Naïve Bayes or Support Vector Machines (SVM) require their probabilities to be calibrated.

The most common metric for evaluating predicted probabilities is log loss, also referred to as cross-entropy. It can be calculated as follows:

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}(y_i\ log(p_i) + (1-y_i)\ log(1-p_i))$$

where N is the total number of samples, y_i is the true label of sample i, and p_i is the predicted probability of sample i belonging to class 1.

To address the challenge posed by imbalanced datasets, a tailored metric has been proposed called Cost-Sensitive cross-entropy error function, which can also be called balanced log loss (Aurelio et al., 2019). This metric is the result of assigning a weight to the log loss function mentioned above. Weights W_i can be inversely proportional to class frequencies or can be set manually based on domain knowledge. Because of its adaptability, the model may prioritize correctly classifying the minority class based on varying degrees of class imbalance. The formula for it is as follows:

$$ImbalancedLogLoss = -\frac{1}{N}\sum_{i=1}^{N}W_i(y_i\ log(p_i) + (1-y_i)\ log(1-p_i))$$

Log loss functions summarize the average difference between two probability distributions. The closer to zero, the better the results.

Brier Score is another probabilistic metric used to evaluate the predicted probabilities. As opposed to log loss, it is focused on the positive class and is calculated as the mean squared error between expected probabilities for the positive class and the predicted probabilities. A perfect classifier has a Brier score of zero. It can be calculated as follows:

$$BrierScore = \frac{1}{N}\times\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

2.4. RESAMPLING STRATEGIES

There is an endless array of resampling techniques, offering a spectrum of options to address imbalanced class distributions in training datasets. As the no-free-lunch theorem states, no best technique can be used in any and every case where resampling is needed. All the techniques have their strengths and weaknesses and all of them have been tested on numerous datasets, often obtaining very good results depending on various factors that characterize the datasets. Of all the existing techniques in the scientific community, and those that continue to emerge after new research, there is a set that is most often the widely used and discussed in various articles and it is with some of these that we will experiment further. For now, we're going to look at several of these best-known techniques, which will be presented in 3 different sections:

- Oversampling techniques
- Undersampling techniques
- Hybrid techniques

This exploration not only serves as a guide through the details of these techniques but also sets the stage for our subsequent experimentation, shedding light on their practical implications and effectiveness in dealing with unbalanced datasets through resampling.

2.5. OVERSAMPLING TECHNIQUES

Oversampling stands out as one of the techniques where new methods are emerging, often different versions of existing ones, or even new methods. The main idea behind oversampling is to intentionally increase the representation of the minority class in the training dataset. This increase is done by replicating existing instances or by introducing synthetic instances generated according to the different methods. By introducing new instances of the minority class, some balance is established between the two classes, so that the classifier is more capable of understanding the representation of both.

2.5.1. Random Oversampling

The simplest oversampling method involves randomly duplicating examples from the minority class in the training dataset until the desired class distribution is reached. It's fairly simple to use and is therefore the most widely used among practitioners. However, because it replicates existing instances, it does not add new information to the data. It might even be adding noise by replicating noisy instances of the minority class (Kraiem et al., 2021). This increase in data that is the same as existing data increases the risk of overfitting since the model will memorize duplicate instances, leading to reduced generalization on unseen data.

To put it briefly, random oversampling works best when there is a minor class imbalance, and the dataset size permits duplication without significantly increasing computing overhead.

Evaluating measures like F1-Score, G-Mean, and AUC-ROC is essential for determining overfitting and determining how effective this resampling was.

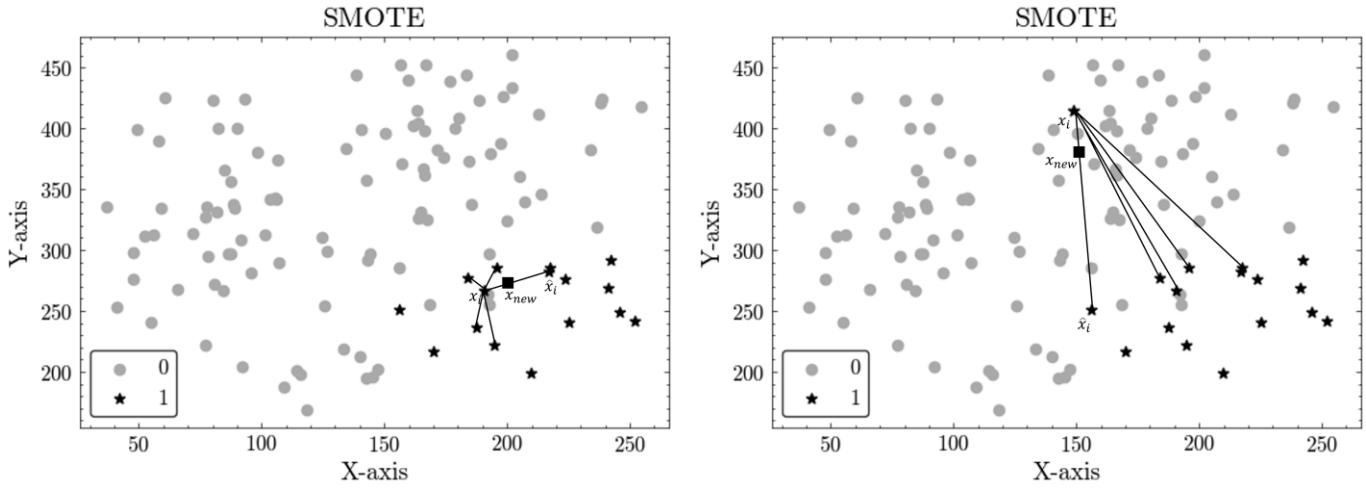
2.5.2. SMOTE

The Synthetic Minority Over-Sampling Technique (SMOTE) is the most famous oversampling method and is subject to constant new developments based on the same principle but with different nuances. In this approach "the minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors" (Chawla et al., 2002). The SMOTE algorithm starts by randomly selecting a minority class instance x_i, then it defines the set of k-nearest neighbors (S_{knn}) which typically is made up of five. Next, it randomly selects another minority class sample $\hat{x}_i$ from S_{knn} and generates the new instance x_{new} by using linear interpolation between the two instances. This can be represented as:

$$x_{new} = x_i + (\hat{x}_i - x_i) \times \delta$$

where $\delta \in [0,1]$ is a random number. Therefore, the resulting instance x_{new} is a point along the line joining $\hat{x}_i$ and x_i.

Figure 6-a shows an example of the SMOTE procedure. The stars and circles represent examples of the minority and majority class, respectively and the lines connect the chosen instance x_i with five nearest neighbors. The generated instance x_{new} is represented by a square which was created along the line between x_i and $\hat{x}_i$.



(a) (b)

Figure 6 – Representations of SMOTE algorithm

While SMOTE provides valuable rebalancing, its impact can vary depending on factors such as the choice of the nearest neighbors (k), the distribution of the dataset, and the characteristics of the learning algorithm employed. One of the main limitations of SMOTE is the potential

generation of synthetic instances that might be deemed as noisy data points. This process, using geometric relationships among minority class samples, can inadvertently introduce synthetic examples in regions of the feature space that may not accurately represent the underlying data distribution. The main reasons why SMOTE might lead to create noisy data points are the selection of the initial instance; the selection of the k-nearest neighbors; using instances from two different minority class clusters; the use of heuristics that assume that the input space has a simple manifold structure. By the latter, we mean that by observing SMOTE's behaviour in 2 dimensions, we are greatly simplifying the feature space, which normally has a large number of variables. This complexity in terms of variables means that the minority class data is not grouped in the way that we observe in 2 dimensions, but is sometimes grouped in small clusters, which leads SMOTE to generate noisy points between distant points.

Figure 6-b shows an example where SMOTE generated a noisy instance x_{new} due to the selection of the initial instance, which is already a noisy instance *per se*.

To deal with these SMOTE limitations, many modifications of the algorithm have been proposed by the scientific community. First, we will dive into Borderline-SMOTE.

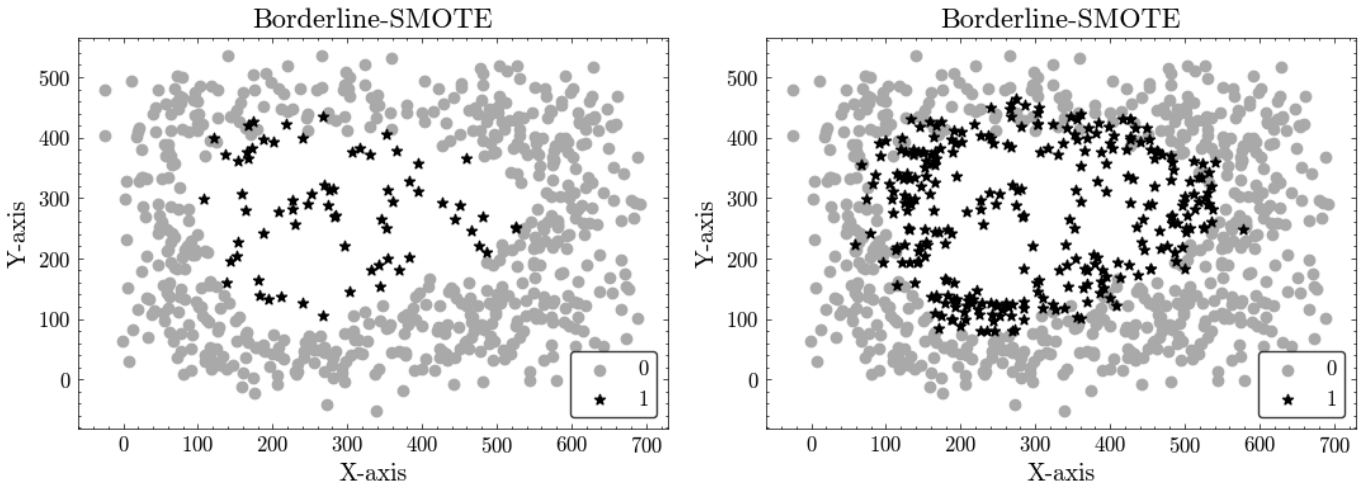
2.5.3. Borderline-SMOTE

Borderline-SMOTE emerges as a sophisticated strategy within oversampling methods, specifically tailored for scenarios where minority class instances reside near the decision boundary. Unlike traditional SMOTE, it focuses on regions where instances are close to the border between classes, attempting to amplify the classifier's discriminatory capabilities in these regions of the feature space (Han et al., 2005). Researchers proposed two borderline-SMOTE algorithms.

Borderline-SMOTE1 starts by identifying the instances of the minority class that are on the border using the k-nearest neighbors. If all the k-nearest neighbors of the chosen point are from the majority class, this point is considered noise and is not targeted for any operations. If half or more of the k-nearest neighbors are from the majority class, and there are some from the minority class, this point is classified as "DANGER" and will be targeted by the algorithm. Finally, if less than half of the k-nearest neighbors are from the majority class, this point is classified as safe and will not take part in the next steps. All the examples considered to be dangerous are targeted by the traditional SMOTE process, thus giving rise to more instances near the borderline and increasing the discriminative capacity of any classifier.

Borderline-SMOTE2 acts in the same way as borderline-SMOTE1, but in addition to generating points through its minority class neighbors, it also does so through the majority class neighbors. The majority class points selected for the artificial generation of data points belong to the k-nearest neighbors of the minority class point and the traditional SMOTE process is also applied, but δ, which was a random value between 0 and 1, is now between 0 and 0.5 so that the point generated is closer to the minority class.

Figure 7-b shows how the data is distributed after borderline-SMOTE was applied to the original data in Figure 7-a.



(a) (b)

Figure 7 – Borderline-SMOTE: before and after applied

2.5.4. ADASYN

Adaptive Synthetic Sampling is another famous approach inside the oversampling techniques. "The essential idea of ADASYN is to use a weighted distribution for different minority class examples according to their level of difficulty in learning" (He et al., 2008).

ADASYN adjusts the sampling density based on the local distribution of instances. It begins by computing the k-nearest neighbors for each minority class instance, then calculates the imbalance ratio for each instance, based on their neighbors. Instances in regions with higher imbalance ratios receive higher weights, indicating that they require more synthetic samples to alleviate the imbalance. Following its foundation on SMOTE, ADASYN generates new instances by leveraging the interpolation technique inherent in SMOTE, while also considering the previously calculated weights. This approach ensures that the creation of synthetic samples is not only influenced by the local characteristics of the minority class instances but is also proportionate to the severity of class imbalance in those specific regions. One drawback is that it does not identify noisy instances, and thus becomes susceptible to outlier values in the dataset. Another constraint is related to the need to tune parameters such as the neighborhood size and the required balance ratio, which makes it sensitive, and getting the optimal values may require experimentation.

Another alternative named KernelADASYN emerged and is based on a kernel density estimation method to estimate the density distribution of minority class (Tang & He, 2015). This approach enables the algorithm to capture complex decision boundaries, offering a more nuanced representation of the minority class.

2.5.5. Geometric SMOTE

Geometric SMOTE (G-SMOTE) is an advanced version of the standard SMOTE algorithm, introducing enhancements to the data generation process. Unlike many variations that modify conditions for generating instances, G-SMOTE focuses on refining the generation mechanism itself. It operates within a geometric region in the input space around each selected minority class instance. G-SMOTE utilizes a hypersphere, allowing deformation into a hyper-spheroid. The objectives of G-SMOTE include defining a safe area around each minority instance to avoid noisy artificial instances and increase sample variety by expanding the minority class area (Douzas & Bacao, 2019).

This process can be summarized in 3 main stages:

- Data Shuffling and Selection:
    - Shuffle the dataset and repeat until N artificial points are generated.
    - Select a minority class instance (x_{center}) as the center, following the shuffled order.
    - Some minority class samples may be selected more than once if N exceeds S_{min} (the set of minority class samples).
- Neighbor Selection Strategy:
    - Three strategies: minority-based, majority-based, and combined which originate $x_{surface}$.
    - Minority strategy (similar to SMOTE): Select k nearest neighbors from the minority class.
    - Majority strategy: Choose the nearest neighbor from the majority class.
    - Combined strategy: Apply both, restricting expansion by the nearest majority neighbor.
- Data Generation Process:
    - Generate a random point (e_{sphere}) on the surface of a unit hyper-sphere at the origin.
    - Transform e_{sphere} into a uniformly distributed point (x_{gen}) within the hyper-sphere.
    - Apply a linear mapping based on α_{trunc}, truncating or expanding the hyper-sphere.
    - Use a deformation transformation controlled by α_{def} to create a hyper-spheroid.
    - Translate the generated point by the x_{center} vector and rescale it by the radius R.

Figure 8 shows a visual representation of how G-SMOTE works using a majority class instance as $x_{surface}$ and the generated point represented as x_{gen}.

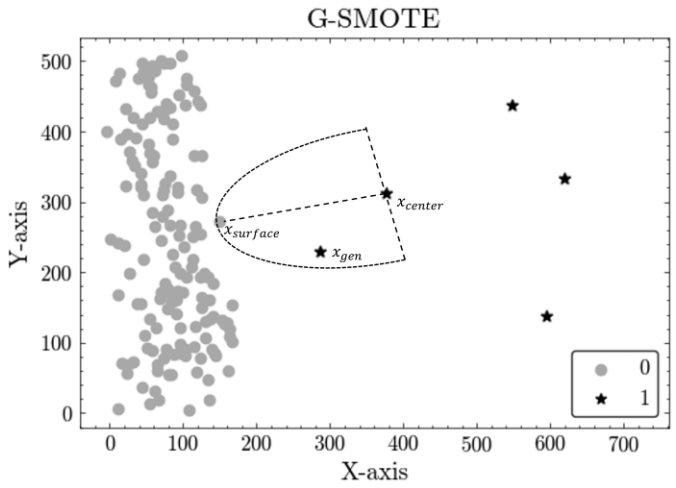


Figure 8 – Representation of G-SMOTE using majority strategy

G-SMOTE has been applied to both classification and regression tasks, showing improved performance compared to other methods in both domains. In classification tasks, G-SMOTE has been found to increase the accuracy of the classifier model compared to SMOTE and class imbalanced data (Bação, 2020). In regression tasks, G-SMOTE outperformed other methods in terms of prediction accuracy (Camacho et al., 2022). The algorithm has also been successfully applied to address the imbalance learning problem in remote sensing, significantly improving the quality of classification results (Douzas et al., 2019).

2.5.6. Other techniques

In addition to those methods explored in the last few sections, there is a wide range of other methods that are less frequently used and that are still emerging in the scientific community. The methods described above are good at dealing with between-class imbalance problems, however, as seen in section 2.1 there is the problem of within-class imbalance. This imbalance requires another type of method, based on clusters.

K-Means SMOTE is a simple and effective way of dealing with such cases of within-class imbalance. It is based on 3 main steps: clustering, filtering, and oversampling. First, it clusters the input space into k groups of clusters using k-means. The k value is one of the hyperparameters of the algorithm since it is crucial to obtaining good results. Then, it filters the clusters obtained, retaining those with a high proportion of minority examples. By default, it retains clusters with at least 50% of minority class examples. Finally, SMOTE is applied in each selected cluster and the number of synthetic samples to generate is greater in sparse clusters (Last et al., 2017).

DBSMOTE uses the DB-SCAN algorithm to discover arbitrarily shaped clusters and then generates synthetic instances along the shortest path from each minority class instance to a pseudo-centroid of the cluster. It is like K-Means SMOTE but uses a density-based clustering algorithm (Bunkhumpornpat et al., 2012).

Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) clusters the minority instances using a semi-unsupervised hierarchical clustering approach and adaptively determines the size to oversample each sub-cluster using its classification complexity and cross-validation. It aims to identify hard-to-learn instances by considering minority instances from each sub-cluster that are closer to the borderline. A-SUWO also avoids generating synthetic minority instances that overlap with the majority class by considering the majority class in the clustering and oversampling stages (Nekooeimehr & Lai-Yuen, 2016).

SOMO (Self-Organizing Map Oversampling) employs a two-dimensional representation of the input space created by a Self-Organizing Map (SOM). The SOM clusters data, preserving neighborhood structures. In the second stage, it generates synthetic data within clusters for the minority class. In the third stage, inter-cluster synthetic data is generated for adjacent clusters in the two-dimensional grid. SOMO adjusts the distribution of intra-cluster and inter-cluster generated data based on cluster density, addressing both between-class and within-class imbalance. Unlike methods assuming a simple manifold structure, SOMO considers the SOM's topology-preserving property (Douzas & Bacao, 2017).

2.6. UNDERSAMPLING TECHNIQUES

Undersampling techniques, like oversampling, are essential for correcting the imbalance between classes. While oversampling increases the instances of the minority class, undersampling focuses on reducing the number of instances of the majority class. This intentional reduction aims to create a more balanced training data set, allowing the classifier to better capture the details of both classes. Undersampling methods involve removing instances of the majority class, either randomly or through strategic selection.

2.6.1. Random Undersampling

To achieve the desired class distribution, the simplest undersampling technique entails randomly deleting cases from the majority class in the training dataset. Random undersampling is a computationally effective technique, but it has a cost: if data is discarded, important information may be lost. This is especially true if the majority class instances are essential for identifying the general pattern in the data (He & Garcia, 2009). Nonetheless, when applied judiciously, random undersampling proves to be a pragmatic solution in mitigating the challenges posed by imbalanced datasets during machine learning model training. This method, along with random oversampling, is suitable for slightly imbalanced datasets, but it is not recommended for high levels of imbalance (Kraiem et al., 2021).

2.6.2. Easy Ensemble and Balance Cascade

Easy ensemble and balance cascade are two examples of informed undersampling that have shown good results trying to explore the majority class examples ignored by random undersampling (X. Y. Liu et al., 2009).

Easy Ensemble employs a bagging approach to create multiple subsets with balanced class distribution. Given an imbalanced dataset, it randomly selects a subset of the majority class instances that matches the size of the minority class. The ensemble is formed by training multiple base classifiers (often decision trees or other weak learners) on these balanced subsets. The final prediction is typically made through a majority voting mechanism.

On the other hand, the ensemble method known as Balanced Cascade functions in a cascade form. It eliminates majority class examples that are incorrectly classified and trains base classifiers iteratively. Multiple stages of the process ensure that instances of the majority class that have been incorrectly classified are eventually deleted. When a predetermined stopping condition is satisfied, the algorithm comes to an end. This method concentrates on gradually resolving the imbalance problem by eliminating examples that provide difficulties for the classifier.

2.6.3. NearMiss Undersampling

Near Miss refers to a collection of undersampling methods that select examples to keep from the majority class. The main idea behind Near Miss is to reduce the number of majority class instances by retaining those that are close to the minority class instances. The metric distance used is Euclidean distance or similar. This method aims to create a subset of the majority class instances that are more representative of the minority class, making the training dataset more balanced (Zhang & Mani, 2003). There are 3 different versions of this technique:

- NearMiss-1 selects examples from the majority class that have the smallest average distance to the three closest examples from the minority class.
- NearMiss-2 selects examples from the majority class that have the smallest average distance to the three furthest examples from the minority class.
- NearMiss-3 selects a given number of majority class examples for each example in the minority class that is closest.

2.6.4. Condensed Nearest Neighbor Rule

Condensed Nearest Neighbor (CNN) is another technique that selects examples of majority class to keep. The primary objective of CNN is to condense the dataset by removing redundant majority class instances without compromising the representativeness of the dataset (Kraiem et al., 2021). The algorithm operates iteratively. Initially, it starts with an empty set of selected instances. Subsequently, it iteratively adds instances from the majority class to the selected set if they are misclassified by a k-nearest neighbor classifier using the current set of selected instances. This procedure ends when no more data points are added to the selected set. CNN is based on a strategy of data reduction method used to improve the efficiency in the implementation of the k-nearest neighbor decision rule for classification problems (Hart, 1968).

2.6.5. Tomek Links

Tomek Links is one of the modifications of CNN that can be used to remove instances from the majority class (Tomek, 1976). The main objective of the method is not returning balanced data, but it can be used as a guided undersampling method. The idea is to identify pairs of instances (say, *x* and *y*) that fulfil two main properties:

- *x* and *y* are from different classes.
- Let d(*x,y)* be the distance between points. There is no other point *z* where d(*x,z*)<d(*x,y*) or d(*y,z*)<d(*x,y*).

These will identify potential sources of misclassification, typically occurring near the decision boundary. Removing these links helps refine the dataset by eliminating ambiguous instances.

2.6.6. Other techniques

One-Sided Selection (OSS) combines Tomek Links and the CNN rule. Tomek Links are used as an undersampling method and remove noisy and borderline majority class instances. Then, CNN is used to remove redundant examples from the majority class that are far from the decision boundary (Kubat, 2000).

Radial-Based Undersampling (RBU) introduces the concept of mutual class potential, a real-valued function assigned to each data point based on Gaussian radial basis functions and class polarity (Koziarski, 2020). This function finds regions where class affiliation is unclear and retains occurrences in areas where the level of certainty is lower. RBU seeks to reduce classifier bias in favour of the majority class and move the decision boundary in the minority class's favour by lowering the absolute value of potential. Promising properties of the approach are shown, especially when dealing with disjoint distributions and outliers. Its computational cost is still an issue, though, particularly for big datasets.

2.7. HYBRID TECHNIQUES

Hybrid techniques combine oversampling and undersampling methods into one, in order to obtain the best of both. They are increasingly being explored and used and in this study we will explore two of them.

2.7.1. SMOTE-Tomek Links

SMOTE-Tomek combines the strengths of both SMOTE (Synthetic Minority Over-sampling Technique) and Tomek Links, providing a hybrid approach that addresses the limitations of each individual method (Z. Wang et al., 2019). SMOTE generates data points for the minority class to balance it. Tomek Links identifies and removes borderline instances from the majority class, refining the decision boundary.

As stated in section 2.5.2, SMOTE generates points along a line that connects the points of the minority class to its nearest neighbors. On the other hand, Tomek Links identifies the pairs of

instances (one from each class) that are both nearest neighbors and removes the one from the majority class, thus reducing noise on the borderline and class overlapping.

SMOTE-Tomek is particularly useful in scenarios where there is significant class overlap, and noisy instances are prevalent. By using this method, it enhances the minority class representation while simultaneously reducing the majority class noise. However, removing points from the majority class can lead to the loss of relevant information (as with any undersampling techniques), especially in more complex datasets where it is more difficult to distinguish decision boundaries.

2.7.2. SMOTE-ENN

SMOTE-ENN, like the previous method, also uses SMOTE as an oversampling technique and in this case uses Edited Nearest Neighbors (ENN) as an undersampling technique which tend to remove more examples than Tomek Links (Batista et al., 2004).

The idea behind ENN is simple: eliminate points that are surrounded (k-nearest neighbors) by a greater number of points from the opposite class. Usually, the 3 nearest neighbors are used for comparison and if the majority of these 3 neighbors are from a different class, that point is removed. Despite being an undersampling technique, it is possible to remove points from both classes, as this is one of the hyperparameters.

3. METHODOLOGY

The objective of this research is to compare different resampling techniques, whether using oversampling, undersampling, or a hybrid approach, on a significantly large set of datasets. Therefore, to obtain the broadest possible picture, we will use a variety of 35 datasets, a set of metrics, and classifiers that we find most appropriate. This section presents a description of this setup.

3.1. EXPERIMENTAL DATA

To ensure meaningful insights we tested the different resampling techniques on a total of 35 binary datasets. These datasets consist of commonly used datasets from the OpenML repository, more specifically the OpenML-CC18 Curated Classification benchmark (Bischl et al., 2019), and from the KEEL repository. Some of them are from real-world problems and others are artificially generated data. To ensure that a wide variety of data is covered, these datasets concern different topics such as finance, physics, and healthcare. Also, there are very different values of imbalance ratio, varying from 2 to 600. Table 3 presents some details of the datasets.

Table 3 – Description of the datasets

ID	Dataset Name	#instances	#attributes	IR	Source
1	abalone19	4174	9	129.438	KEEL
2	adult	48842	15	3.179	OpenML
3	bank-marketing	45211	17	7.548	OpenML
4	blood-transfusion-service-center	748	5	3.202	OpenML
5	car-good	1728	7	24.043	KEEL
6	churn	5000	21	6.072	OpenML
7	climate-model-simulation-crashes	540	21	10.739	OpenML
8	connect-4	67557	43	6.349	OpenML
9	CreditCardSubset	14240	31	618.13	OpenML
10	credit-g	1000	21	2.333	OpenML
11	ecoli1	336	8	3.364	KEEL
12	ecoli2	336	8	5.462	KEEL
13	glass1	214	10	1.816	KEEL
14	glass6	214	10	6.379	KEEL
15	haberman	306	4	2.778	KEEL
16	ilpd	583	11	2.491	OpenML
17	Internet-Advertisements	3279	1559	6.144	OpenML
18	jm1	10885	22	4.168	OpenML
19	jungle_chess_2pcs_raw_endgame_complete	44819	7	9.339	OpenML
20	kc1	2109	22	5.469	OpenML
21	kc2	522	22	3.879	OpenML
22	mammography	11183	7	42.012	OpenML
23	nomao	34465	119	2.501	OpenML
24	ozone-level-8hr	2534	73	14.838	OpenML

25	page-blocks0	5472	11	8.789	KEEL
26	pc1	1109	22	13.403	OpenML
27	pc3	1563	38	8.768	OpenML
28	pc4	1458	38	7.191	OpenML
29	phoneme	5404	6	2.407	OpenML
30	poker-8-9_vs_6	1485	11	58.4	KEEL
31	segment0	2308	20	6.015	KEEL
32	sick	3772	30	15.329	OpenML
33	wilt	4839	6	17.540	OpenML
34	yeast4	1484	8	28.098	KEEL
35	Yeast-0-5-6-7-9_vs_4	528	8	9.35	KEEL

3.2. RESAMPLING METHODS

Not all the resampling methods explored in section 2 will be applied in the experiment. Therefore, we have selected a set of techniques that we think cover different paradigms in the resampling field and that are also the most explored in the scientific community. We will use: ROS, SMOTE, G-SMOTE and ADASYN as oversamplers and RUS, Near-Miss and Tomek Links as undersamplers. To test the use of hybrid techniques we will use SMOTE-Tomek Links and SMOTE-ENN.

3.3. CLASSIFIERS

Different classifier algorithms are chosen to evaluate the performance of the selected resampling techniques and to make sure the results are not dependent on the classifier. The four selected classifiers are Logistic Regression, K-Nearest Neighbors, Random Forest, and XGBoost. Logistic Regression is a linear model used for binary or multiclass classification, estimating the probability of an instance belonging to a particular class, and can be used as a baseline model (Dreiseitl & Ohno-Machado, 2002). KNN is a non-parametric algorithm that classifies an instance based on the majority class of its k nearest neighbors (Guo Gongdeand Wang, 2003). Random Forest is an ensemble method that builds multiple decision trees during training and combines their prediction through voting to improve accuracy and robustness (Biau & Scornet, 2016). XGBoost is an ensemble technique that sequentially builds decision trees, with each tree aiming to correct errors made by the previous ones, resulting in a powerful predictive model (Chen & Guestrin, 2016).

3.4. ASSESSMENT METRICS

Considering the entire spectrum of metrics discussed in the previous part, the F1-Score, G-Mean, and AUC-ROC will be the main metrics that we rely on in this study. These metrics were chosen because they are useful in reflecting the complexity of scenarios with imbalanced classification and because they are relevant to the particular goals of this study. We will be able to completely assess the findings by concentrating on this set of measures, ensuring that the goals of the research are clearly and precisely satisfied.

3.5. CHARACTERIZING DATASET THROUGH THE USE OF META-FEATURES

To describe each dataset according to its characteristics, we extracted some meta-features based on the Extended Complexity Library (ECoL) (Lorena et al., 2020) and additional meta-learning features (Rivolli et al., 2019). The ECoL package implemented in R extract a set of measures to characterize the complexity of classification problems based on aspects related to the presence of informative feature, the sparsity and dimensionality of the datasets. We used the following meta-features: number of instances (#instances), number of attributes (#attr), imbalance ratio (IR), instance to attribute ratio (inst2attr) which represents the sparsity of the data, number of dummy attributes (#dummy), number of categorical attributes (#cat), number of numerical attributes (#num), categorical to numerical ratio (cat2num), number of attributes normal distributed (#norm), average correlation with target (avg_corr_class), maximum Fisher's discriminant ratio (F1) and the directional-vector Fisher's discriminant ratio (F1v).

The last two are very important to assess the overlapping between classes since maximum Fisher's discriminant ratio measures the degree of class separation by calculating the ratio of the variance between classes to the variance within classes. A higher value indicates better separation between classes, suggesting that the features are more discriminative for classification. And the directional-vector Fisher's discriminant ratio extends the concept of Fisher's discriminant ratio by considering the directional information of features. It evaluates the discriminative power of features in a specific direction, capturing their effectiveness in separating classes along that direction.

3.6. EXPERIMENTAL FRAMEWORK

To ensure consistency across all datasets, the data preprocessing steps were standardized. This included creating dummy variables for categorical features and applying a Min-Max scaler to normalize the data. This preprocessing step was crucial to ensure that all features contributed equally to the model's performance.

To evaluate the performance of each combination of resampler and classifier, n-fold cross-validation with $n = 5$ was applied to aim to maximize the F1-Score metric. For each dataset, synthetic data was generated within every cross-validation iteration using various resampling methods to ensure complete balance in the training set. This involved oversampling techniques, where additional synthetic instances were created to augment the minority class, and undersampling techniques, where instances from the majority class were selectively removed to achieve balance.

During the training phase, a variety of hyper-parameters were used both for the classifiers and resampling methods. Hyperparameter tuning was performed using grid search methods to identify the optimal parameter settings for each model. Although the search space for the classifiers could have been more extensive, the parameter selection was constrained to

balance computational efficiency with model performance, allowing for a more comprehensive search of the resamplers' hyperparameters.

The implementation of classifiers and resampling techniques was carried out using several libraries: Scikit-learn (Pedregosa et al., 2011) for general machine learning tasks, Imbalanced-learn (Lemaître et al., 2017) for handling imbalanced using the resamplers, XGBoost (Chen & Guestrin, 2016) for gradient boosting, and Geometric SMOTE (Douzas & Bacao, 2019) for generating synthetic samples using G-SMOTE.

After completing the cross-validation process, each resampler-classifier combination's performance metrics (F1-Score, G-Mean, and AUC-ROC) were aggregated and analysed. The results were compared across different resampling methods to identify which techniques perform best for various imbalance ratios and dataset characteristics. A ranking score was assigned for each resampling method with the best and worst receiving scores from 1 to 10, respectively.

The Friedman test was conducted to determine the significance of the observed differences in performance, followed by Holm's correction to adjust the p-values. Additionally, we performed paired t-tests to further examine the differences between the performance of the resampling methods across various classifiers. Statistical tests, such as the Friedman test and Holm's correction, help in assessing the significance of the performance differences and ensuring robust conclusions about the effectiveness of each resampling method in improving classifier performance on imbalanced datasets.

In addition, we developed a decision tree model based on the meta-features described at the section above. Each dataset has its own meta-features, and the target class consists of the best resampling method for that dataset. Through this approach it is possible to draw valuable guidelines about which method to use.

4. RESULTS AND DISCUSSION

In this section, we present and discuss the results of our experiments. We begin with a comparative presentation of the mean cross-validation scores across datasets for each combination of classifiers, metrics and resamplers. Following this, we present the ranking score assigned to each resampler for each combination of classifier and evaluation metric. Next, we discuss the results of the statistical significance tests conducted to determine the performance differences among the resampling methods. Finally, we include results from a decision tree model developed to identify the most suitable resampling method based on dataset meta-features.

4.1. COMPARATIVE PRESENTATION

In Table 4 are presented the mean cross validation scores across datasets for each combination of classifiers, metrics and resamplers.

Table 4 – Mean cross validation scores of resamplers across the datasets

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	0.774	0.786	0.726	**0.791**	0.779	0.788	0.776	0.780	0.671	0.659
LR	F-SCORE	0.552	0.556	0.529	0.534	0.525	0.567	0.552	**0.575**	0.466	0.437
LR	G-MEAN	0.746	0.763	0.685	**0.781**	0.776	0.779	0.745	0.754	0.526	0.495
KNN	AUC	0.787	0.765	0.709	0.756	**0.795**	0.788	0.794	0.788	0.725	0.711
KNN	F-SCORE	0.574	**0.592**	0.502	0.573	0.507	0.585	0.565	0.579	0.591	0.570
KNN	G-MEAN	0.766	0.740	0.668	0.703	0.778	**0.779**	0.773	0.767	0.657	0.629
RF	AUC	0.753	0.771	0.733	0.699	0.731	**0.784**	0.769	0.756	0.727	0.732
RF	F-SCORE	0.543	0.585	0.506	0.501	0.464	**0.637**	0.551	0.552	0.567	0.597
RF	G-MEAN	0.680	0.706	0.641	0.565	0.607	**0.754**	0.663	0.667	0.635	0.639
XGB	AUC	0.743	0.786	0.736	0.732	0.731	**0.797**	0.765	0.780	0.741	0.755
XGB	F-SCORE	0.528	0.636	0.495	0.518	0.456	**0.659**	0.550	0.594	0.580	0.618
XGB	G-MEAN	0.669	0.734	0.660	0.619	0.614	**0.768**	0.676	0.720	0.631	0.686

4.2. RANKING SCORE

To better identify the most effective resampling method, we assigned a ranking score to each resampler based on its performance. These scores were then aggregated using the mean to provide an overall ranking. Table 5 presents the results with the best one in each row underlined.

Table 5 – Mean ranking of resamplers across the datasets

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	4.771	4.486	6.8	**3.6**	4.6	4.029	4.229	3.971	8.057	8.457
LR	F-SCORE	5.086	4.629	5.6	4.971	5.971	3.429	4.914	**3.343**	6.914	7.514
LR	G-MEAN	4.8	4.286	6.657	**3.286**	4.2	4.057	4.286	3.971	8.171	8.629
KNN	AUC	4.514	5.257	8.114	5.057	3.914	3.971	4.086	**3.543**	6.514	7.829
KNN	F-SCORE	5.486	**4.029**	7.171	4.6	6.229	4.457	5.771	4.657	4.571	5.571
KNN	G-MEAN	4.343	5.429	7.714	5	3.743	3.971	4.057	**3.6**	6.686	8.171
RF	AUC	5.114	4.114	5.257	6.229	5.029	**3.971**	4.257	4.8	6.2	6.171
RF	F-SCORE	5.743	4.371	6.2	6.029	6.914	**3.457**	4.857	4.629	4.743	4.657
RF	G-MEAN	5.029	4.086	5	6.171	4.771	**4.029**	4.286	4.914	6.314	6.429
XGB	AUC	6.029	4.314	5.714	5.086	5.657	**3.429**	4.771	4.114	5.943	5.543
XGB	F-SCORE	6.143	4.114	6.571	4.771	7.886	**3.171**	6.257	4.086	3.857	4
XGB	G-MEAN	5.914	4.457	5.371	5.143	5.6	**3.486**	4.514	4.114	6.086	6

4.3. STATISTICAL ANALYSIS

Table 6 presents the results of the Friedman test, with p-values adjusted using the Holm's correction method.

Table 6 – Friedman test and Holm's correction results

Classifier	Metric	p-value	Adjusted p-value (Holm)	Significance
LR	F-SCORE	$6.3 \cdot 10^{-10}$	$4.4 \cdot 10^{-9}$	True
LR	G-MEAN	$3.4 \cdot 10^{-23}$	$4.0 \cdot 10^{-22}$	True
LR	AUC	$8.2 \cdot 10^{-20}$	$8.2 \cdot 10^{-19}$	True
KNN	F-SCORE	$2.9 \cdot 10^{-4}$	$9.0 \cdot 10^{-4}$	True
KNN	G-MEAN	$7.9 \cdot 10^{-18}$	$7.9 \cdot 10^{-17}$	True
KNN	AUC	$2.4 \cdot 10^{-16}$	$2.2 \cdot 10^{-15}$	True
RF	F-SCORE	$1.7 \cdot 10^{-5}$	$1.1 \cdot 10^{-4}$	True
RF	G-MEAN	$1.4 \cdot 10^{-4}$	$7.2 \cdot 10^{-4}$	True
RF	AUC	$6.4 \cdot 10^{-4}$	$9.0 \cdot 10^{-4}$	True
XGB	F-SCORE	$2.2 \cdot 10^{-14}$	$1.8 \cdot 10^{-13}$	True
XGB	G-MEAN	$2.2 \cdot 10^{-4}$	$9.0 \cdot 10^{-4}$	True
XGB	AUC	$3.0 \cdot 10^{-4}$	$9.0 \cdot 10^{-4}$	True

At a significance level of $p = 0.05$, the null hypothesis is rejected, indicating that the classifiers perform differently in the mean rankings between the resampling methods and the evaluation metrics.

In addition to the Friedman test results presented in Table 6, paired t-tests were conducted to further examine the differences between the performance of the resampling methods across various classifiers. Holm's correction method was applied to the p-values obtained from these t-tests to control for the family-wise error rate.

The paired t-tests compared the performance metrics (F-Score, G-Mean, AUC) between each classifier's results with different resampling methods and a control method. Significant p-values, highlighted in bold, indicate that the resampling method in question performs differently from the control, potentially offering a performance improvement or degradation. Below are presented the adjusted p-values (Holm) for the two resamplers with best results, SMOTE in Table 7 and G-SMOTE in Table 8. The other results are presented in Appendix A.

Table 7 – Adjusted p-values using Holm's method for control group SMOTE

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	**$6.8\cdot10^{-4}$**	1.21	**$1.8\cdot10^{-8}$**	0.78	0.67	**$1.5\cdot10^{-3}$**	**0.03**	**$1.5\cdot10^{-14}$**	**$6.2\cdot10^{-14}$**
LR	F-SCORE	**$9.6\cdot10^{-3}$**	0.09	**0.01**	**$4.1\cdot10^{-6}$**	**$2.4\cdot10^{-6}$**	**$9.5\cdot10^{-3}$**	0.07	**$5.1\cdot10^{-5}$**	**$6.5\cdot10^{-6}$**
LR	G-MEAN	0.13	0.78	**$5.5\cdot10^{-5}$**	0.89	1.13	0.11	0.27	**$2.5\cdot10^{-9}$**	**$3.1\cdot10^{-10}$**
KNN	AUC	1.41	**$3.4\cdot10^{-9}$**	**$4.7\cdot10^{-14}$**	**$1.8\cdot10^{-6}$**	0.67	0.59	0.95	**$2.4\cdot10^{-14}$**	**$3.4\cdot10^{-14}$**
KNN	F-SCORE	0.15	0.28	**$1.2\cdot10^{-13}$**	0.35	**$6.5\cdot10^{-18}$**	**$4.3\cdot10^{-4}$**	0.38	0.51	0.31
KNN	G-MEAN	1.36	**$5.9\cdot10^{-3}$**	**$3.1\cdot10^{-7}$**	**0.02**	0.65	0.72	0.98	**$1.1\cdot10^{-6}$**	**$4.8\cdot10^{-8}$**
RF	AUC	**$3.6\cdot10^{-6}$**	**0.04**	**$1.7\cdot10^{-8}$**	**$1.5\cdot10^{-8}$**	**$1.5\cdot10^{-6}$**	0.06	**$5.4\cdot10^{-5}$**	**$8.8\cdot10^{-14}$**	**$3.2\cdot10^{-9}$**
RF	F-SCORE	**$3.9\cdot10^{-7}$**	**$8.1\cdot10^{-5}$**	**$1.2\cdot10^{-9}$**	**$2.3\cdot10^{-6}$**	**$1.9\cdot10^{-11}$**	**$5\cdot10^{-5}$**	**$2.8\cdot10^{-5}$**	**$2.1\cdot10^{-5}$**	**$5.1\cdot10^{-3}$**
RF	G-MEAN	**0.01**	**0.01**	**$6\cdot10^{-4}$**	**$1.1\cdot10^{-6}$**	**$4\cdot10^{-4}$**	**0.02**	**0.01**	**$4.9\cdot10^{-6}$**	**$1.3\cdot10^{-6}$**
XGB	AUC	**$6.8\cdot10^{-12}$**	**0.01**	**$1.5\cdot10^{-9}$**	**$3.5\cdot10^{-8}$**	**$2.4\cdot10^{-10}$**	**$1.7\cdot10^{-4}$**	**$2.6\cdot10^{-3}$**	**$7.8\cdot10^{-10}$**	**$3.4\cdot10^{-9}$**
XGB	F-SCORE	**$4.4\cdot10^{-10}$**	**$4\cdot10^{-3}$**	**$5.6\cdot10^{-11}$**	**$3\cdot10^{-7}$**	**$1.6\cdot10^{-15}$**	**$9.3\cdot10^{-8}$**	**$8.1\cdot10^{-4}$**	**$5\cdot10^{-4}$**	**$1.4\cdot10^{-3}$**
XGB	G-MEAN	**$6.8\cdot10^{-5}$**	**0.03**	**$1.4\cdot10^{-4}$**	**$8.8\cdot10^{-5}$**	**$6.3\cdot10^{-5}$**	**0.01**	**0.03**	**$1.1\cdot10^{-4}$**	**$8.8\cdot10^{-5}$**

Table 8 – Adjusted p-values using Holm's method for control group G-SMOTE

Classifier	Metric	ADASYN	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	**$1.3\cdot10^{-4}$**	**$7.2\cdot10^{-10}$**	1.03	0.61	0.6	**0.02**	0.35	**$5.1\cdot10^{-12}$**	**$1.8\cdot10^{-11}$**
LR	F-SCORE	0.65	0.1	**$3.5\cdot10^{-3}$**	**$8.1\cdot10^{-4}$**	0.13	0.48	**0.03**	**$3\cdot10^{-3}$**	**$3\cdot10^{-5}$**
LR	G-MEAN	**$1.2\cdot10^{-4}$**	**$1.4\cdot10^{-7}$**	**$1.8\cdot10^{-3}$**	0.44	0.26	**$4.3\cdot10^{-4}$**	0.07	**$1.6\cdot10^{-9}$**	**$3.7\cdot10^{-10}$**
KNN	AUC	**$9.2\cdot10^{-8}$**	**$9.6\cdot10^{-13}$**	**0.01**	**$2.7\cdot10^{-6}$**	**$3.4\cdot10^{-9}$**	**$6.2\cdot10^{-8}$**	**$5\cdot10^{-9}$**	**$2\cdot10^{-11}$**	**$1.7\cdot10^{-11}$**
KNN	F-SCORE	**0.01**	**$6.2\cdot10^{-13}$**	0.06	**$1.3\cdot10^{-15}$**	0.28	**$2\cdot10^{-5}$**	**0.04**	0.88	0.06
KNN	G-MEAN	0.08	**$1.5\cdot10^{-8}$**	0.04	**$1.4\cdot10^{-5}$**	**$4.9\cdot10^{-4}$**	**0.03**	0.06	**$5.5\cdot10^{-8}$**	**$1\cdot10^{-9}$**
RF	AUC	**$2.7\cdot10^{-4}$**	**$1\cdot10^{-8}$**	**$1.4\cdot10^{-9}$**	**$4.4\cdot10^{-8}$**	**0.04**	0.68	**0.02**	**$1.1\cdot10^{-8}$**	**$1.2\cdot10^{-8}$**
RF	F-SCORE	**$2.2\cdot10^{-5}$**	**$7.9\cdot10^{-9}$**	**$2.8\cdot10^{-5}$**	**$3.2\cdot10^{-10}$**	**$2\cdot10^{-4}$**	**0.02**	**0.02**	**0.02**	**0.01**
RF	G-MEAN	0.06	**$2.3\cdot10^{-3}$**	**$6.4\cdot10^{-7}$**	**$9.3\cdot10^{-4}$**	0.05	0.09	0.08	**$4.3\cdot10^{-7}$**	**$2.8\cdot10^{-9}$**
XGB	AUC	**$1.5\cdot10^{-11}$**	**$4.1\cdot10^{-10}$**	**$1.3\cdot10^{-8}$**	**$2.7\cdot10^{-11}$**	**0.02**	**$2\cdot10^{-4}$**	0.13	**$8.1\cdot10^{-12}$**	**$5.4\cdot10^{-6}$**

XGB	F-SCORE	**$1.2 \cdot 10^{-10}$**	**$1.8 \cdot 10^{-10}$**	**$9.1 \cdot 10^{-7}$**	**$6 \cdot 10^{-15}$**	**$8 \cdot 10^{-3}$**	**$1.2 \cdot 10^{-7}$**	**$7 \cdot 10^{-3}$**	**$4.9 \cdot 10^{-3}$**	**0.02**
XGB	G-MEAN	**$2 \cdot 10^{-5}$**	**$1.9 \cdot 10^{-4}$**	**$3.1 \cdot 10^{-5}$**	**$2.3 \cdot 10^{-5}$**	**0.03**	**0.03**	0.36	**$6.2 \cdot 10^{-5}$**	**$2.3 \cdot 10^{-4}$**

SMOTE and G-SMOTE often show no significant differences in performance, indicating that they perform very similarly. Another resampling method, SMOTE-Tomek Links, also ranks similarly to these two, as seen in the previous sections where we presented average and ranking results. Since these resamplers frequently demonstrate the best performance, significant differences observed through this test likely indicate a degradation in performance of the other methods compared to the control group.

4.4. DECISION TREE ANALYSIS OF DATASET META-FEATURES

After assessing which resampler is better for each classifier and metric scenario, the next step was to perform an analysis to obtain a deeper knowledge about the impact of dataset characteristics (meta-features) on resampling strategies.

For this purpose, we conducted a decision tree analysis focusing on the XGBoost classifier and the F1-Score metric. This focused analysis was chosen due to the robustness of XGBoost and the relevance of the F1-Score metric in imbalanced classification scenarios. XGBoost has demonstrated superior performance in various machine learning tasks, making it a suitable choice for this analysis. Additionally, the F1-Score metric provides a balanced measure of classification performance, particularly suitable for imbalanced datasets, as it considers both precision and recall, and it was used in the search for hyperparameters.

Given the limited size of our dataset, consisting of only 35 datasets, traditional methods of training and testing a decision tree using a train-test split may not be optimal. Instead, to ensure robustness and stability in our analysis, we trained the decision tree using all datasets. This approach allows us to maximize the utilization of available data and generate a decision tree model that captures the underlying relationships between dataset meta-features and the choice of resampling strategy more accurately. By analysing the node partition rules we can derive some guidelines for determining which resampling method may be most effective in specific contexts. With a macro-averaged F1-Score of 0.81 for the decision tree model, we can extract some raw guidelines, presented in Appendix A - Table A9, on the best resampling method to use based on the dataset meta-features.

In essence, this proved once again that G-SMOTE and SMOTE are the two methods that allow us to obtain the best results, although under certain conditions there are other methods that surpass them. By analysing the decision tree, we can conclude which conditions are most suitable for the following methods:

- G-SMOTE:
    - Performs best with medium to high class overlap, a high ratio of instances to attributes, and significant class imbalance (greater than 4.4).

    - Also excels when class separability is very high, combined with a very high imbalance ratio.
- SMOTE:
    - Effective in scenarios with moderate to high class separability and a high imbalance ratio (greater than 16.2).
    - Works well for smaller datasets with medium to high class overlap.
- SMOTE-Tomek Links:
    - Yields the best results for datasets with the highest class overlap ratio and a high ratio of instances to attributes.
- Random Oversampling:
    - Effective when there are a high number of instances and few attributes, combined with an average imbalance ratio (smaller than 16.2).
- Near Miss:
    - Recommended for datasets with a low imbalance ratio (less than 4.4), and handles medium to high class overlap effectively
- No resampling:
    - In certain conditions, using the raw data without resampling yields better results. This is typically the case for datasets with medium to low imbalance ratios and a high number of instances and attributes, where the classifier can learn patterns effectively without the need for resampling.

4.5. DISCUSSION

By analyzing the aggregated results from the mean cross-validation scores and mean rankings, we can identify the most consistent and the least effective resamplers. SMOTE stands out as the top performer across the majority of metric-dataset combinations. This trend was reflected in the mean ranking as well, where SMOTE often ranked among the top resamplers, demonstrating its reliability across multiple datasets.

While SMOTE Tomek sometimes showed slightly lower mean cross-validation scores compared to SMOTE, it exhibited more consistent rankings. For example, in the KNN classifier for the G-Mean metric, SMOTE had a higher mean score (0.779) than SMOTE Tomek (0.767). However, SMOTE Tomek achieved a better mean ranking (3.6) compared to SMOTE (3.971). This suggests that SMOTE Tomek, despite having marginally lower scores, performed more reliably across different datasets, minimizing drastic performance fluctuations.

However, a significant observation is the competitive performance of G-SMOTE. It exhibited performance metrics closely aligned with SMOTE, combining high performance with an added layer of consistency and versatility by generating new points within a geometric region around a minority class point instead of along a straight line. Its ability to consistently rank among the top resamplers across different classifiers and metrics highlights its robustness and this is particularly evident in classifiers like RF and XGBoost, where G-SMOTE's rankings were often in the top 3.

Oppositely, no resampling or resamplers like Tomek Links and ADASYN consistently showed poorer performance across most metrics and classifiers. For instance, in Logistic Regression, using the original dataset had some of the lowest mean scores (AUC: 0.659, F-Score: 0.437, G-Mean: 0.495), and this was reflected in their lower rankings. This indicates that datasets with imbalanced classes significantly benefit from resampling techniques, as opposed to using raw, imbalanced data.

We aggregated the data to a lower degree of granularity, focusing on four categories: oversampling, undersampling, hybrid, and no resampling. For each category, we selected the best-performing method. This analysis utilizes the XGBoost classifier, consistent with our decision tree analysis. Additionally, we present the data using radar plots combining the three metrics with three main meta-features, inspired by Kraiem et al. (2021), which prove to be a very effective approach. Our findings indicate that the no resampling strategy often yields worse results compared to the resampling strategies and among the resampling strategies, oversampling frequently emerges as the most effective approach.

For the AUC metric, in Figure 9, the resampling methods show irregular behaviour, although it is possible to conclude that for high levels of class overlap (indicated by lower values) a hybrid approach is always positioned as one of the best and this reinforces the idea that SMOTE-Tomek Links is quite good when datasets suffer from class overlap. When it comes to the instance to attribute ratio, SMOTE is effective for any level of this ratio.

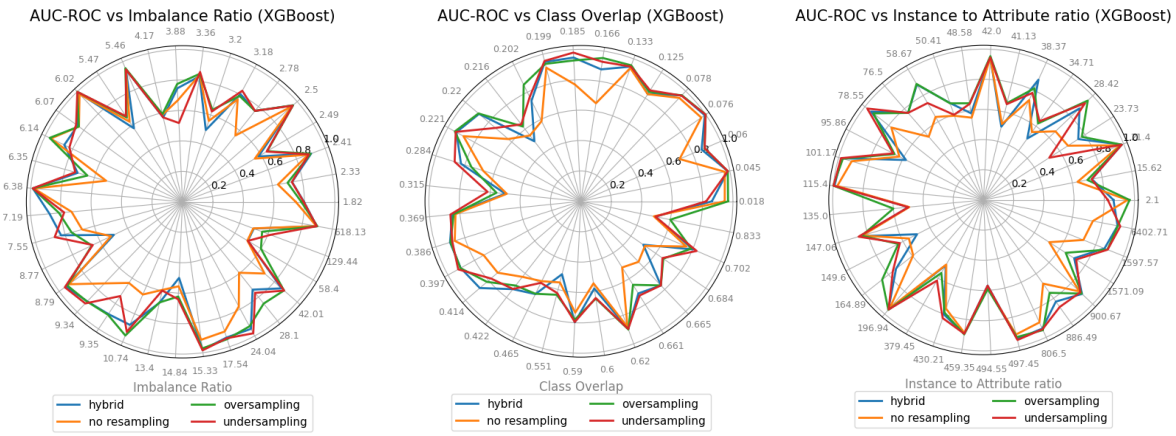


Figure 9 – Impact of dataset meta-features on AUC

In Figure 10, as the imbalance ratio increases, it becomes clearer that oversampling consistently achieves better results. In contrast, when considering the instance-to-attribute ratio, undersampling nearly always aligns with oversampling, although the latter demonstrates greater consistency in achieving superior outcomes.

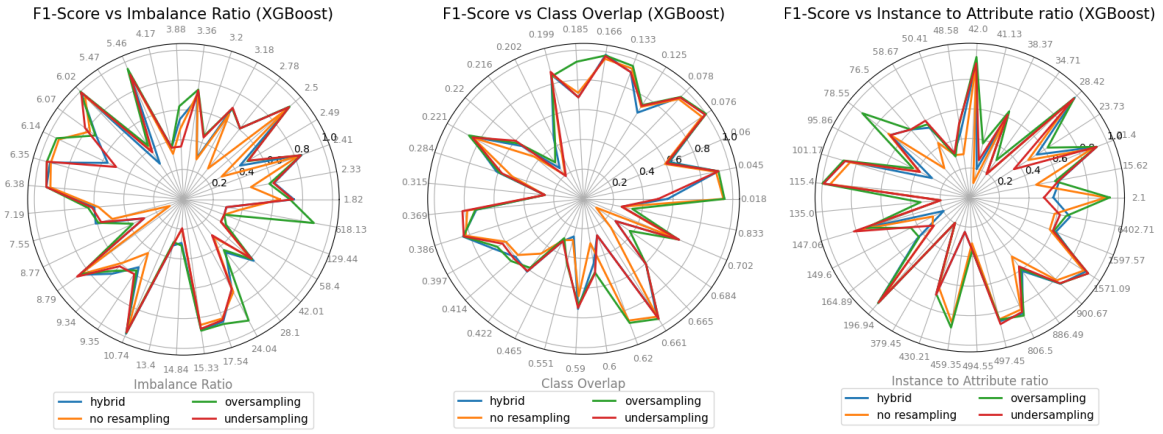


Figure 10 – Impact of dataset meta-features on F1-Score

Finally, the G-Mean analysis in Figure 11 proves that no resampling consistently has inferior results compared to using some resampling method. Notably, the orange line is positioned below the others across all three meta-features plots. Interestingly, undersampling had better results for the two highest imbalance ratios and performed very well when dealing with datasets where the number of instances far exceeds the number of attributes.

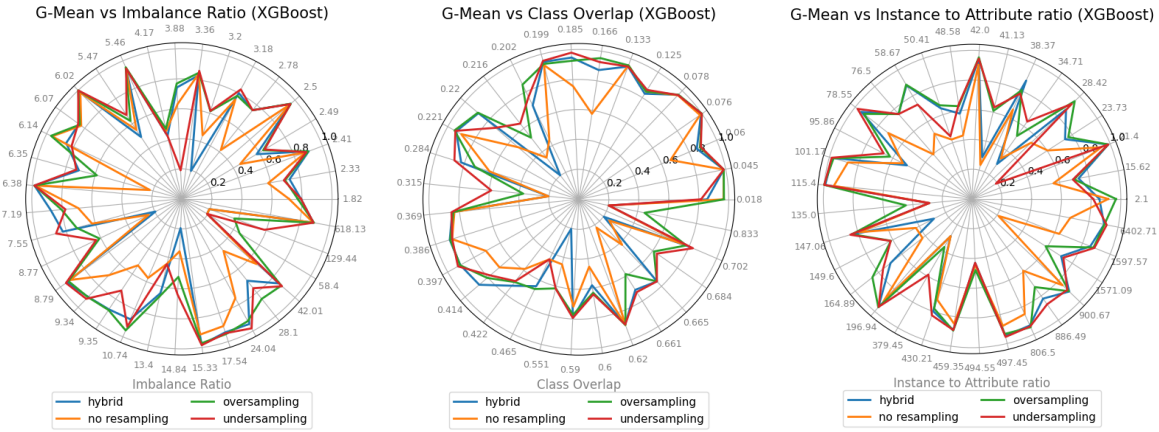


Figure 11 – Impact of dataset meta-features on G-Mean

In addition, and based on the decision tree analysis, we can broadly state that G-SMOTE and SMOTE are effective in scenarios with high imbalance ratios. SMOTE Tomek is a suitable choice for datasets with a high ratio of instances per attribute and high class overlapping. ROS and Near Miss present viable options to consider and test. Additionally, using the original imbalanced dataset may sometimes yield results similar to certain resamplers, also depending on the classifier used and its ability to deal with imbalanced data natively.

5. CONCLUSION

In this thesis, we explored the results of applying data-level approaches to address imbalanced data, specifically focusing on oversampling, undersampling and hybrid methods, in binary classification problems. We presented an overview of existing methods in the scientific community and discussed some of the challenges associated with class imbalance. We also examine the most common metrics for evaluating imbalanced data scenarios and explained their reliability. We then carried out an experimental analysis using 35 datasets from different domains and origins, some of which were artificially generated. For each dataset, we tested the application of 9 different resampling techniques, as well as a control scenario with no resampling technique applied. We used four different classifiers: Logistic Regression, K-Nearest Neighbors, Random Forest and XGBoost.

The analysis showed that the method with best ranking and average score was SMOTE, which performs consistently across all datasets. In addition to SMOTE, other methods such as G-SMOTE and SMOTE Tomek showed very promising and consistent results, closely aligning with those obtained by SMOTE. As expected, not using any resampling method resulted in highly inconsistent outcomes, which depend on the characteristics of each dataset. In some cases, this approach yielded good metrics, while in others, it performed poorly. The simplest techniques, which are random oversampling and random undersampling, produced better results than more complex methods such as ADASYN. However, due to their random nature, these methods can often introduce unnecessary noise, increase overfitting, and degrade the decision boundary between the classes.

Some datasets have unique features, and to represent these features accurately, we decided to extract a set of meta-features that characterize each dataset. These meta-features range from simple ones, such as the number of instances and variables, to more complex ones, such as the directional-vector Fisher's discriminant ratio (F1v), which measures class overlap. Using these meta-features, we trained a decision tree where the output is the type of resampling method to be used. In other words, with the decision tree, we could derive guidelines on which resampling method is most beneficial given certain meta-feature values of the dataset. For instance, SMOTE Tomek, a hybrid method, is particularly effective for datasets with a high degree of class overlap. On the other hand, SMOTE and G-SMOTE are well-suited for datasets with a high imbalance ratio.

To conclude, although the results of our experimental analysis and the guidelines obtained provide a valuable starting point for researchers and practitioners, it remains essential to test different approaches to dealing with each specific problem. It is not possible to definitively conclude that a single method or exclusively using oversampling techniques will always be optimal. It has been proven here that hybrid approaches can offer superior results in certain scenarios.

6. LIMITATIONS AND FUTURE WORK

Although this thesis has provided valuable insights regarding the different resampling techniques and offered guidelines on when to use specific methods, which may be useful for both literature and practice, there are limitations and points for improvement in future work. Starting with the fact that the analysis was based on only 35 datasets. While this may seem adequate, it is not because we were not able to mimic all the characteristics and diversity of real-world datasets. Additionally, we focused exclusively on binary classification problems. Extending this study to multiclass classification and regression problems would provide a more comprehensive understanding, especially since imbalance in regression problems is less explored than in multiclass problems and deserves further study.

Although we used classifiers with different underlying concepts, they do not cover all the possibilities of existing machine learning algorithms and the hyperparameter tuning for these classifiers was also not exhaustive due to computational constraints. Given these limitations, we focused more on the resamplers than on an extensive search for optimal classifier hyperparameters. Future studies should consider a deeper search for hyperparameters, leveraging more computational resources for robust evaluations.

Regarding the analysis using meta-features, in the future and considering a broader study as described here, even more meta-features could be used, and the decision tree model could be trained and tested on datasets not seen during training to evaluate the reliability of our guidelines.

All things considered, this study is very significant for the scientific community, not least because it is unusual to see comparisons of oversampling, undersampling, and hybrid methods, but also because it opens the door to future research.

BIBLIOGRAPHICAL REFERENCES

Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). *KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework* (Vol. 17). http://the-data-mine.com/bin/view/Software

Ali, A., Ralescu, A., Shamsuddin, S. M., & Ralescu, A. L. (2013). Classification with class imbalance problem: A review. *Classification Int. J. Advance Soft Compu. Appl*, *5*(3). https://www.researchgate.net/publication/288228469

Al-Maqaleh, B., Al-Dohbai, M., & Shahbazkia, H. (2012). An Evolutionary Algorithm for Automated Discovery of Small-Disjunct Rules. *International Journal of Computer Applications*, *41*, 33–34. https://doi.org/10.5120/5547-7615

Aurelio, Y. S., de Almeida, G. M., de Castro, C. L., & Braga, A. P. (2019). Learning from Imbalanced Data Sets with Weighted Cross-Entropy Function. *Neural Processing Letters*, *50*(2), 1937–1949. https://doi.org/10.1007/s11063-018-09977-1

Bação, F. J. F. L. (2020). *Small data oversampling: improving small data prediction accuracy using the geometric SMOTE algorithm*. http://hdl.handle.net/10362/99077

Batista, G., Prati, R., & Monard, M.-C. (2004). A Study of the Behavior of Several Methods for Balancing machine Learning Training Data. *SIGKDD Explorations*, *6*, 20–29. https://doi.org/10.1145/1007730.1007735

Biau, G., & Scornet, E. (2016). A random forest guided tour. *TEST*, *25*(2), 197–227. https://doi.org/10.1007/s11749-016-0481-7

Bischl, B., Casalicchio, G., Feurer, M., Hutter, F., Lang, M., Mantovani, R. G., van Rijn, J. N., & Vanschoren, J. (2019). OpenML Benchmarking Suites. *ArXiv:1708.03731v2 [Stat.ML]*.

Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2012). DBSMOTE: Density-Based Synthetic Minority Over-sampling TEchnique. *Applied Intelligence*, *36*(3), 664–684. https://doi.org/10.1007/s10489-011-0287-y

Camacho, L., Douzas, G., & Bacao, F. (2022). Geometric SMOTE for regression. *Expert Systems with Applications*, *193*, 116387. https://doi.org/https://doi.org/10.1016/j.eswa.2021.116387

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res. (JAIR)*, *16*, 321–357. https://doi.org/10.1613/jair.953

Chen, T., & Guestrin, C. (2016, August). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785

Davis, J., Burnside, E., Dutra, I., Ramakrishnan, R., Costa, V., & Shavlik, J. (2005). View Learning for Statistical Relational Learning: With an Application to Mammography. In *IJCAI*.

Denil, M., & Trappenberg, T. (2010). Overlap versus imbalance. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6085 LNAI*, 220–231. https://doi.org/10.1007/978-3-642-13059-5_22

Douzas, G., & Bacao, F. (2017). Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning. *Expert Systems with Applications*, *82*, 40–52. https://doi.org/https://doi.org/10.1016/j.eswa.2017.03.073

Douzas, G., & Bacao, F. (2019). Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE. *Information Sciences*, *501*, 118–135. https://doi.org/https://doi.org/10.1016/j.ins.2019.06.007

Douzas, G., Bacao, F., Fonseca, J., & Khudinyan, M. (2019). Imbalanced Learning in Land Cover Classification: Improving Minority Classes' Prediction Accuracy Using the Geometric SMOTE Algorithm. *Remote Sensing*, *11*(24). https://doi.org/10.3390/rs11243040

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics*, *35*(5), 352–359. https://doi.org/https://doi.org/10.1016/S1532-0464(03)00034-0

Elkan, C. (2001). The Foundations of Cost-Sensitive Learning. *Proceedings of the Seventeenth International Conference on Artificial Intelligence: 4-10 August 2001; Seattle*, *1*.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/https://doi.org/10.1016/j.patrec.2005.10.010

Ferri, C., Hernández-Orallo, J., & Modroiu, R. (2009). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, *30*(1), 27–38. https://doi.org/https://doi.org/10.1016/j.patrec.2008.08.010

Guo Gongde and Wang, H. and B. D. and B. Y. and G. K. (2003). KNN Model-Based Approach in Classification. In Z. and S. D. C. Meersman Robert and Tari (Ed.), *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE* (pp. 986–996). Springer Berlin Heidelberg.

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. In *Expert Systems with Applications* (Vol. 73, pp. 220–239). Elsevier Ltd. https://doi.org/10.1016/j.eswa.2016.12.035

Han, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. *Adv Intell Comput*, *3644*, 878–887. https://doi.org/10.1007/11538059_91

Hart, P. (1968). The condensed nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory*, *14*(3), 515–516. https://doi.org/10.1109/TIT.1968.1054155

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284. https://doi.org/10.1109/TKDE.2008.239

Ho, T. K., & Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(3), 289–300. https://doi.org/10.1109/34.990132

Holte, R. C., Acker, L. E., & Porter, B. W. (1989). Concept Learning and the Problem of Small Disjuncts. *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1*, 813–818.

Japkowicz, N. (2001). Concept-Learning in the Presence of Between-Class and Within-Class Imbalances. In E. Stroulia & S. Matwin (Eds.), *Advances in Artificial Intelligence* (pp. 67–77). Springer Berlin Heidelberg.

Japkowicz, N. (2013). Assessment Metrics for Imbalanced Learning. In *Imbalanced Learning* (pp. 187–206). John Wiley & Sons, Ltd. https://doi.org/https://doi.org/10.1002/9781118646106.ch8

Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, *6*(1), 40–49. https://doi.org/10.1145/1007730.1007737

Koziarski, M. (2020). Radial-Based Undersampling for imbalanced data classification. *Pattern Recognition*, *102*, 107262. https://doi.org/10.1016/j.patcog.2020.107262

Kraiem, M. S., Sánchez-Hernández, F., & Moreno-García, M. N. (2021). Selecting the suitable resampling strategy for imbalanced data classification regarding dataset properties. An approach based on association models. *Applied Sciences (Switzerland)*, *11*(18). https://doi.org/10.3390/app11188546

Kubat, M. (2000). Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. *Fourteenth International Conference on Machine Learning*.

Last, F., Douzas, G., & Bação, F. (2017). *Oversampling for Imbalanced Learning Based on K-Means and SMOTE*.

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research*, *18*(17), 1–5. http://jmlr.org/papers/v18/16-365.html

Ling, C. X., & Sheng, V. S. (2010). Cost-Sensitive Learning. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of Machine Learning* (pp. 231–235). Springer US. https://doi.org/10.1007/978-0-387-30164-8_181

Liu, X. Y., Wu, J., & Zhou, Z. H. (2009). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *39*(2), 539–550. https://doi.org/10.1109/TSMCB.2008.2007853

Liu, Z., Wei, P., Wei, Z., Yu, B., Jiang, J., Cao, W., Bian, J., & Chang, Y. (2021). *Handling Inter-class and Intra-class Imbalance in Class-imbalanced Learning*. http://arxiv.org/abs/2111.12791

López, V., Fernández, A., García, S., Palade, V., & Herrera, F. (2013). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, *250*, 113–141. https://doi.org/10.1016/j.ins.2013.07.007

Lorena, A. C., Garcia, L. P. F., Lehmann, J., Souto, M. C. P., & Ho, T. K. (2020). *How Complex is your classification problem? A survey on measuring classification complexity*.

Luengo, J., Fernández, A., García, S., & Herrera, F. (2011). Addressing data complexity for imbalanced data sets: Analysis of SMOTE-based oversampling and evolutionary undersampling. *Soft Computing*, *15*(10), 1909–1936. https://doi.org/10.1007/s00500-010-0625-8

Maimon, O., & Rokach, L. (2010). *Data Mining and Knowledge Discovery Handbook, 2nd ed*.

Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V, & Herrera, F. (2012). A unifying view on dataset shift in classification. *Pattern Recognition*, *45*(1), 521–530. https://doi.org/https://doi.org/10.1016/j.patcog.2011.06.019

Napierała Krystyna and Stefanowski, J. and W. S. (2010). Learning from Imbalanced Data in Presence of Noisy and Borderline Examples. In M. and R. S. and J. R. and H. Q. Szczuka Marcin and Kryszkiewicz (Ed.), *Rough Sets and Current Trends in Computing* (pp. 158–167). Springer Berlin Heidelberg.

Nekooeimehr, I., & Lai-Yuen, S. K. (2016). Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets. *Expert Systems with Applications*, *46*, 405–416. https://doi.org/https://doi.org/10.1016/j.eswa.2015.10.031

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel V. and Thirion, B., Grisel, O., Blondel, M., Prettenhofer P. and Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Qiu, H., Tchetgen, E. T., & Dobriban, E. (2023). *Efficient and Multiply Robust Risk Estimation under General Forms of Dataset Shift*.

Ranawana, R., & Palade, V. (2006). Optimized Precision - A New Measure for Classifier Performance Evaluation. *2006 IEEE International Conference on Evolutionary Computation*, 2254–2261. https://doi.org/10.1109/CEC.2006.1688586

Rivolli, A., Garcia, L. P. F., Soares, C., Vanschoren, J., & de Carvalho, A. C. P. L. F. (2019). *Characterizing classification datasets: a study of meta-features for meta-learning*.

Santos, M. S., Abreu, P. H., Japkowicz, N., Fernández, A., & Santos, J. (2023). A unifying view of class overlap and imbalance: Key concepts, multi-view panorama, and open avenues for research. In *Information Fusion* (Vol. 89, pp. 228–253). Elsevier B.V. https://doi.org/10.1016/j.inffus.2022.08.017

Seiffert, C., Khoshgoftaar, T., Van Hulse, J., & Folleco, A. (2007). An Empirical Study of the Classification Performance of Learners on Imbalanced and Noisy Software Quality Data. *Information Sciences*, *259*. https://doi.org/10.1016/j.ins.2010.12.016

Singla, P., & Domingos, P. (2005). Discriminative Training of Markov Logic Networks. *Proceedings of the National Conference on Artificial Intelligence*, *2*, 868–873.

Sterner, P., Goretzko, D., & Pargent, F. (n.d.). *COST-SENSITIVE LEARNING 1 Everything has its Price: Foundations of Cost-Sensitive Learning and its Application in Psychology*. https://osf.io/cvks7/.

Sun, Y., Cai, L., Liao, B., Zhu, W., & Xu, J. (2023). A Robust Oversampling Approach for Class Imbalance Problem With Small Disjuncts. *IEEE Transactions on Knowledge and Data Engineering*, *35*(6), 5550–5562. https://doi.org/10.1109/TKDE.2022.3161291

SUN, Y., WONG, A. K. C., & KAMEL, M. S. (2009). CLASSIFICATION OF IMBALANCED DATA: A REVIEW. *International Journal of Pattern Recognition and Artificial Intelligence*, *23*(04), 687–719. https://doi.org/10.1142/S0218001409007326

Tang, B., & He, H. (2015). KernelADASYN: Kernel based adaptive synthetic data generation for imbalanced learning. *2015 IEEE Congress on Evolutionary Computation (CEC)*, 664–671. https://doi.org/10.1109/CEC.2015.7256954

Tischio, R. M., & Weiss, G. M. (n.d.). *Identifying Classification Algorithms Most Suitable for Imbalanced Data*.

Tomek, I. (1976). A Generalization of the k-NN Rule. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-6*(2), 121–126. https://doi.org/10.1109/TSMC.1976.5409182

Van Hulse, J., Khoshgoftaar, T. M., & Napolitano, A. (2007). Experimental perspectives on learning from imbalanced data. *ACM International Conference Proceeding Series*, *227*, 935–942. https://doi.org/10.1145/1273496.1273614

Wang, L., Zhang, Z., Zhang, X., Zhou, X., Wang, P., & Zheng, Y. (2021). A Deep-forest based approach for detecting fraudulent online transaction. *Advances in Computers*, *120*, 1–38. https://doi.org/10.1016/BS.ADCOM.2020.10.001

Wang, Z., Wu, C., Zheng, K., Niu, X., & Wang, X. (2019). SMOTETomek-based Resampling for Personality Recognition (July 2019). *IEEE Access*, *PP*, 1. https://doi.org/10.1109/ACCESS.2019.2940061

Weiss, G. (2010). The Impact of Small Disjuncts on Classifier Learning. In *Annals of Information Systems* (Vol. 8, pp. 193–226). https://doi.org/10.1007/978-1-4419-1280-0_9

Weiss, G. M. (1995). Learning with Rare Cases and Small Disjuncts. In A. Prieditis & S. Russell (Eds.), *Machine Learning Proceedings 1995* (pp. 558–565). Morgan Kaufmann. https://doi.org/https://doi.org/10.1016/B978-1-55860-377-6.50075-X

Weiss, G., & Provost, F. (2003). Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction. *J. Artif. Intell. Res. (JAIR)*, *19*, 315–354. https://doi.org/10.1613/jair.1199

Welvaars, K., Oosterhoff, J. H. F., van den Bekerom, M. P. J., Doornberg, J. N., van Haarst, E. P., van der Zee, J. A., van Andel, G. A., Lagerveld, B. W., Hovius, M. C., Kauer, P. C., Boevé, L. M. S., van der Kuit, A., Mallee, W., & Poolman, R. (2023). Implications of resampling data to address the class imbalance problem (IRCIP): an evaluation of impact on performance between classification algorithms in medical data. *JAMIA Open*, *6*(2). https://doi.org/10.1093/jamiaopen/ooad033

Wilson, D. L. (1972). Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-2*(3), 408–421. https://doi.org/10.1109/TSMC.1972.4309137

Wu, O. (2023). *Rethinking Class Imbalance in Machine Learning*. http://arxiv.org/abs/2305.03900

Zhang, J., & Mani, I. (2003). KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets*.

APPENDIX A

Table A1 – Adjusted p-values using Holm's method for control group ADASYN

Classifier	Metric	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	**$1.3 \cdot 10^{-4}$**	**$1.4 \cdot 10^{-8}$**	0.13	0.35	**$5.7 \cdot 10^{-4}$**	0.51	0.08	**$1.3 \cdot 10^{-12}$**	**$7.7 \cdot 10^{-12}$**
LR	F-SCORE	0.64	0.24	0.09	**0.02**	**0.01**	0.98	**$5.1 \cdot 10^{-3}$**	**$1.7 \cdot 10^{-3}$**	**$1.8 \cdot 10^{-4}$**
LR	G-MEAN	**$9.6 \cdot 10^{-5}$**	**$6.2 \cdot 10^{-6}$**	**$6.6 \cdot 10^{-6}$**	**0.03**	0.06	0.75	0.07	**$3.8 \cdot 10^{-9}$**	**$6.4 \cdot 10^{-10}$**
KNN	AUC	**$1.5 \cdot 10^{-7}$**	**$2.3 \cdot 10^{-19}$**	**$7.9 \cdot 10^{-8}$**	0.23	0.71	**0.04**	0.99	**$9.7 \cdot 10^{-14}$**	**$2.9 \cdot 10^{-13}$**
KNN	F-SCORE	**0.01**	**$4.2 \cdot 10^{-9}$**	0.92	**$5.7 \cdot 10^{-11}$**	0.13	0.09	0.33	0.33	1.32
KNN	G-MEAN	0.14	**$4.7 \cdot 10^{-15}$**	**0.01**	0.54	0.9	0.2	0.7	**$9.7 \cdot 10^{-10}$**	**$1.6 \cdot 10^{-10}$**
RF	AUC	**$4 \cdot 10^{-4}$**	**$1.1 \cdot 10^{-3}$**	**$9.9 \cdot 10^{-7}$**	**0.01**	**$7.2 \cdot 10^{-6}$**	**0.03**	0.45	**$5 \cdot 10^{-5}$**	**0.01**
RF	F-SCORE	**$1.9 \cdot 10^{-5}$**	**$4.2 \cdot 10^{-4}$**	**0.02**	**$9.2 \cdot 10^{-8}$**	**$4.4 \cdot 10^{-7}$**	0.38	0.45	**$2.7 \cdot 10^{-3}$**	**$1.4 \cdot 10^{-5}$**
RF	G-MEAN	0.06	**0.01**	**$7 \cdot 10^{-6}$**	**0.01**	**0.01**	0.45	0.55	**$4.3 \cdot 10^{-7}$**	**0.02**
XGB	AUC	**$1.5 \cdot 10^{-11}$**	0.21	0.21	**0.04**	**$6.8 \cdot 10^{-12}$**	**$4 \cdot 10^{-3}$**	**$1.4 \cdot 10^{-6}$**	0.57	**$4 \cdot 10^{-3}$**
XGB	F-SCORE	**$1.4 \cdot 10^{-10}$**	**$8.9 \cdot 10^{-4}$**	0.32	**$6.7 \cdot 10^{-10}$**	**$5.1 \cdot 10^{-10}$**	**0.02**	**$7.1 \cdot 10^{-8}$**	**$1.5 \cdot 10^{-5}$**	**$9.2 \cdot 10^{-9}$**
XGB	G-MEAN	**$2 \cdot 10^{-5}$**	0.86	0.08	0.06	**$5.3 \cdot 10^{-5}$**	0.73	**$6.1 \cdot 10^{-5}$**	0.16	0.16

Table A2 – Adjusted p-values using Holm's method for control group Near Miss

Classifier	Metric	ADASYN	G-SMOTE	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	**$1 \cdot 10^{-8}$**	**$9.2 \cdot 10^{-10}$**	**$3.5 \cdot 10^{-9}$**	**$4.4 \cdot 10^{-8}$**	**$1 \cdot 10^{-8}$**	**$1.1 \cdot 10^{-9}$**	**$7.4 \cdot 10^{-9}$**	**$9 \cdot 10^{-7}$**	**$9.5 \cdot 10^{-7}$**
LR	F-SCORE	0.24	0.13	1.36	0.7	**0.02**	0.2	**$2.8 \cdot 10^{-3}$**	**$2.9 \cdot 10^{-4}$**	**$4.1 \cdot 10^{-5}$**
LR	G-MEAN	**$3.5 \cdot 10^{-6}$**	**$1.4 \cdot 10^{-7}$**	**$2.1 \cdot 10^{-7}$**	**$5.8 \cdot 10^{-6}$**	**$7.9 \cdot 10^{-6}$**	**$4.3 \cdot 10^{-7}$**	**$3.1 \cdot 10^{-6}$**	**$2.8 \cdot 10^{-9}$**	**$6.2 \cdot 10^{-10}$**
KNN	AUC	**$1.8 \cdot 10^{-19}$**	**$3.2 \cdot 10^{-13}$**	**$2.7 \cdot 10^{-13}$**	**$1.9 \cdot 10^{-13}$**	**$4 \cdot 10^{-14}$**	**$1.6 \cdot 10^{-19}$**	**$1.1 \cdot 10^{-19}$**	**$6.6 \cdot 10^{-3}$**	0.81
KNN	F-SCORE	**$2.1 \cdot 10^{-9}$**	**$6.2 \cdot 10^{-13}$**	**$1.6 \cdot 10^{-10}$**	0.47	**$1.4 \cdot 10^{-13}$**	**$4.3 \cdot 10^{-9}$**	**$2.6 \cdot 10^{-11}$**	**$1.7 \cdot 10^{-10}$**	**$2.3 \cdot 10^{-7}$**
KNN	G-MEAN	**$4.7 \cdot 10^{-15}$**	**$1.1 \cdot 10^{-8}$**	0.08	**$2.2 \cdot 10^{-8}$**	**$1.5 \cdot 10^{-7}$**	**$7.9 \cdot 10^{-15}$**	**$7.1 \cdot 10^{-15}$**	0.16	**$8.2 \cdot 10^{-4}$**
RF	AUC	**$1.1 \cdot 10^{-3}$**	**$1.2 \cdot 10^{-8}$**	**$2 \cdot 10^{-6}$**	1.64	**$2.3 \cdot 10^{-8}$**	**$1.3 \cdot 10^{-4}$**	**$1 \cdot 10^{-3}$**	0.55	0.85
RF	F-SCORE	**$3.3 \cdot 10^{-4}$**	**$6.9 \cdot 10^{-9}$**	0.69	**$3.2 \cdot 10^{-7}$**	**$1.2 \cdot 10^{-9}$**	**$2.2 \cdot 10^{-3}$**	**$1.8 \cdot 10^{-3}$**	**$3.5 \cdot 10^{-6}$**	**$1.7 \cdot 10^{-10}$**
RF	G-MEAN	**0.01**	**$2.9 \cdot 10^{-3}$**	**$4.5 \cdot 10^{-6}$**	0.19	**$9.7 \cdot 10^{-4}$**	0.34	0.08	1.1	0.89
XGB	AUC	0.42	**$6.2 \cdot 10^{-10}$**	0.51	0.66	**$2 \cdot 10^{-9}$**	**$4.3 \cdot 10^{-4}$**	**$1.3 \cdot 10^{-6}$**	0.65	**$4 \cdot 10^{-3}$**
XGB	F-SCORE	**$5.9 \cdot 10^{-4}$**	**$1.8 \cdot 10^{-10}$**	**0.01**	**$1.1 \cdot 10^{-4}$**	**$6.3 \cdot 10^{-11}$**	**$9.8 \cdot 10^{-5}$**	**$2.3 \cdot 10^{-8}$**	**$1.5 \cdot 10^{-8}$**	**$1.6 \cdot 10^{-10}$**
XGB	G-MEAN	0.43	**$2.6 \cdot 10^{-4}$**	0.09	0.06	**$2.9 \cdot 10^{-4}$**	0.58	**$2.2 \cdot 10^{-5}$**	0.16	0.33

Table A3 – Adjusted p-values using Holm's method for control group ROS

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	0.21	1.03	**$4 \cdot 10^{-9}$**	**0.04**	0.78	0.26	0.51	**$4.2 \cdot 10^{-9}$**	**$4.4 \cdot 10^{-9}$**
LR	F-SCORE	0.06	**$2.3 \cdot 10^{-3}$**	0.87	0.27	**$3.6 \cdot 10^{-6}$**	**$3.3 \cdot 10^{-6}$**	**$1.7 \cdot 10^{-7}$**	**$2 \cdot 10^{-3}$**	**$1.3 \cdot 10^{-4}$**
LR	G-MEAN	**$5.5 \cdot 10^{-6}$**	**$1.3 \cdot 10^{-3}$**	**$2.5 \cdot 10^{-7}$**	1.31	0.89	**$7.6 \cdot 10^{-6}$**	**$6.8 \cdot 10^{-7}$**	**$4.4 \cdot 10^{-9}$**	**$9.1 \cdot 10^{-10}$**
KNN	AUC	**$6.6 \cdot 10^{-8}$**	**0.01**	**$6.1 \cdot 10^{-13}$**	**$1.4 \cdot 10^{-6}$**	**$1.1 \cdot 10^{-6}$**	**$3.3 \cdot 10^{-8}$**	**$1.3 \cdot 10^{-7}$**	**$1.5 \cdot 10^{-8}$**	**$2.8 \cdot 10^{-8}$**
KNN	F-SCORE	0.92	0.15	**$2.9 \cdot 10^{-10}$**	**$1.7 \cdot 10^{-9}$**	0.43	1.27	1.36	0.42	1.38
KNN	G-MEAN	**0.01**	0.07	**0.04**	**$4 \cdot 10^{-3}$**	**0.01**	**$3.1 \cdot 10^{-3}$**	**0.01**	**$2.9 \cdot 10^{-3}$**	**$7.5 \cdot 10^{-6}$**
RF	AUC	**$6.6 \cdot 10^{-7}$**	**$1.4 \cdot 10^{-9}$**	**$1.4 \cdot 10^{-6}$**	**$8.1 \cdot 10^{-5}$**	**$1.5 \cdot 10^{-9}$**	**$2.8 \cdot 10^{-6}$**	**$5.1 \cdot 10^{-6}$**	**$1.8 \cdot 10^{-4}$**	**$3.2 \cdot 10^{-7}$**
RF	F-SCORE	**0.01**	**$3.3 \cdot 10^{-5}$**	0.69	**0.01**	**$3.1 \cdot 10^{-6}$**	**$4.8 \cdot 10^{-3}$**	**0.01**	**$2.4 \cdot 10^{-3}$**	**$1.2 \cdot 10^{-6}$**
RF	G-MEAN	**$5.2 \cdot 10^{-6}$**	**$8.2 \cdot 10^{-7}$**	**$3.5 \cdot 10^{-6}$**	**$2.7 \cdot 10^{-3}$**	**$1 \cdot 10^{-6}$**	**$4.5 \cdot 10^{-6}$**	**$1.8 \cdot 10^{-5}$**	**$7.8 \cdot 10^{-4}$**	**$6.8 \cdot 10^{-4}$**
XGB	AUC	0.28	**$2.3 \cdot 10^{-8}$**	1.02	0.86	**$7.1 \cdot 10^{-8}$**	**$7.3 \cdot 10^{-4}$**	**$2.6 \cdot 10^{-5}$**	0.26	**0.01**
XGB	F-SCORE	0.32	**$1.5 \cdot 10^{-6}$**	**0.04**	**$1.7 \cdot 10^{-4}$**	**$5.3 \cdot 10^{-7}$**	**0.03**	**$2.6 \cdot 10^{-5}$**	**$1.8 \cdot 10^{-6}$**	**$5.5 \cdot 10^{-6}$**
XGB	G-MEAN	0.07	**$4 \cdot 10^{-5}$**	0.05	0.44	**$1 \cdot 10^{-4}$**	**$2.4 \cdot 10^{-4}$**	**$7.4 \cdot 10^{-4}$**	0.34	0.05

Table A4 – Adjusted p-values using Holm's method for control group RUS

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links	None
LR	AUC	1.04	1.02	**$1 \cdot 10^{-7}$**	**0.04**	0.89	1.18	0.9	**$4.7 \cdot 10^{-10}$**	**$6.8 \cdot 10^{-10}$**
LR	F-SCORE	**$9.4 \cdot 10^{-3}$**	**$5.8 \cdot 10^{-4}$**	0.7	0.27	**$2.2 \cdot 10^{-6}$**	**$9.3 \cdot 10^{-5}$**	**$5.6 \cdot 10^{-8}$**	**$4.7 \cdot 10^{-4}$**	**$4.7 \cdot 10^{-5}$**
LR	G-MEAN	**0.04**	0.66	**$2 \cdot 10^{-5}$**	0.66	1.13	**0.04**	0.12	**$4.7 \cdot 10^{-9}$**	**$6.4 \cdot 10^{-10}$**
KNN	AUC	0.31	**$6.7 \cdot 10^{-6}$**	**$3.3 \cdot 10^{-13}$**	**$4.2 \cdot 10^{-6}$**	0.33	0.82	0.48	**$4.7 \cdot 10^{-10}$**	**$4.2 \cdot 10^{-10}$**
KNN	F-SCORE	**$3.2 \cdot 10^{-11}$**	**$1.2 \cdot 10^{-15}$**	0.47	**$6.2 \cdot 10^{-10}$**	**$6.5 \cdot 10^{-18}$**	**$2.7 \cdot 10^{-11}$**	**$2.1 \cdot 10^{-13}$**	**$2.1 \cdot 10^{-10}$**	**$5.2 \cdot 10^{-7}$**
KNN	G-MEAN	0.54	**$1.4 \cdot 10^{-5}$**	**$3.6 \cdot 10^{-8}$**	**$3.3 \cdot 10^{-3}$**	0.65	0.35	0.42	**$1.6 \cdot 10^{-7}$**	**$1.6 \cdot 10^{-8}$**
RF	AUC	**0.02**	**$7.9 \cdot 10^{-8}$**	1.64	**$2.8 \cdot 10^{-4}$**	**$7.2 \cdot 10^{-6}$**	**$2.5 \cdot 10^{-4}$**	**0.02**	1.65	0.93
RF	F-SCORE	**$4.1 \cdot 10^{-8}$**	**$2.5 \cdot 10^{-10}$**	**$1.6 \cdot 10^{-7}$**	**$4.4 \cdot 10^{-3}$**	**$1.6 \cdot 10^{-11}$**	**$4.7 \cdot 10^{-8}$**	**$1.9 \cdot 10^{-7}$**	**$7.1 \cdot 10^{-9}$**	**$1.7 \cdot 10^{-12}$**
RF	G-MEAN	**0.02**	**$1.2 \cdot 10^{-3}$**	0.14	**0.02**	**$5.9 \cdot 10^{-4}$**	**$2.8 \cdot 10^{-3}$**	**0.02**	0.22	0.31
XGB	AUC	**0.03**	**$3.5 \cdot 10^{-11}$**	0.66	0.86	**$2.4 \cdot 10^{-10}$**	**$1 \cdot 10^{-5}$**	**$6.8 \cdot 10^{-7}$**	**0.01**	**$4.7 \cdot 10^{-4}$**
XGB	F-SCORE	**$3.3 \cdot 10^{-10}$**	**$5.4 \cdot 10^{-15}$**	**$5.4 \cdot 10^{-5}$**	**$4.3 \cdot 10^{-5}$**	**$1.6 \cdot 10^{-15}$**	**$5.2 \cdot 10^{-9}$**	**$3 \cdot 10^{-11}$**	**$4.3 \cdot 10^{-10}$**	**$1.5 \cdot 10^{-13}$**
XGB	G-MEAN	**0.04**	**$2.5 \cdot 10^{-5}$**	**0.03**	0.44	**$5.5 \cdot 10^{-5}$**	**$5 \cdot 10^{-5}$**	**$5 \cdot 10^{-4}$**	0.07	**0.03**

Table A5 – Adjusted p-values using Holm's method for control group SMOTE-ENN

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE Tomek	Tomek Links	None
LR	AUC	0.51	**0.02**	**$9.6 \cdot 10^{-10}$**	0.26	0.59	**$1.8 \cdot 10^{-3}$**	0.26	**$4.2 \cdot 10^{-13}$**	**$1.5 \cdot 10^{-12}$**
LR	F-SCORE	0.98	0.95	0.16	**$3.7 \cdot 10^{-6}$**	**$9.3 \cdot 10^{-5}$**	**$7.6 \cdot 10^{-3}$**	**$3.3 \cdot 10^{-5}$**	**$1.1 \cdot 10^{-4}$**	**$1.2 \cdot 10^{-5}$**
LR	G-MEAN	0.75	**$4.3 \cdot 10^{-4}$**	**$6.1 \cdot 10^{-7}$**	**$1.1 \cdot 10^{-5}$**	**0.03**	0.06	0.14	**$1.7 \cdot 10^{-9}$**	**$3 \cdot 10^{-10}$**
KNN	AUC	**0.04**	**$7.7 \cdot 10^{-8}$**	**$1.8 \cdot 10^{-19}$**	**$3.3 \cdot 10^{-8}$**	0.82	**0.4**	0.17	**$2.9 \cdot 10^{-12}$**	**$7.4 \cdot 10^{-12}$**
KNN	F-SCORE	0.05	**$2 \cdot 10^{-5}$**	**$1.2 \cdot 10^{-8}$**	0.63	**$4.1 \cdot 10^{-11}$**	**$3.7 \cdot 10^{-4}$**	**$2.6 \cdot 10^{-3}$**	**0.03**	0.64
KNN	G-MEAN	0.2	**0.04**	**$1 \cdot 10^{-14}$**	**$2.7 \cdot 10^{-3}$**	0.71	0.72	0.45	**$2.8 \cdot 10^{-10}$**	**$1.9 \cdot 10^{-10}$**
RF	AUC	0.05	0.68	**$1.8 \cdot 10^{-4}$**	**$6.2 \cdot 10^{-6}$**	**$2.5 \cdot 10^{-4}$**	0.12	**0.02**	**$1.7 \cdot 10^{-4}$**	**$2.2 \cdot 10^{-3}$**
RF	F-SCORE	0.75	**0.04**	**$4.3 \cdot 10^{-3}$**	**$4.8 \cdot 10^{-3}$**	**$8.4 \cdot 10^{-8}$**	**$1.3 \cdot 10^{-4}$**	0.93	0.42	**$2.9 \cdot 10^{-3}$**
RF	G-MEAN	0.9	0.56	0.57	**$8.2 \cdot 10^{-6}$**	**$3.2 \cdot 10^{-3}$**	0.07	0.81	0.78	1.09
XGB	AUC	**$2 \cdot 10^{-3}$**	**$3.9 \cdot 10^{-4}$**	**$3.6 \cdot 10^{-4}$**	**$4.9 \cdot 10^{-4}$**	**$1.4 \cdot 10^{-5}$**	**$4.1 \cdot 10^{-4}$**	**0.01**	**$6.4 \cdot 10^{-6}$**	0.19
XGB	F-SCORE	**0.02**	**$1.2 \cdot 10^{-7}$**	**$1.3 \cdot 10^{-4}$**	**0.02**	**$1.5 \cdot 10^{-8}$**	**$1.1 \cdot 10^{-7}$**	**$1.1 \cdot 10^{-7}$**	**$3.1 \cdot 10^{-4}$**	**$6 \cdot 10^{-6}$**
XGB	G-MEAN	0.73	**0.04**	0.87	**$2.4 \cdot 10^{-4}$**	**$5 \cdot 10^{-5}$**	**0.02**	0.13	**$2 \cdot 10^{-6}$**	1.35

Table A6 – Adjusted p-values using Holm's method for control group SMOTE-Tomek

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	Tomek Links	None
LR	AUC	0.09	0.26	**$8.6 \cdot 10^{-9}$**	0.34	0.9	**0.04**	0.35	**$1 \cdot 10^{-13}$**	**$4.2 \cdot 10^{-13}$**
LR	F-SCORE	**$2.1 \cdot 10^{-3}$**	**0.01**	**$1.6 \cdot 10^{-3}$**	**$1.5 \cdot 10^{-7}$**	**$5.6 \cdot 10^{-8}$**	0.07	**$2.3 \cdot 10^{-5}$**	**$6.5 \cdot 10^{-6}$**	**$9 \cdot 10^{-7}$**
LR	G-MEAN	0.1	0.11	**$6.2 \cdot 10^{-6}$**	**$8 \cdot 10^{-7}$**	0.09	0.13	0.07	**$6.8 \cdot 10^{-9}$**	**$1.1 \cdot 10^{-9}$**
KNN	AUC	0.99	**$6 \cdot 10^{-9}$**	**$1.1 \cdot 10^{-19}$**	**$1.6 \cdot 10^{-7}$**	0.48	0.95	0.22	**$9.2 \cdot 10^{-14}$**	**$2.8 \cdot 10^{-13}$**
KNN	F-SCORE	0.43	0.06	**$2.9 \cdot 10^{-11}$**	0.45	**$2.6 \cdot 10^{-13}$**	0.38	**$3.7 \cdot 10^{-3}$**	0.6	0.67
KNN	G-MEAN	0.7	0.15	**$7.9 \cdot 10^{-15}$**	**0.02**	0.62	0.98	0.59	**$4.5 \cdot 10^{-9}$**	**$9.6 \cdot 10^{-10}$**
RF	AUC	0.45	**0.02**	**$1.5 \cdot 10^{-3}$**	**$1.5 \cdot 10^{-5}$**	**0.02**	**$1.4 \cdot 10^{-4}$**	**0.02**	**$1.9 \cdot 10^{-4}$**	**0.01**
RF	F-SCORE	0.45	**0.02**	**$5.3 \cdot 10^{-3}$**	**0.01**	**$8.6 \cdot 10^{-7}$**	**$5.6 \cdot 10^{-5}$**	0.93	0.37	**$4.4 \cdot 10^{-3}$**
RF	G-MEAN	0.55	0.15	0.08	**$4.1 \cdot 10^{-5}$**	**0.03**	**0.02**	0.81	0.07	0.45
XGB	AUC	**$1.2 \cdot 10^{-6}$**	0.13	**$1.3 \cdot 10^{-6}$**	**$1.9 \cdot 10^{-5}$**	**$7.8 \cdot 10^{-7}$**	**$3.8 \cdot 10^{-3}$**	**0.01**	**$5.4 \cdot 10^{-7}$**	**$2.5 \cdot 10^{-3}$**
XGB	F-SCORE	**$9.9 \cdot 10^{-8}$**	**$7 \cdot 10^{-3}$**	**$3.7 \cdot 10^{-8}$**	**$2.6 \cdot 10^{-5}$**	**$4.5 \cdot 10^{-11}$**	**$1.1 \cdot 10^{-3}$**	**$8.6 \cdot 10^{-8}$**	0.09	0.12
XGB	G-MEAN	**$6.1 \cdot 10^{-5}$**	0.36	**$2.2 \cdot 10^{-5}$**	**$7.4 \cdot 10^{-4}$**	**$5.8 \cdot 10^{-4}$**	0.1	0.07	**$6.9 \cdot 10^{-4}$**	0.06

Table A7 – Adjusted p-values using Holm's method for control group Tomek Links

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	None
LR	AUC	**$8.7 \cdot 10^{-13}$**	**$2.8 \cdot 10^{-12}$**	**$1.8 \cdot 10^{-6}$**	**$1.4 \cdot 10^{-9}$**	**$2.1 \cdot 10^{-10}$**	**$1.5 \cdot 10^{-14}$**	**$3.3 \cdot 10^{-13}$**	**$9.2 \cdot 10^{-14}$**	**$5.8 \cdot 10^{-4}$**
LR	F-SCORE	**$4.2 \cdot 10^{-4}$**	**$1.9 \cdot 10^{-4}$**	**$2.2 \cdot 10^{-4}$**	**$3.9 \cdot 10^{-4}$**	**$4.7 \cdot 10^{-4}$**	**$6.8 \cdot 10^{-5}$**	**$1.6 \cdot 10^{-4}$**	**$9.7 \cdot 10^{-6}$**	**$5.8 \cdot 10^{-4}$**
LR	G-MEAN	**$2.4 \cdot 10^{-9}$**	**$1.8 \cdot 10^{-9}$**	**$2.1 \cdot 10^{-9}$**	**$2.2 \cdot 10^{-9}$**	**$1.8 \cdot 10^{-9}$**	**$2.2 \cdot 10^{-9}$**	**$1.7 \cdot 10^{-9}$**	**$1.7 \cdot 10^{-9}$**	**$1.1 \cdot 10^{-4}$**
KNN	AUC	**$8.5 \cdot 10^{-14}$**	**$1.4 \cdot 10^{-11}$**	**$3.3 \cdot 10^{-3}$**	**$5.8 \cdot 10^{-9}$**	**$2.7 \cdot 10^{-10}$**	**$2.4 \cdot 10^{-14}$**	**$2.1 \cdot 10^{-12}$**	**$9.2 \cdot 10^{-14}$**	**$3.2 \cdot 10^{-5}$**
KNN	F-SCORE	0.33	0.88	**$2.6 \cdot 10^{-10}$**	0.35	**$4.2 \cdot 10^{-10}$**	1.01	**0.04**	0.59	**$1.1 \cdot 10^{-3}$**
KNN	G-MEAN	**$1.1 \cdot 10^{-9}$**	**$4.7 \cdot 10^{-8}$**	0.16	**$7.3 \cdot 10^{-4}$**	**$1.1 \cdot 10^{-7}$**	**$6.5 \cdot 10^{-7}$**	**$3.6 \cdot 10^{-10}$**	**$4.5 \cdot 10^{-9}$**	**$2.8 \cdot 10^{-6}$**
RF	AUC	**$3 \cdot 10^{-5}$**	**$1.5 \cdot 10^{-8}$**	0.55	**$7.3 \cdot 10^{-4}$**	0.55	**$8.8 \cdot 10^{-14}$**	**$1.5 \cdot 10^{-4}$**	**$1.3 \cdot 10^{-4}$**	0.42
RF	F-SCORE	**$3.4 \cdot 10^{-3}$**	**0.02**	**$5.6 \cdot 10^{-6}$**	**$2.4 \cdot 10^{-3}$**	**$1.1 \cdot 10^{-8}$**	**$2.9 \cdot 10^{-5}$**	0.14	0.25	**$4.8 \cdot 10^{-3}$**
RF	G-MEAN	**$4.3 \cdot 10^{-7}$**	**$4.3 \cdot 10^{-7}$**	1.1	**$2.3 \cdot 10^{-3}$**	0.65	**$4.9 \cdot 10^{-6}$**	0.78	0.06	0.66
XGB	AUC	0.57	**$8.1 \cdot 10^{-12}$**	0.43	0.26	**$7.6 \cdot 10^{-3}$**	**$8.9 \cdot 10^{-10}$**	**$4.3 \cdot 10^{-6}$**	**$4.2 \cdot 10^{-7}$**	0.08
XGB	F-SCORE	**$2.2 \cdot 10^{-5}$**	**$3.7 \cdot 10^{-3}$**	**$1.9 \cdot 10^{-8}$**	**$1.8 \cdot 10^{-6}$**	**$9.3 \cdot 10^{-10}$**	**$4.9 \cdot 10^{-4}$**	**$5.1 \cdot 10^{-4}$**	0.09	**0.02**
XGB	G-MEAN	0.12	**$8.2 \cdot 10^{-5}$**	0.08	0.17	0.15	**$1.5 \cdot 10^{-4}$**	**$2 \cdot 10^{-6}$**	**$8.2 \cdot 10^{-4}$**	0.11

Table A8 – Adjusted p-values using Holm's method for control group None

Classifier	Metric	ADASYN	G-SMOTE	Near Miss	ROS	RUS	SMOTE	SMOTE ENN	SMOTE Tomek	Tomek Links
LR	AUC	**$7.8 \cdot 10^{-12}$**	**$1.1 \cdot 10^{-11}$**	**$9.5 \cdot 10^{-7}$**	**$1.9 \cdot 10^{-9}$**	**$3.4 \cdot 10^{-10}$**	**$7 \cdot 10^{-14}$**	**$1.3 \cdot 10^{-12}$**	**$4.2 \cdot 10^{-13}$**	**$5.8 \cdot 10^{-4}$**
LR	F-SCORE	**$6 \cdot 10^{-5}$**	**$2 \cdot 10^{-5}$**	**$2.1 \cdot 10^{-5}$**	**$4.4 \cdot 10^{-5}$**	**$2.7 \cdot 10^{-5}$**	**$7.5 \cdot 10^{-6}$**	**$1.1 \cdot 10^{-5}$**	**$1.2 \cdot 10^{-6}$**	**$1.9 \cdot 10^{-4}$**
LR	G-MEAN	**$3.5 \cdot 10^{-10}$**	**$2.9 \cdot 10^{-10}$**	**$4.1 \cdot 10^{-10}$**	**$3 \cdot 10^{-10}$**	**$2.8 \cdot 10^{-10}$**	**$2.7 \cdot 10^{-10}$**	**$3 \cdot 10^{-10}$**	**$2.4 \cdot 10^{-10}$**	**$1.1 \cdot 10^{-4}$**
KNN	AUC	**$2.9 \cdot 10^{-13}$**	**$1.1 \cdot 10^{-11}$**	0.81	**$1.2 \cdot 10^{-8}$**	**$2.1 \cdot 10^{-10}$**	**$3.9 \cdot 10^{-14}$**	**$6.3 \cdot 10^{-12}$**	**$3.2 \cdot 10^{-13}$**	**$3.2 \cdot 10^{-5}$**
KNN	F-SCORE	1.3	0.09	**$1 \cdot 10^{-6}$**	0.69	**$2.1 \cdot 10^{-6}$**	0.52	1.93	1.35	**$1.1 \cdot 10^{-3}$**
KNN	G-MEAN	**$1.9 \cdot 10^{-10}$**	**$8.2 \cdot 10^{-10}$**	**$2.7 \cdot 10^{-4}$**	**$2.5 \cdot 10^{-6}$**	**$9 \cdot 10^{-9}$**	**$2.1 \cdot 10^{-8}$**	**$1.9 \cdot 10^{-10}$**	**$7.2 \cdot 10^{-10}$**	**$1.9 \cdot 10^{-6}$**
RF	AUC	**0.01**	**$1.4 \cdot 10^{-8}$**	1.7	**$3.2 \cdot 10^{-7}$**	0.93	**$4.1 \cdot 10^{-9}$**	**$2.6 \cdot 10^{-3}$**	**0.01**	0.62
RF	F-SCORE	**$1.2 \cdot 10^{-5}$**	0.1	**$1.5 \cdot 10^{-10}$**	**$9.5 \cdot 10^{-7}$**	**$1.7 \cdot 10^{-12}$**	**0.01**	**$2.1 \cdot 10^{-3}$**	**$2.5 \cdot 10^{-3}$**	**$3.6 \cdot 10^{-3}$**
RF	G-MEAN	**0.02**	**$2.8 \cdot 10^{-9}$**	0.89	**$1.6 \cdot 10^{-3}$**	0.63	**$1.3 \cdot 10^{-6}$**	1.09	0.74	1.31
XGB	AUC	**$3.5 \cdot 10^{-3}$**	**$1.1 \cdot 10^{-5}$**	**$3.9 \cdot 10^{-3}$**	**$5 \cdot 10^{-3}$**	**$6.6 \cdot 10^{-4}$**	**$6.2 \cdot 10^{-9}$**	0.19	**$3.7 \cdot 10^{-3}$**	**0.04**
XGB	F-SCORE	**$1.1 \cdot 10^{-8}$**	**0.04**	**$1.6 \cdot 10^{-10}$**	**$5.5 \cdot 10^{-6}$**	**$2 \cdot 10^{-13}$**	**$2.9 \cdot 10^{-3}$**	**$6 \cdot 10^{-6}$**	0.06	**0.04**
XGB	G-MEAN	0.16	**$4.6 \cdot 10^{-4}$**	0.22	0.06	**0.04**	**$1.3 \cdot 10^{-4}$**	0.67	0.07	0.09

Table A9 – Guidelines based on dataset meta-features

Dataset meta-features	**Best Method**
F1v∈]0.053, 0.192] & inst2attr>35.2 & IR>4.4	G-SMOTE
F1v>0.727 & IR>16.19	G-SMOTE
F1v≤0.053 & #instances>7328 & inst2attr>51.6	SMOTE Tomek
F1v∈]0.192, 0.727] & IR>16.19	SMOTE
F1v≤0.053 & #instances≤2694	SMOTE
F1v>0.192 & IR≤16.19 & #instances>874 & #attr<=30 & inst2attr<=33	ROS
F1v∈]0.053, 0.129] & inst2attr∈]35.2, 60.3] & IR≤4.4	Near Miss
F1v>0.192 & IR≤8 & #instances>874 & #attr>30	None

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa