PROBABILISTIC DIAGNOSTIC MODEL FOR HANDLING CLASSIFIER DEGRADATION IN MACHINE LEARNING

by

Gustavo A. Valencia-Zapata

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

School of Electrical and Computer Engineering

West Lafayette, Indiana

December 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Gerhard Klimeck, Chair
School of Electrical and Computer Engineering
Dr. Kadri O. Ersoy
School of Electrical and Computer Engineering
Dr. Vinayak A. Rao
Department of Statistics
Dr. Michael G. Zentner
School of Electrical and Computer Engineering

Approved by:

Dimitrios Peroulis
Head of the Graduate Program

To Flor, my sea flower.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Dr. Gerhard Klimeck, Dr. Michael G. Zentner, and Dr. Okan Ersoy. I wish to thank my family for their unconditional love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Author: Valencia-Zapata, Gustavo, A. Ph.D.
Institution: Purdue University
Degree Received: December 2019
Title: Probabilistic Diagnostic Model for Handling Classifier Degradation in Machine Learning.
Committee Chair: Dr. Gerhard Klimeck

Several studies point out different causes of performance degradation in supervised machine learning. Problems such as class imbalance, overlapping, small-disjuncts, noisy labels, and sparseness limit accuracy in classification algorithms. Even though a number of approaches either in the form of a methodology or an algorithm try to minimize performance degradation, they have been isolated efforts with limited scope. Most of these approaches focus on remediation of one among many problems, with experimental results coming from few datasets and classification algorithms, insufficient measures of prediction power, and lack of statistical validation for testing the real benefit of the proposed approach. This research consists of three main parts: In the first part, a novel probabilistic diagnostic model based on identifying signs and symptoms of each problem is presented. Thereby, early and correct diagnosis of these problems is to be achieved in order to select not only the most convenient remediation treatment but also unbiased performance metrics. Secondly, the behavior and performance of several supervised algorithms are studied when training sets have such problems. Therefore, prediction of success for treatments can be estimated across classifiers. Finally, a probabilistic sampling technique based on training set diagnosis for avoiding classifier degradation is proposed.

1. INTRODUCTION

A diagnosis is an investigation or analysis of causes or nature of a condition, situation, or problem. For example, in Health Care the diagnostic process gathers information related to a patient health problem in order to plan the best path or treatment. Recent retrospective studies of adult healthcare and post-mortem examinations show a diagnostic error in up to 17% of hospital adverse events and in approximately 10% of patient deaths [1]. Therefore, several efforts in Health Care focus on improving diagnosis such as reducing uncertainty, early diagnosis, and second opinion [2], [3]. Using the analogy of "Medical Diagnosis", it is plausible to argue that the concept of "Diagnostic" as human cognitive function can be incorporated to the Artificial Intelligence domain. This research investigates whether the implementation of diagnostic tests aids to characterize the most common problems on classification tasks in order to select the most convenient treatment for avoiding classification degradation. Thereby, the questions now arise:

- What are the most representative problems on classification? there is any correlation, causality or association between them?
- Is it possible to measure a direct impact of each problems to the classifier degradation?
- Is there any process or technique to diagnose training datasets and subsequent remediation (treatments) selection before the training stage?

Mainly there are 5 degradation problems related to training datasets that limit the learning process: *class imbalance*, *sparseness*, *small-disjuncts*, *overlapping*, and *noise labels*. The class imbalance is one of the most recurrent problems and has received big interest on the artificial intelligence field for the last years [4]–[13]. A dataset is imbalanced when its class distribution or classification categories do not have similar proportions in terms of the number of instances. That situation is a problem because classification algorithms are not designed to handle unbalanced datasets and tend to favor the major classes over minor ones [14]. A second important issue related to class imbalance is the performance measurement. The most used performance metric for classification algorithms is the area under the ROC curve (AUC) [15], which may provide an

overly optimistic performance evaluation [16] and wrong performance evaluation for unbalance cases [16].

Several questions have arisen about the class imbalance problem:

- What is the optimal class distribution for training a classification algorithm?
- What are treatments for handling class imbalance?
- Is the class imbalance only responsible for classifier degradation?

Several approaches [17]–[19] have tackled these questions applying resampling (oversampling and undersampling) to reach the balance between classes modifying the distribution of the dataset. More sophisticated techniques change that distribution generating synthetic data as a balance strategy, not only for reducing overfitting and improving performance in classification tasks, but also for handling the sparseness in the feature space [13]. On the other hand, cost-sensitive learning treat the class imbalance problem arguing that each type of error has its own costs [20], [21]. Therefore, instead of changing the distribution of the whole dataset, cost-sensitive based models assume that misclassification cost for the observed dataset is known and use a "cost matrix" for describing it. Different approaches suggest that class imbalance is not completely responsible for degradation and argue that the phenomena of small-disjuncts contributes to overfit and misclassification [12], [22]–[26]. Small-disjuncts are subclasses of few instances inside the minor class. Other studies stated that the "*overlapping*" regions between classes [9], [27] and "*noisy labels"* due mislabeling [28] have strong negative effect over the classification algorithms performance. Different approaches based on ensemble methods [6], [11], [28]–[31] combine multiple learning algorithms and strategies for handling class imbalance, noisy labels detection [28], and improve performance.

This research presents a comprehensive empirical study of degradation problems in supervised learning through datasets from real-world domains. The thesis is organized as follows: Chapter 2 describes the importance of diagnostic methods in machine learning, research aims and approach. Chapter 3 describes the five major degradation problems and remediation techniques for handling degradation in supervised learning algorithms. Chapter 4 presents the implementation of a probabilistic diagnostic model for classifiers degradation, which supports the framework proposed in this research. Chapter 5 describes a new technique for handling degradation problems called "PSATE" (**P**robabilistic **SA**mpling **TE**chnique), which is directly related to the diagnostic model

outputs. Chapter 6 presents experimental results over 49 datasets across classification algorithms and techniques for handling degradation problems. Finally, Chapter 7 presents general conclusions and recommendations for future work.

Additionally, Appendices present information related to equations and details of the diagnostic model, information of the datasets involved in the experiments, performance metrics descriptions, and results associated with nonparametric statistic tests.

2. RESEARCH PROBLEM: LACK OF DIAGNOSIS

The motivation of the present work is to reduce degradation on classification algorithms caused by inherent problems associated with datasets. Previous studies make evident the lack of diagnosis for training sets before building classifiers. Although there exist methodologies or techniques that try to mitigate degradation, most of those efforts are blind to the characteristics of the datasets.

2.1 Importance of Diagnostic Methods in Supervised Learning

The diagnostic process in Health Care has several components [32] such as stages related to information gathering, diagnostic testing, clinical history, etc. This research uses similar components applied to the artificial intelligence domain, specifically associated with supervised learning. In other words, the diagnostic process starts with the "dataset" or target, who experiences a problem. After the dataset engages with the process, information is gathered from several sources (characteristics, testing, records, best practices, and so on) for building a diagnosis. Consecutively, an explanation of the problem is built. Finally, a treatment is defined based on the diagnosis and its outcomes are available as inputs to the diagnostic process.

Two fundamental principles are adopted from Health Care:

- **Correct diagnosis avoid inappropriate treatments** [32]. Even though a number of approaches (treatments) either in the form of a methodology or a technique try to minimize performance degradation in supervised learning, they have been isolated efforts without the input from a diagnostic process.
- **Treatment procedures according to the needs and characteristics of individual patients** [33]–[35]. Treatment effectiveness for classifier degradation does not exclusively depend on the degradation problem (class imbalance, sparseness, small-disjuncts, overlapping, and noise labels), but also characteristics associated with the training set are relevant for treatment selection and reaching optimal results.

2.2 Research Aims and Approach

Given the lack of diagnostic process in supervised learning, this research focuses on the study of degradation problems, as well as the statistical approach for measuring their criticality levels. Classification tasks usually involve different kinds of challenges: complexity of the dataset, algorithm selection, and performance metrics for classification algorithms. This study aims to answer the following question: How do diagnostic model reduce performance degradation in supervised learning?

The objective of this research is to prove that the use of diagnostic procedures before the training stage provides useful information for decision making and handling recurrent degradation problems and challenges in supervised learning. Thus, two hypotheses are defined:

- Hypothesis 1: it is not possible to find one treatment that is the best in remediation for all degradation problems, datasets or classifiers [36], [37]. The selection of the "best treatment" or even the most convenient classification algorithm depends on the available information and knowledge associate with the target dataset.
- Hypothesis 2: the diagnostic process helps to minimize performance degradation. A correct diagnosis provides useful information for planning the best path of remediation, it can include several methodologies or techniques to mitigate detected problems.

3. DEGRADATION PROBLEMS FOR SUPERVISED LEARNING

Data complexity can be associated with issues as class imbalance, overlapping, sparseness, and others [38]. This chapter presents five degradation problems for supervised learning and their most well-known remediation techniques. Those degradation problems damage can range from hampering the training stage to dramatically reduce classification algorithm performance.

3.1 Class Imbalance

Imbalanced data is present in many fields and represents a continuous challenge for Data Mining and Artificial Intelligence tasks. For instance, in medical diagnosis some diseases are infrequent or hard to detect since the limited number of cases. Fraud detection is a well-known case of class imbalance, where the number of fraud transactions is much smaller than the legitimate ones. As previously mentioned, part of the misclassification levels in supervised learning tasks may be attributed to the effect of the class imbalance and the fact that most of the classifiers are not designed to address this effect. Class imbalance occurs when one or more classes from the training set have few instances as compared to other classes. Usually, the level of class imbalance is measured using the *Imbalance Ratio* (IR), which indicates the relationship between two classes that expresses how much bigger one is than the other. For example, in a binary classification task (dataset with two classes), the class with few observations or minority class is called the "Positive class". On the other hand, the majority class is called the "Negative class" and the IR is estimated using the ratio between the total number of observations for each class. Figure 3.1 (a) shows an example of a two-class dataset in two dimensions, where the issue of class imbalance is easy to detect.

Generally, the class imbalanced problem has been addressed since two main approaches: methods based on resampling (oversampling and undersampling) and synthetic data techniques and cost-sensitive methods. In the case of the latter approach, it argues that errors do not have same cost. The present research focus on the resampling and synthetic data techniques, because cost-sensitive techniques assume the prior knowledge about the cost of misclassification for each class, which is a condition unknown in most of the cases and defined by the nature and domain of the dataset. The most representative resampling methods are oversampling and undersampling [39],

[40], these two techniques seek to reach the balance between classes modifying the distribution of the dataset. The first replicates randomly some instances from the Negative class, this is shown in Figure 3.1 (b) by the color intensity of the observation for the positive class. More color intensity means that the observation was sampled more times. The main disadvantages of oversampling are the overfitting [13], [40] and the low performance for unseen testing instances at the training stage [41]. Contrary, undersampling subtracts a random sample of a specified size from the observations of the Negative class (Refer Figure 3.1 (c)), leading losing important information and limiting the learning process for such class. C. Drummond et al. [42] tested oversampling and undersampling techniques over class imbalance datasets using the decision tree learner C4.5, showing better performance for classification tasks when undersampling technique was selected. However, more modern strategies such as *Synthetic Minority Oversampling Technique* (SMOTE) [13], *Borderline-SMOTE* [14], *Adaptative Synthetic Sampling Approach for Imbalanced Learning* (ADASYN) [43], *Density-base SMOTE* (DBSMOTE) [44], and *BalanceCascade* and *EasyEnsemble* [11] combine oversampling, undersampling, and algorithms in order to produce synthetic entries, reach balance, avoid overfitting, and obtain better performance.

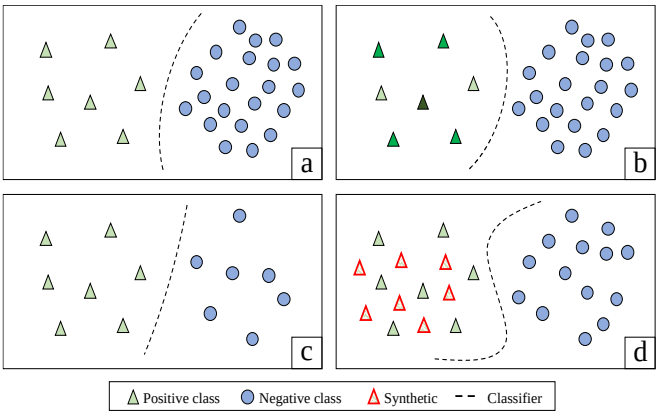


Figure 3.1 Class Imbalance and basic solutions: example for a binary classification. (a) Two-class dataset in two dimensions, (b) oversampling over the positive class, (c) undersampling over the Negative class, and (d) combination of resampling and synthetic data

3.2 Sparseness

Several studies argue that the imbalance issue in data should be understood as the level of sparseness and the low degree of instances for some classes [13], [14], [17]. Therefore, synthetic data techniques such as SMOTE [13] not only generates new synthetic entries for balancing, but

also builds larger and better defined decision regions for the minority class filling holes in the space (Refer Figure 3.1 (d)).

3.3 Small-disjuncts

Holte et al. [22] introduced the term of small-disjuncts, which would be the major cause of learning problems in the positive class [22], [45]. The small-disjuncts could be interpreted as a direct consequence of the "Sparseness". In simple words, small-disjuncts are “sub-clusters” or subclasses of few instances inside the Positive class. Generally, the small-disjunct concept can be interpreted as the imbalance problem between-class and within-class, the between-class case refers the imbalanced between the Negative class and the Positive class, which is usually estimated by the IR. On the other hand, the within-class refers to how observations are distributed within each class, since a class could be composed by several subclasses with imbalanced in number of instances [23]–[26]. This research extends the concept of small-disjuncts to the Negative class. Therefore, using the new definition of small-disjuncts: both classes can have subclasses of few instances that difficult the training stage for classification algorithms. Figure 3.2 illustrates the small-disjuncts problems for both classes.

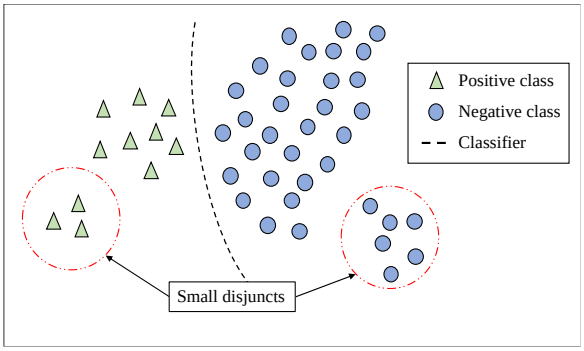


Figure 3.2 Small-disjuncts: example for a binary classification

3.4 Overlapping

Datasets can have ambiguous regions, which contain observations from two or more classes with similar probability. This problem is known as overlapping. Diverse studies claim that classifier degradation is not only affected by the class imbalance problem or small-disjuncts, but

perhaps even more so, in terms of the degree of overlapping [9], [27], [46]. Figure 3.3 shows the overlapping concept for a binary classification problem.

Garcia et al. [9] conducted several experiments over two-dimensional artificial datasets with two classes separated by a line orthogonal to one of the axis and six different classifiers. They found that by changing the IR of the overlapping region is more beneficial than changing the overall IR in the dataset. Although the simplicity of experiments is far from real datasets, the results point out the degree of overlap as an important factor in the process of building classifiers. Most of the treatments for handling Noisy labels focus on cleaning overlap regions by using the k-nearest neighbors' rules as a component of their procedure. The degradation problem below lists some of those techniques.

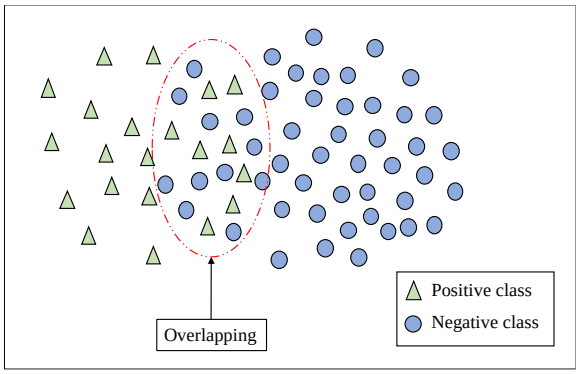


Figure 3.3 Overlapping: example for a binary classification.
Overlapping regions contain observations from two or more classes with similar probability

3.5 Noisy Labels

Labeling error is a recurrent problem in supervised learning datasets. This problem describes instances with wrong label assignation due to different causes (i.e. data-entry error, subjectivity, and lack of information for the labeling process), Figure 3.4 illustrates the noisy labels problem. Quinlan [47]–[49] conducted several experiments in order to measure the effect of class noise over decision trees performance, finding two patterns: class noise is more harmful than feature (characteristics) noise and cleaning noisy observations from training sets produces classifiers with better accuracy. *Condensed Nearest Neighbors* algorithm (CNN) [50] was the first attempt of filtering noisy observations or building a better subset from the training set using *Nearest Neighbor Rule* (NN Rule) [51], [52]. Several subsequent proposals based on the CNN concepts have been

discussed, such as *Edited Nearest Neighbors* (ENN) [53], *Neighborhood Cleaning Rule* NCL [54], Tomek [55], *Mutual Nearest Neighborhood* (MNN) [56], and *One-Side Selection* (OSS) [40]. For instance, Tomek algorithm extends that approach calling ENN for increasing the number of neighbors and focus on removing noisy observations from the Negative class close to the borderline with the Positive class and returning a better subset for the training stage. On the other hand, Brodley et al. [57] used Ensemble filters for detecting noisy observations. In this approach the filter is composed by a set of classifiers called base-level detectors, which receive the training set and detect the misclassified cases. Finally, the filter removes observations based on majority voting or consensus and generate a training subset.

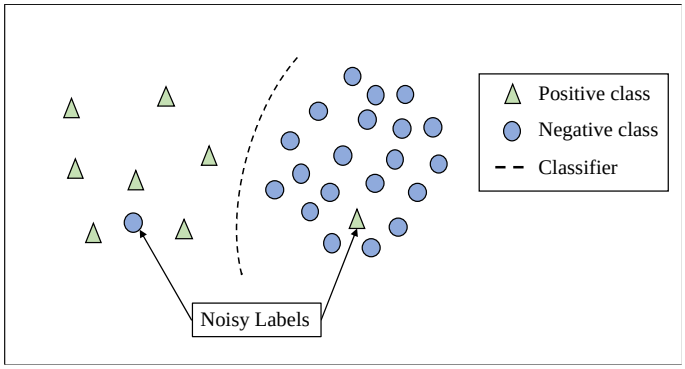


Figure 3.4 Noisy Labels: example for a binary classification.
Labeling error refers to instances with wrong label assignation

3.6 Treatments and side effects

As previously stated, there are numerous treatments in the form of methodology or technique that seeks to mitigate negative effects of the degradation problems. However, the effectiveness of such treatments may be reduced by the lack of diagnosis. The following is the list of techniques implemented in the Chapter 6 related to experimental framework.

- *Resampling* (Random)

  This technique combines oversampling and undersampling to obtain a balanced dataset. Naïve oversampling replicates randomly some instances from the Positive (minority) class. On the other hand, naïve undersampling removes randomly some instances from the Negative (majority) class.

- *Synthetic Minority Oversampling Technique* (SMOTE) [13]

  This algorithm generates synthetic instances of the Positive (minority) class based on the feature space similarities, using k-Nearest Neighbor algorithm. Besides, the Negative (majority) class instances are under-sampled in order to reach the balance between classes.
- *Borderline-SMOTE* (B-SMOTE) [14]

  This technique is an extension of SMOTE, which focus on generating synthetic Positive instances in the neighborhood of the borders.
- *Adaptative Synthetic Sampling Approach for Imbalanced Learning* (ADASYN) [43]

  This algorithm is an extension of SMOTE, which generates synthetic instances using a density distribution based on k-Nearest Neighbors algorithm. Thus, this technique automatically decides the number of synthetic instances that must be generated for each real Positive instance.
- *Density-base SMOTE (DBSMOTE)* [44]

  This technique initially clusters the Positive class using DBSCAN algorithm [58] and generates synthetic instances along a shortest path from each Positive instance to a pseudo-centroid of the cluster.
- *Edited Nearest Neighbors* (ENN) [53]

  Edited Nearest Neighbor removes any instance whose class label differs from the class of at least half of its k-Nearest Neighbors. Therefore, both classes may be under-sampled.
- Neighborhood Cleaning Rule NCL [54]

  This algorithm is a modification of ENN, which only excludes instances from the Negative class (majority) in two steps: first, it removes Negatives instances which are misclassified by their 3-Nearest Neighbors. Then, the neighbors of each Positive instance are found and the ones belonging to the Negative class are removed.
- *One-Side Selection* (OSS) [40]

  This technique combines *Tomek* links [55] and *Condensed Nearest Neighbor* (CNN) [50]. First, Tomek links focus on cleaning overlapping between classes by removing Negative instances using 1nearest neighbors. Then, CNN removes instances from both classes that are not correctly classifying by the 1-Nearest Neighbor rule.

This research categorizes the above treatments based on the degradation problems that intend to remedy and possible side effects. Table 3.1 summarizes such categorization. The symbol (+) refers to the problem that the treatment intends to remediate. Contrarily, the symbol (−) refers to negative side effects due to the treatment. For instance, Random combines undersampling and oversampling to obtain a balanced dataset; however, the first one removes instances that may be important for the training stage and the second one produces overfitting. A second very concrete example is OSS, which solves the problems of overlapping and noisy labels by removing instances. However, it reduces the amount of data, produces holes in the space, and may increase the class imbalance. On the other hand, SMOTE and its extensions not only regenerate synthetic instances for reaching balance between classes, but also fill the space of the Positive class. Therefore, those techniques reduce the sparseness effect and build better decision regions.

Table 3.1 Treatments: benefits and side effects.
The symbol (+) refers to the problem that the treatment intends to remediate.
Contrarily, the symbol (−) refers to negative side effects due to the treatment
Treatments target and side effects

	Degradation problem						
Treatment	**Class Imbalance**	**Sparseness**	**Small-disjuncts**	**Overlapping**	**Noisy Labels**	**Amount of data**	**Overfitting**
Random	(+)					(−)	(−)
SMOTE	(+)	(+)		(−)	(−)	(−)	
B-SMOTE	(+)					(−)	
DBSMOTE	(+)	(+)	(+)			(−)	
ADASYN	(+)	(+)				(−)	
ENN	(−)	(−)		(+)	(+)	(−)	
NCL	(+)	(−)		(+)	(+)	(−)	
OSS	(−)	(−)		(+)	(+)	(−)	

4. PROBABILISTIC DIAGNOSTIC MODEL FOR HANDLING CLASSIFIER DEGRADATION

After the main degradation problems and their remediation treatments have been mentioned, this Chapter presents a diagnostic model called “Probabilistic Diagnostic Model for Handling Classifier Degradation”, which allows to determine the level of degradation problems presence for classification datasets. References of previous datasets diagnosis efforts point to basic description of the dataset such as missing values, duplicate instances, and statistics (frequency, percentage, mean, standard deviation, etc.). Nevertheless, there is no explicitly reference related to tools, techniques, methodologies or any other approach for diagnosis of degradation problems on supervised learning.

4.1 Diagnostic Model Components

The diagnostic model consists of five main components, which have a direct connection with the five degradation problems discussed in Chapter 3. Those components are connected through a workflow where each component provides information to the next one during the diagnostic process. Figure 4.1 illustrates the sequence of the diagnostic model, where dataset (training set) is the input to the flow and the diagnostic report is the output. Subclasses detection is the first component, which is conducted for each class by separated. Then, the process runs several “diagnostic tests” that measure the criticality levels of the degradation problems. Finally, the output of the process is a diagnostic report.

4.1.1 Subclasses detection

Previous proposals have formulated strategies of segmenting for training sets [4], [24]–[26], [59], [60] by using algorithms such as Principal Direction Devise Partition (PDDP) [61], K-means [62], [63], and DBSCAN [58]. Those strategies focus on finding subclasses (clusters) for the positive class or even for the complete training set regardless the membership class. Nevertheless, the major drawbacks of those approaches are the lack of parameters associated with the subclasses description and the need of provide the number of subclasses (clusters) in the training set a priori.

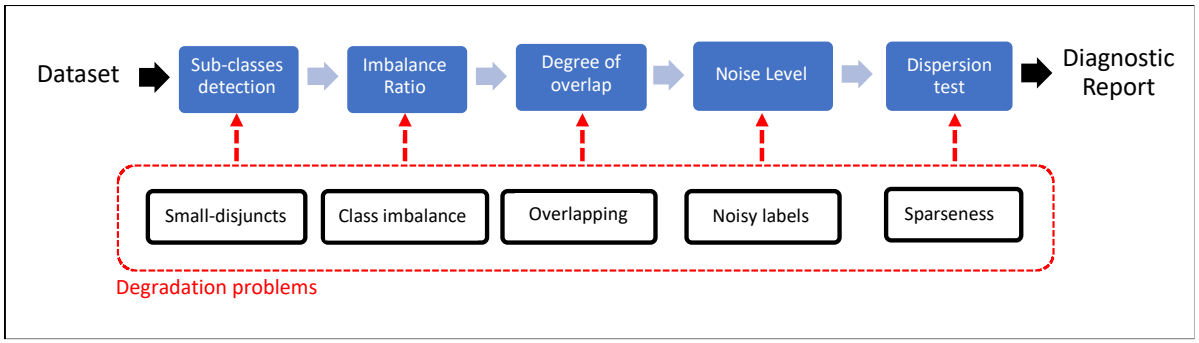


Figure 4.1 Diagnostic model process.
The training set is the input to the flow and the diagnostic report is the output

The first component of the Diagnostic model runs a segmentation process of partitioning each class into subclasses by separated. Such segmentation is implemented by Gaussian Mixture Models (GMM) [64], [65] and the EM algorithm [66], which is used to estimate the mixture parameters. The advantage of this approach is to obtain statistical parameters that describe each subclass for subsequent analysis. However, there is an inevitable, indeed necessary, question at the time of modeling a mixture of probabilistic distributions or any segmentation technique: how many clusters (subclasses) should be included in the model? A second question is related to the covariance matrix model selection for the GMM. For this research, the total number of subclasses per class and the covariance matrix model are determined by seeking to balance the increase in likelihood and the complexity of the mixture model, introducing a penalty term for each parameter. Then, the selection of the best model is addressed by the Bayesian Information Criterion (BIC) [67], [68], which is described with more details at Appendix A.

Once the number of subclasses per class is defined, it is possible not only moving to the next component of the diagnostic tool, but also knowing more detail about the complexity of the dataset in terms of small-disjuncts. Greater number of subclasses is associated with higher difficulty levels of classification [17], [26]. Moreover, it should be taken into consideration that not all detected subclasses are small-disjuncts, such category usually describes subclasses with few instances.

4.1.2 Imbalance Ratio (IR)

The IR component not only estimating the IR between classes, but also between subclasses. This information enables to focus any resampling or synthetic data process over specific subclasses in future remediation treatments. The IR is estimated as many times as the combination of detected subclasses taken 2 at a time without repetition.

4.1.3 Degree of Overlap

The third component measures the degree of overlap based on the *separation index* (J^*) proposed by Qiu et al. [69], [70]. This index measures the magnitude of the gap between pairs of subclasses. It has a value between -1 and $+1$, where negative values indicate subclasses are overlap, zero means subclasses are touching, and positive values indicate subclasses are separated. Experiments based on Multivariate Gaussian Distributions (MGD) [58] suggest that values larger than $0{,}21$ indicate subclasses are well-separated. The separation index is estimated as many times as the combination of subclasses taken 2 at a time without repetition. Further information about estimation process of the separation index estimation can be found at Appendix A.

One recurrent challenge of classification tasks is to build a mathematical function sufficiently flexible and able to discriminate instances even with high levels of overlapping or merge between classes. Therefore, some techniques focus on solving overlapping issues before the training stage. For instance, generating Positive instances in the overlap regions by reinforcing borderlines [14] or removing noisy instances from the Negative class [54], [55], [40]. The feasibility, advisability, and parametrization of those technique could be elements easy influenced by the separation index.

4.1.4 Noise level

The noise level component detects noisy instances by using the 3-Nearest Neighbor rule and estimates the noise ratio for the training dataset, the Positive class, and the Negative class. Since separation indexes between subclasses are known, this component is able to identity noisy instances between well-separated subclasses. In other words, this component detects noisy labels unrelated to the overlapping issue and estimates a ratio related to the noise due overlap issues.

4.1.5 Dispersion test

This component implements two procedures: the first one, estimates the degree of sparseness for each subclass by calculating the average and standard deviation of the distances (Euclidian, Mahalanobis, or Manhattan) to the subclass median. On the other hand, the second procedure implements the Anderson's test (PERMDISP2) [71]–[73] for multivariate homogeneity of groups dispersion, which is a multivariate analogue of Levene's test for homogeneity of variances [74]. Anderson's test seeks to validate if the dispersion of two or more subclasses is equal (similar sparseness).

4.2 Case in point Artificial Domain

This subsection presents a systematic example of the procedures and outputs of the diagnostic model. The purpose of this example is to provide a full understanding of the diagnostic tool components and outputs. The case in point considers a two-dimensional artificial dataset for classification, which has been generated on the basis of Gaussian distributions and three controlled parameters: size of the subclasses, degree of overlapping, and distributional parameters (mean vector and covariance matrix).

4.2.1 Output 1: Training set basic description

The first output is related to basic characteristic of the dataset under analysis. In the supervised learning case, such dataset is the training set. Table 4.1 shows the first output, which is a list of basic characteristics and values for the artificial training set. The Negative class with 404 instances represents 82.4% of the data. On the other hand, the Positive class with 86 instances represents 17.6% of the data. Therefore, this training set is said to have an Imbalance Ratio of $4.7{:}\,1$ or $\mathrm{IR} = 4.7$. In other words, there is one instance in the Positive class per 4.7 instances in the Negative class.

Table 4.1 Output 1: Basic description for the artificial training set

Dataset description	
Dimensions	2
Duplicate Instances	NA
Total instances	490
Instances Positive class	86
Instances Negative class	404
Imbalance Ratio (IR)	4.7

4.2.2 Output 2: Subclasses detection

Table 4.2 shows the next output related to the fist component of the diagnostic tool, which uses MGG for sub-subclass detection. For the Positive class two subclasses were detected using the covariance matrix model VEE (ellipsoidal, equal shape and orientation). On the other hand, the Negative class has three subclasses detected by the covariance matrix model VEI (diagonal, varying volume, equal shape).

Table 4.2 Output 2: Subclass detection for the artificial training set.
Selection of the best covariance matrix model and number of subclasses

Subclasses detection	
Positive class	
Instances	86
Covariance matrix model	VEE
Subclasses	2
Negative class	
Instances	404
Covariance matrix model	VEI
Subclasses	3

Figure 4.2 illustrates the best number of subclasses and top three covariance matrix models based on the BIC for the artificial dataset. In the Figure 4.2 (a) all covariance matrix models are in agreement with two subclasses for the Positive class, where covariance best model is VEE. In the Figure 4.2 (b) two out of three covariance matrix models are in agreement with three subclasses

for the Negative class and VEI is the best covariance model. For further information about covariance matrix models implemented in GMM may be consulted in the Appendix A.

Figure 4.3 makes a visual representation of the segmentation results through the GMM over the artificial dataset. The overall result is five subclasses, two for the Positive class and three for the Negative class.

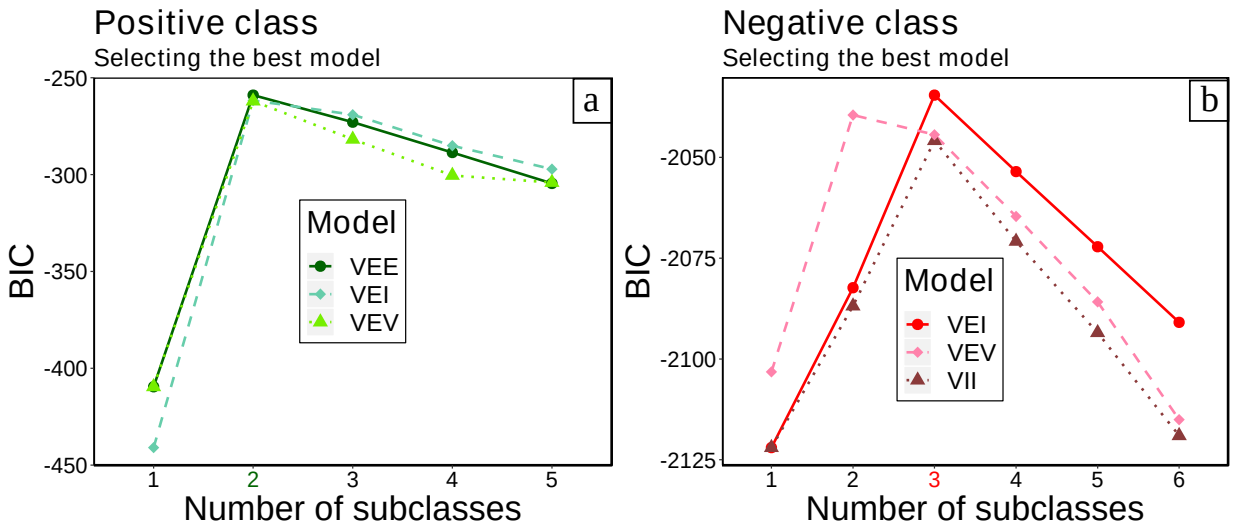


Figure 4.2 Number of subclasses and covariance matrix selection for the artificial training set. (a) The top three models for the Positive class are: VEE (ellipsoidal, equal shape and orientation), VEI (diagonal, varying volume, equal shape), and VEV (ellipsoidal, equal shape). (b) The top three models for the Negative class are: VEI (diagonal, varying volume, equal shape), VEV (ellipsoidal, equal shape), and VII (spherical, unequal volume)

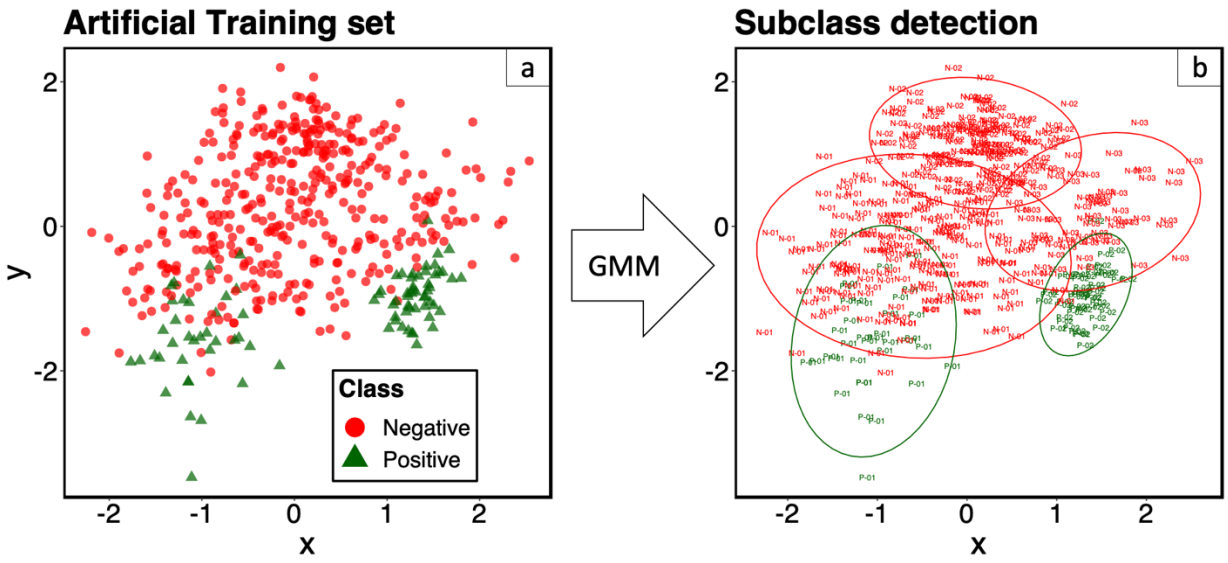


Figure 4.3 Detecting subclasses by using GMM for the artificial training set.
(a) The original training set. (b) Training set with five subclasses.
N = Negative subclasses and P = Positive subclasses

Figure 4.3 makes a visual representation of the segmentation results through the GMM over the artificial dataset. The overall result is five subclasses, two for the Positive class and three for the Negative class.

4.2.3 Output 3: IR and Overlap matrix (IRO)

Table 4.3 below shows the third output related to the IR and Overlap matrix (IRO matrix) for the artificial training set. This is a square matrix, whose size is determined by the number of subclasses. The upper triangular part shows the IR between subclasses. For instance, the IR between **N-01** (Negative subclass 01) and **N-03** (Negative subclass 03) is 3. That means, there is one instance in **N-03** per 3 instances in **N-01**. The diagonal of the IRO matrix contains the number of instances per subclass. For example, P-01 (Positive subclass 01) is the smaller subclass with 36 instances in the artificial training set. Finally, the lower triangular part shows the *separation index* J^* between subclasses. For instance, $J^* = 0.29$ between **N-02** and **P-02** 9, which makes sense since both subclasses are far from each other according to Figure 4.3 (b).

Table 4.3 Output 3: IR and Overlap matrix (IRO) for the artificial training set. The upper triangular part shows the IR between subclasses, the diagonal contains the number of instances per subclass, and the lower triangular part shows the *separation index* between subclasses. N = Negative subclasses and P = Positive subclasses

IRO matrix

	N-01	**N-02**	**N-03**	**P-01**	**P-02**
N-01	**189**	1.24	3	5.25	3.78
N-02	0	**152**	2.41	4.22	3.04
N-03	0	0	**63**	1.75	1.26
P-01	-0.3	0.21	0.2	**36**	0.72
P-02	0	0.29	-0.1	0.4	**50**

4.2.4 Output 4: Noise level

The noise level component has several tables as outputs. Table 4.4 (a) summarizes the number of noisy and valid instances per class for the artificial training set. Additionally, Table 4.4 (b) breakdowns the number of noisy instances per subclass and includes the noise location (last two columns). The column called “Noise overlap” refers to noise between subclasses with a *separation index* less than 0.2 (close subclasses). Then, this noise could be due to the overlap effect instead of mislabeled issues. On the other hand, the column called “Noise label” refers to noise between subclasses with a *separation index* larger or equal to 0.2 (well-separated subclasses). Therefore, noisy instances are likely associated with mislabeled issues. For the artificial training set, all noisy instances are associated with the overlap effect.

Table 4.4 Output 4: Noise per class and subclass for the artificial training set. a) Noisy and valid instances per class. b) Breakdowns the number of noisy instances per subclass and includes the noise location. N = Negative subclasses and P = Positive subclasses

Noise per class		
	Instances	
Class	**Noise**	**Valid**
Negative	15	389
Positive	12	74

(a)

Noise per subclass					
	Instances	**Valid**	**Noise**	**Noise overlap**	**Noise label**
N-01	189	177	12	12	0
N-02	152	152	0	0	0
N-03	63	60	2	3	0
P-01	36	26	10	10	0
P-02	50	48	2	2	0

(b)

Under the noise location concept, the diagnostic tool proposes three new metrics, which describes noise proportions for training sets (refer Table 4.5 (a)). The first metric is the Noise Ratio (NR), which is the quantitative relation between valid and all noisy instances. For instance, the artificial training set has a $NR = 17.14$. That means there is one noise instance per 17.14 valid instances. The second metric is the Noise Overlap Ration (NOR), that considers noisy instances associated with close subclasses. Finally, the Noise Label Ratio (NLR) considers noise related to well-separated subclasses. In the example, all noisy instances are associated with close subclasses. Therefore, $NR = NOR$ and NLR does not apply. Table 4.5 (b) shows the Noise matrix. This is a square matrix, whose size is determined by the number of subclasses. Each column of the matrix represents the instances in an estimated subclass based on the three-nearest neighbor rule while each row represents the instances in an actual subclass (original membership). Thus, the diagonal of the matrix contains the number of valid instances per subclass (instances that kept the same subclass label after the noise analysis). For instance, reading the first row: 176 instances from N-01 did not change their membership, one instance moved to N-03 (it is not considered noise because N-01 and N-03 belong to the same Negative class), 10 noisy instances from P-01, and 2 noisy instances from P-02.

Table 4.5 Output 4: Noise Ratios and Noise matrix for the artificial training set.
(a) Noise Rations estimates the relation between valid and noisy instances.
(b) Noise matrix shows the amount of noise between subclasses.
N = Negative subclasses and P = Positive subclasses

Noise Ratios	
Noise Ratio (NR)	17.14
Noise Overlap Ratio (NOR)	17.14
Noise Label Ratio (NLR)	NA

(a)

Noise matrix

Membership \ Noise Level estimation	N-01	N-02	N-03	P-01	P-02
N-01	**176**	0	1	10	2
N-02	2	**149**	1	0	0
N-03	1	1	**58**	0	3
P-01	10	0	0	**26**	0
P-02	0	0	2	0	**48**

(b)

4.2.5 Output 5: Dispersion test

The dispersion test component has two outputs. The first one presents the dispersion per subclass by estimating the average and standard deviation of the distances to the subclass median. For instance, Table 4.6 shows such dispersion per each class by separated. Specifically, P-01 and N-01 are the subclasses with larger dispersion values. Besides to the dispersion table, this first output includes dispersion plots per class. Figure 4.4 below presents such dispersions for the artificial training set, where visual sparseness not only coincides with the values from Table 4.6, but also with that observed in the Figure 4.3 (b).

Table 4.6 Output 5: Subclass dispersion for the artificial training set.
N = Negative subclasses and P = Positive subclasses

Dispersion per subclass		
	Mean	**Std**
N-01	0.96	0.46
N-02	0.63	0.32
N-03	0.65	0.33
	Mean	**Std**
P-01	1.05	0.85
P-02	0.57	0.41

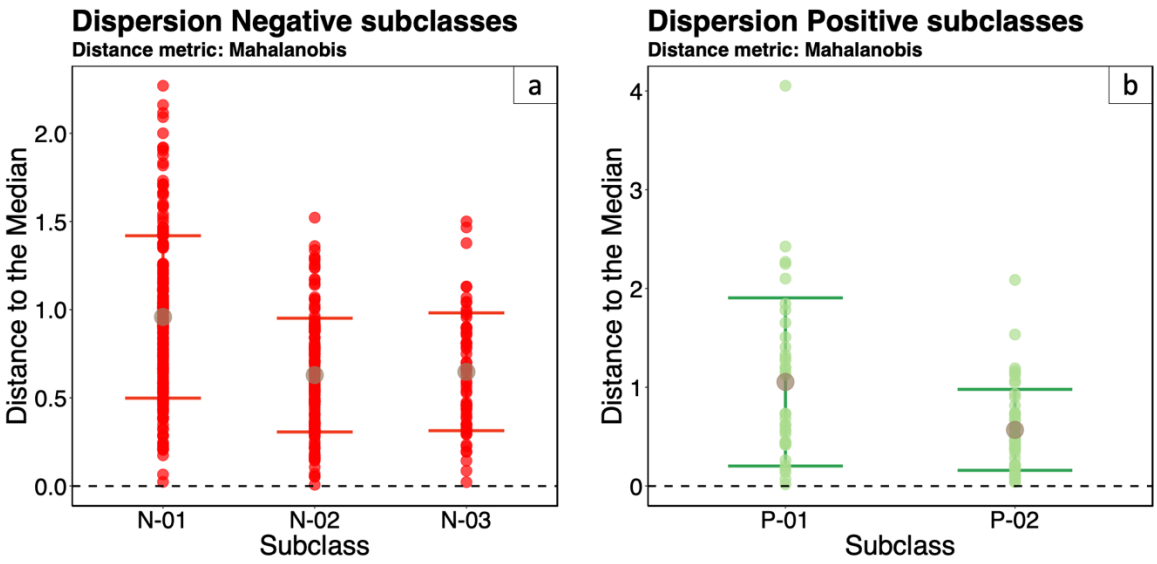


Figure 4.4 Subclasses dispersion for artificial training set.
(a) Dispersion for Negative subclasses. (b) Dispersion for Positive subclasses.
The dotted lines represent the median as a reference point. For each subclass plotted on the X axis, the distance of each observation to the median, is plotted in the Y axis. Where dark dots represent the mean of the distances to the median and the error bars the standard deviation

In case of detecting two or more subclasses within a class (refer subclass detection component), a second output shows the results of the Anderson's multivariate test for homogeneity of variance. The goal is to test the null hypothesis that assumes variance is equal across subclasses. A *p-value* less than 0.05 indicates a violation of the assumption. Therefore, same resampling or synthetic data treatments strategy cannot necessarily be generalized to all subclasses, because the variance is not the same across subclasses. For example, Table 4.7 makes evident the latter statement for the artificial dataset.

Table 4.7 Output 5: Anderson's test for homogeneity of variance for the artificial training set. Df (Degree of freedom), Sum Sq (Sum of squares), Mean Sq (Mean squared), F (The F-test), N.Perm (Number of permutations), and p-value (Probability value)

Test for homogeneity of multivariate dispersion

Negative class						
	Df	**Sum Sq**	**Mean Sq**	**F**	**N.Perm**	**p-value**
Subclasses	1	5	4.9	12.319	999	0.002
Residuals	84	34	0.4			

Positive class						
	Df	**Sum Sq**	**Mean Sq**	**F**	**N.Perm**	**p-value**
Subclasses	2	11	5.3	34.001	999	0.001
Residuals	401	63	0.2			

4.3 Diagnostic Model Algorithm

The Diagnostic model was develop using R open source software. Pseudo-code for the diagnostic model components and procedures is shown below.

Table 4.8 Diagnostic model algorithm

Algorithm 1: Diagnostic tests

Input : S is the training set, $X = \{x_1, x_2, ..., x_{i+1}, x_{i+2}, ...x_M\}$ are the instances of S
$Label$ is the class membership vector of X_i. Positive (P) or Negative (N)
k is the number of k-nearest neighbor
$C.min$ is the minimum num. of instances per component (subclass)
$G.max$ is the maximum num. of mixture components for which the BIC is calculated
$sp.th$ is the separation index threshold
$dist$ is the distance measure to be use. 1 = Mahalanobis, 2 = Manhattan, and 3 = Euclidean

Output : S_1 if $C.min$ is not covered, S_1 = *Subclass membership* for outlier analysis
S_2 if $C.min$ is covered, $S_2 = S_1$ + *Diagnostic report*

Diagnosis($S, k, C.min, G.max, sp.th, dist$)
/* Find subclasses by Gaussian Mixture Models */
foreach *Class* $\in S$ **do**
  Best Model = $\text{argmax}_{x \in Class}\ BIC(x)$, see Eq. (10)
  Let $S_1 = \{Subclass\ membership \cup S\}$
  $P_g = \{p_1, ..., p_{K_P}\}$, set of subclasses from P; K_P = Num. of subclasses in P
  $N_j = \{n_1, ..., n_{K_N}\}$, set of subclasses from N; K_N = Num. of subclasses in N
  $L_P = Length(P_g)$, vector with the length of each subclass P
  $L_N = Length(N_j)$, vector with the length of each subclass N
if *any* $(L_p, L_N) < C.min$ **then**
  Return S_1
else
  /* Calculate IR between subclasses */
  $Upper = IR(Sets)$, upper triangular matrix
  $Diag = \{L_P \cup L_N\}$, diagonal of the triangular matrix
  /* Find the optimal projection between subclasses */
  if *Projection is TRUE* **then**
    $Lower = J^*(S_1)$, lower triangular matrix, see Eq. (12)
  else
    $Lower = \text{IR}(S_1)$, lower triangular matrix, see Eq. (12)
  $IRO = \{Upper \cup Lower \cup Diag\}$, Imbalance Ratio Overlap matrix
  /* Noisy label analysis */
  foreach $X_i \in S$ **do**
    Find *class.neighbors*, class with largest num. of instances among the k-nearest neighbors
    if $Label_i = class.neighbors$ **then**
      x_i has the correct label
    else
      x_i is noise
      if x_i *is located between two subclasses with* $J^* \geq sp.th$ **then**
        $x_i \in NL$, Noisy Label set
      else
        $x_i \in NO$, Noise Overlap set
  /* Dispersion analysis: DA */
  foreach *Class* $\in S$ **do**
    Estimate *mean* and *std* to the *median* per subclass by using $dist$
    Execute the test for homogeneity of multivariate dispersion
  Diagnostic report = *IRO*, *NL*, *NO*, and *DA*
  Let $S_2 = S_1$ + *Diagnostic report*
  Return S_2

5. PROBABILISTIC SAMPLING TECHNIQUE

The purpose of PSATE (**P**robabilistic **SA**mpling **TE**chnique) is to build treatments based on diagnostic tests and its primary focus is reversing the degradation effects due class imbalance between-classes and within-classes (small-disjuncts). Furthermore, PSATE includes a data cleaning procedure of removing instances with wrong label based on 3-nearest neighbor rule. The novelty of the cleaning process lies with the nature of the noise. If a noisy instance is detected between two well-separated subclasses, then it is removed. Otherwise, that noisy instance is retained and associated with overlapping effects instead of a mislabeling causes. The probabilistic component in this technique is due not only to use the GMM detected subclasses, but also to probabilistic parameters that allow to restructure the training set through new synthetic entries, label reallocation, and observations filtering. The early PSATE version proposes remediation for class imbalance and small-disjuncts, which have direct effect on the sparseness problem. In a nutshell, the aim of PSATE is generate a better version of the original training set, which seeks to facilitate the training stage of classification algorithms and reduce misclassification.

5.1 PSATE Components

PSATE consists of three main components, which have a direct connection with class imbalance and small-disjuncts problems discussed in Chapter 4. Those components are connected through a workflow where each component provides information to the next one during the sampling process. Figure 5.1 illustrates the sequence of PSATE, where diagnostic process outputs (subclasses, IRO matrix, and noise instances) and sampling parameters are the inputs in the technique. Noise handling is the first component, which can either removing the noisy instances or change their labels (it is set by the user). Both strategies focus on noise cases between two well-separated subclasses because such case represents in a better way the causes of labeling errors.

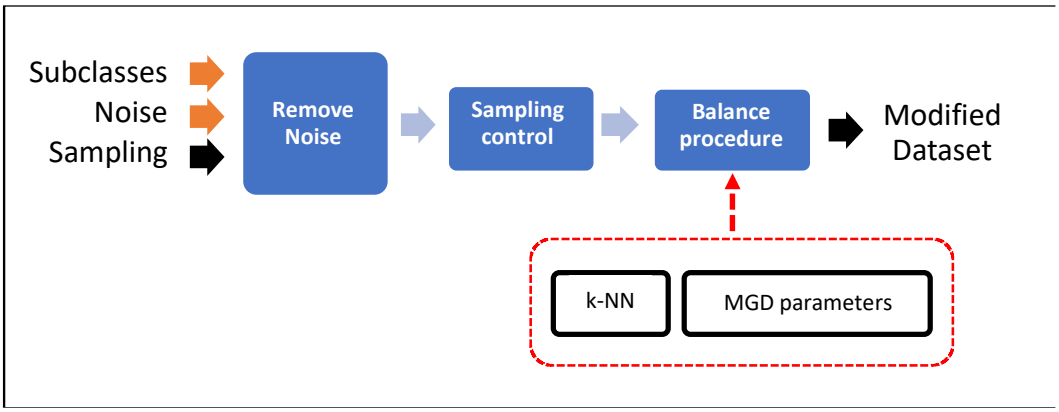


Figure 5.1 PSATE workflow.
The diagnostic process outputs (subclasses, IRO matrix, and noise instances) and sampling parameters are the inputs and the new training dataset (Modified Dataset) is the output.

Then, the "Sampling control" component estimates the number of instances to be random removed or generate synthetically per subclass seeking balance between classes and within classes. Lastly, the "Balance procedure" component applies undersampling and synthetic data generation to each subclass by separated. The synthetic data generation can be conducted by the k-nearest neighbor algorithm or Multivariate Gaussian Distribution (MGD) parameters. Finally, the PSATE flow output is a new version of the training set. More details about these components will be discussed in the following section.

5.2 Case in point Artificial Domain

Continuing the same example from Chapter 4, information from the IRO matrix, noise per subclass, and the sampling parameters are the inputs for PSATE. Since the artificial dataset does not show noisy instances between well-separated subclasses (Refer column "Noise Label" from Table 4.4 (b)), noise cleaning or label reallocation processes are not required.

5.2.1 Sampling control

PSATE seeks to reach the balance between classes changing the subclasses distribution regardless of the origin subclasses. It means the technique attends to solve the problem of imbalance class and small-disjuncts in a novel way, focus on the fact that these problems could be present either in the Positive and Negative class. For instance, the artificial dataset shows 404

instances (82.4%) for the Negative class, which are distributed in three subclasses. On the other hand, the Positive class has 86 instances (17%), which are distributed in two subclasses. Therefore, the $IR = (404/86) = 4.7$. The balance level achieved between classes not only depends on the original distribution of the subclasses, but also the sampling parameters set by the user.

The sampling parameter *Over* controls the number of synthetic instances for the Positive class $New(P)$, which is determined by Equation (1).

$$New(P) = \frac{Size(P) \times Over}{100\%} \qquad (1)$$

where $Size(P)$ is the number of instances in the Positive class. The final size of the Positive class $Final(P)$ is given by $Final(P) = Size(P) + New(P)$. For example, if $Over = 300\%$ and $Size(P) = 86$. Then, $New(P) = 258$ and $Final(P) = 344$. On the other hand, the final size of the Negative class $Final(N)$ is determined by Equation (2).

$$Final(N) = \frac{New(P) \times Under}{100\%} \qquad (2)$$

For example, if $Under = 130$. Then, $Final(N) = 335$. Based on the new size of the classes the new imbalance ratio $IR_{new} = Final(N)/Final(P)$, which is 1.03. The last procedure of the sampling control component calculates the final size of each subclass as a function of the final size of each class and their subclasses.

This procedure defines the sampling strategy to be adopted for each subclass, which is based on the principle to reach balance within classes. For instance, some subclasses may increase in size by adding synthetic instances. On the other hand, other subclasses may decrease in size by removing instances randomly. Equation (3) and Equation (4) estimate the final size of subclasses for the Positive and Negative class respectively. Pseudo-code for the PSATE algorithm is shown in section 5.3.

$$Final(P.sub) = floor\left(\frac{Final(P)}{K_P}\right) \qquad (3)$$

$$Final(N.sub) = floor\left(\frac{Final(N)}{K_N}\right) \qquad (4)$$

where K_P and K_N are the number of subclasses within the Positive and Negative class respectively. For the artificial dataset under study, $Final(P.sub) = floor(344/2) = 172$ and $Final(N.sub) = floor(355/3) = 111$. That means, subclasses which belong the Positive class will have a final size of 172 instances and subclasses belonging to the Negative class will have a final size of 111. Table 5.1 shows the sampling strategy for each subclass.

Table 5.1 Sampling strategy per subclass for the artificial training set. The column Instances shows the original number of instances per subclass, the column Sampling refers to the number of instances to be removed or generated per subclass. The column Final size shows the number of instances per subclass defined by the sampling control stage.

Sampling strategy			
	Instances	**Sampling**	**Final size**
N-01	189	-78	111
N-02	152	-41	111
N-03	63	+48	111
P-01	36	+136	172
P-02	50	+122	172

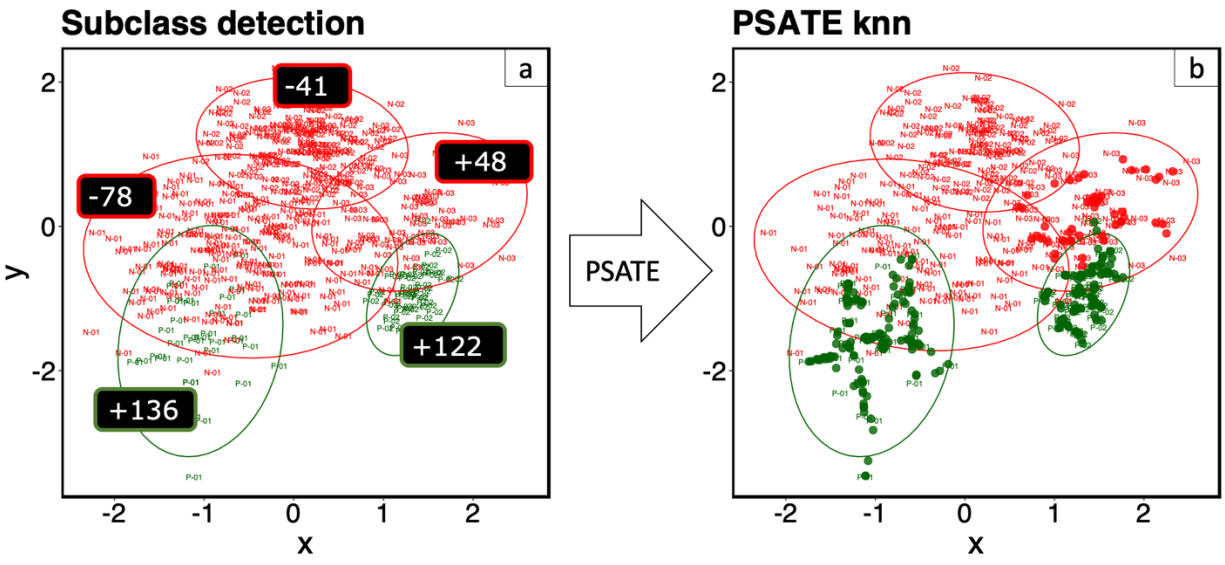


Figure 5.2 PSATE results for the the artificial training set. a) Subclasses detection and sampling control for balance between and within classes. b) Balance procedure based on k-NN algorithm. Dots are synthetic instances generated for specific subclasses based on the sampling strategy.

5.2.2 Balance procedure

Once sampling strategy per subclass has been made known, the procedure for removing instances is conducted by classical undersampling. On the other hand, synthetic samples generation can be performed by two methods. The First one is called “PSATE k-NN” because is based on the k-NN rule (k-nearest neighbors), which generates synthetic instances by linearly interpolating selected instances and their neighbors. Equation 5 describes the generation of a synthetic sample.

$$\vec{x} = \vec{a} + w \times \left(\vec{b} - \vec{a}\right) \qquad (5)$$

where, $\vec{x}$ is the synthetic sample, $\vec{a}$ is a real instance and $\vec{b}$ one on its k-nearest neighbors selected randomly. Finally, w is a random weight in [0,1]. The second method is called “PSATE Gauss” because it uses the Multivariate Gaussian Distribution parameters (mean vector and covariance matrix) associated with each subclass, which were estimated by the GMM at the diagnostic framework. More details about Multivariate Gaussian Distribution parameters can be found at Appendix A. Figure 5.2 shows the implementation of PSATE k-NN using the sampling values defined above section. Note that this sampling strategy not only cover the global imbalance between classes, but also lack of instances in the subclasses (small-disjuncts) for the Positive and Negative class.

5.2.3 Sampling strategies comparison

As previously mentioned in the Chapter 3, exist several techniques for handling imbalance data in supervised learning. Most of them are extension of the well-known SMOTE algorithm. For example: ADASYN, DBSMOTE, and B-SMOTE. Figure 5.3 illustrates sampling results over the artificial dataset by using such techniques and the two versions of PSATE. All sampling techniques use the same sampling parameter (Refer Table B. 2). In general, the PSATE approach seeks to approximate the true data distribution for the Positive and Negative class in order to generate synthetic samples. On the other hand, PSATE competitors focus primarily on the Positive class and its local information. In particular, the major difference between PSATE and its competitors is that it effectively combats the imbalance between classes and within classes, extending the synthetic samples generation to the Negative class. For instance, only in cases (a) and (b) from Figure 5.3 synthetic data generation for one of the subclasses (small-disjunct) in the Negative class

is performed. Another important benefit to consider about PSATE is to avoid punishment over small-disjuncts (subclasses) in the Negative class by removing instances from larger size subclasses.

PSATE k-NN and SMOTE have similar synthetic sample pattern generation because both techniques implement k-NN, which generate synthetic samples along the line segmented that join Positive instances (Refer cases from Figure 5.3 (a) and Figure 5.3 (c)). On the other hand, PSATE Gauss generates synthetic samples randomly through the MGD probabilistic parameters that describe each subclass. In a nutshell, PSATE Gauss effectively combats the sparseness in subclasses. ADASYN is characterized by the fact that the synthetic samples are controlled by a density distribution based on k-NN. Therefore, it reduces linear segments of synthetic samples recurrent in SMOTE. The DBSCAN pattern is characterized by the generation of synthetic samples along a shortest path from each Positive instance to its centroid. For this reason, DBSCAN pattern looks similar to an asterisk shape. Finally, B-SMOTE concentrates the synthetic sample generation in the neighborhood of the borders, which are the most problematic region in the presence of overlapping. However, this strategy does not contribute to minimize the sparseness problem.

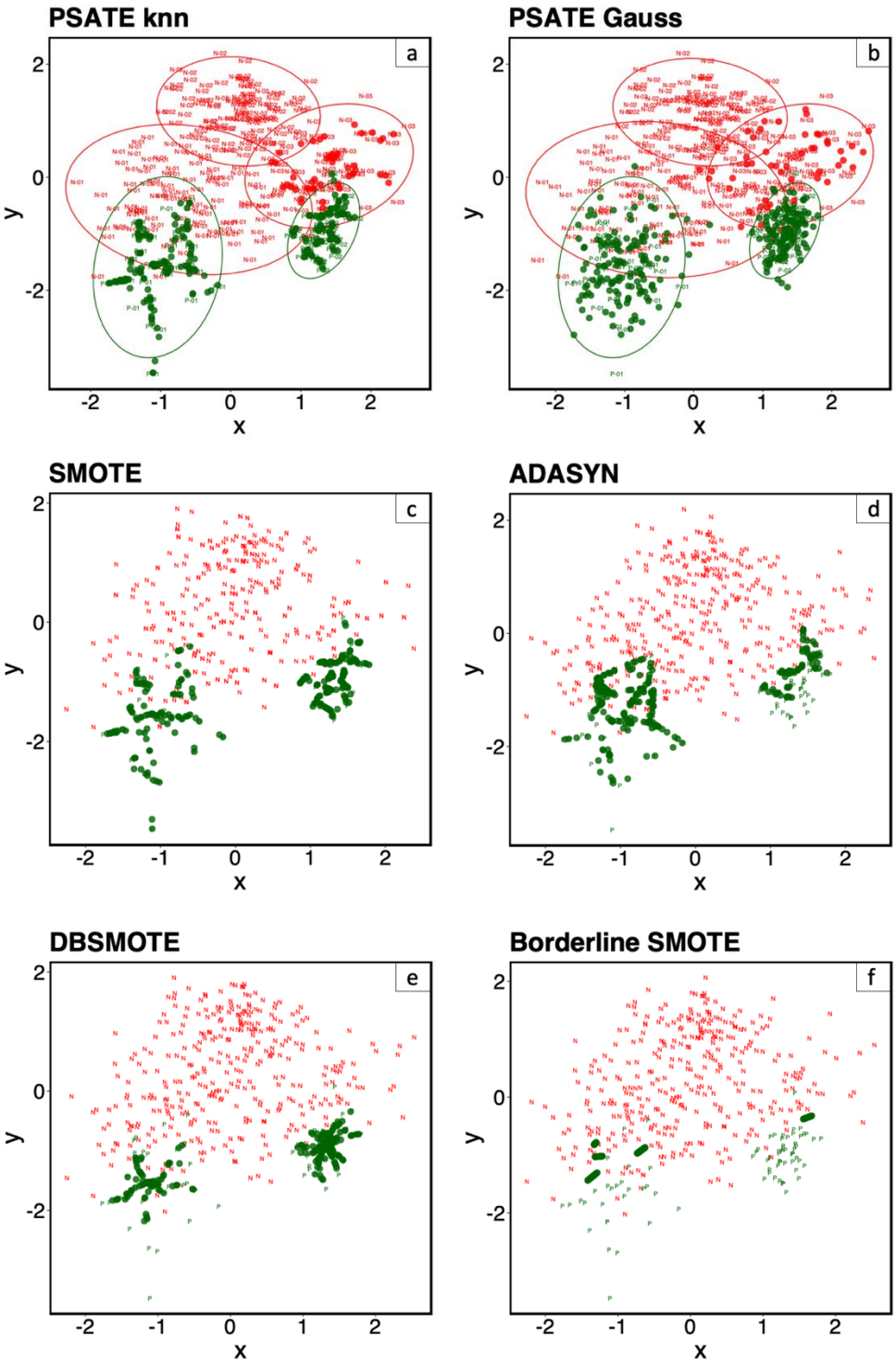


Figure 5.3 Comparison between sampling techniques for the artificial training set. Synthetic instance generation for subclasses (Positive and Negative): PSATE k-NN (a) and PSATE Gauss (b). Synthetic instances generation only for the Positive class: SMOTE (c), ADASYN (d), DBSMOTE (e), and Borderline SMOTE (f)

5.3 PSATE Algorithm

PSATE was develop using R open source software. Pseudo-code for the PSATE components is shown below.

Table 5.2 PSATE algorithm

Algorithm 2: PSATE

Input	:	S	is the training set, $X = \{x_1, x_2, ..., x_{t+1}, x_{t+2}, ...x_M\}$ are the instances of S
		$Label$	is the class membership vector of X_t. Positive (P) or Negative (N)
		$Subclasses$	is a vector that discriminate subclasses for P and N classes
		L_P	vector with the length of each subclass P
		L_N	vector with the length of each subclass N
		$Method$	Synthetic oversampling method: k-NN or MGD
		k	is the number of k-nearest neighbor
		MGD	List with Multivariate Gaussian Distribution parameters for each subclass
		$Noise$	is a vector that discriminate noise between well-separated subclasses
		$Over$	Oversampling
		$Under$	Undersampling
Output	:	$psate.df$	Data frame that contains original and synthetic instances based on sampling parameters Over and Under

PSATE($S, Subclasses, Method, k, MGD, Noise, Over, Under$)
/* Remove noise */
$S^* = (S - Noise)$
$New(P)$ = num. of synthetic instances for Positive class, see Eq. (1)
$Final(N)$ = final size of the Negative class, see Eq. (2)
$Final(P.sub)$ = final size of subclasses for Positive class, see Eq. (3)
$Final(N.sub)$ = final size of subclasses for Negative class, see Eq. (4)
/* Sampling control */
$Sampling(P.sub) = L_P - Final(P.sub)$, vector with the size of the sampling
$Sampling(N.sub) = L_N - Final(N.sub)$, vector with the size of the sampling
/* Balance procedure */
foreach $subclass \in S$ **do**
    Generate synthetic samples based on $Method$, see Eq. (5,6)
    Remove real instances based on $Over$, Build new subclass set
$psate.df$ = collection of all new subclass sets
Return $psate.df$

6. EXPERIMENTAL METHODOLOGY

Previous studies have been suggested that the class imbalance is not an issue by itself, but performance degradation on classifiers is also related to other problems such as overlapping, noisy labels, and small-disjuncts [9], [12], [17], [25], [27], [40], [46]. Although those studies indicate a relationship between such problems, the true correlation and causality between them is not yet well-established. Furthermore, limitations of previous research are based on the use of artificially generated dataset, simple datasets, and limited number of datasets and classifiers. Therefore, their conclusions may hold only for those limited scenarios.

6.1 Experimental Framework

Figure 6.1 illustrates the experimental methodology implemented in the present research, which is composed by three experimental stages. First, stage A seeks to build a Dataset Degradation Profile (DDP) based on the diagnostic model output. This makes possible to know deeper the relationship between degradation problems and their level of criticality. The stage B builds and evaluates several classification algorithms in order to assess the level of concordance between performance metrics and tests classification algorithms robustness in presence of degradation problems. Finally, stage C compares PSATE with other techniques for handling degradation problems by making use of the diagnostic model outputs. This stage mainly seeks to make evident that the success of classification tasks depends on the combination of several factors related to dataset diagnosis, treatment, and classification algorithm.

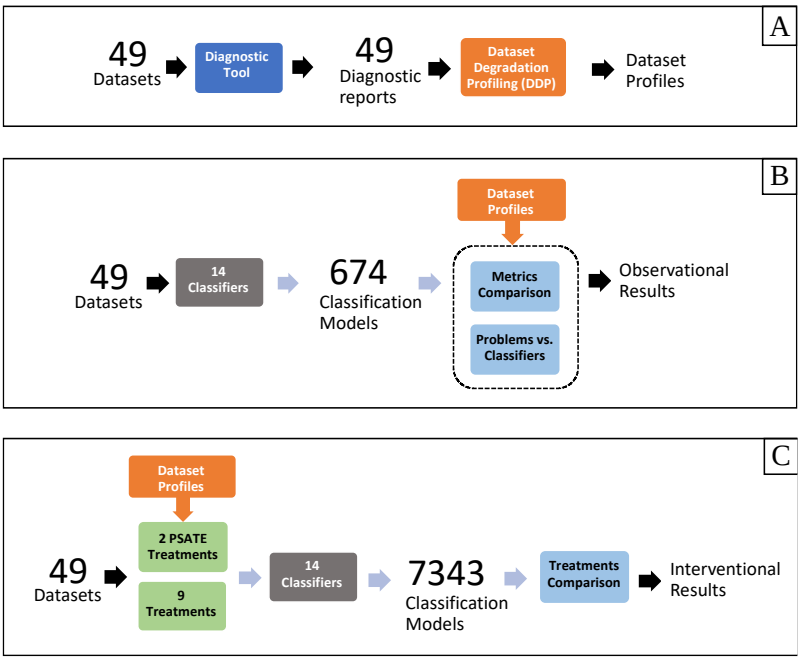


Figure 6.1 Experimental methodology
Stage A: Dataset Degradation Profiling (DDP). Stage B: Classifiers performance.
Stage C: Treatments for degradation problems comparison

The Diagnostic model, PSATE (**P**robabilistic **SA**mpling **TE**chnique) and other treatments (techniques for degradation problems) results were studied empirically by using 49 datasets from real-world domains. Table 6.1 describes main characteristics of that collection, which varies in their number of features (Feat.), size (Instances), missing (Miss.) and duplicated (Dupl.) values, class imbalance (IR), and size of subset for training and testing. First 35 datasets are from the UCI machine learning repository [75], 7 datasets, identified with "+", from R packages, 6 datasets from different repositories, and one new dataset, the "Simulated dataset", generated by the author. Those multiclass datasets marked with "*" were binarized by using the original dataset description. Finally, datasets identified with "°" are subsets of the original datasets. Complete names and references of the datasets can be found at the Appendix B. Experiments from stages B and C are based on 14 classification algorithms, whose parameter values were set based on two criteria: authors recommendations and model tuning using resampling. In the latter case, the "caret" (Classification And Regression Training) package [76]–[79] was used, which is available in R open source software. This package offers a framework for building machine learning models. For instance, it includes several functions for data splitting, pre-processing, feature selection, variable

importance estimation, and model tuning by using resampling. This research used that latter component, which allowed to choose the optimal model across parameters for each classification algorithm. Finally, the experimental results are obtained based on 10-fold cross-validation.

In summary, stage A results come from the 49 datasets analysis. Stage B experimental results stem from every combination of the 49 datasets and 14 classifiers (674 trained models); in short, all classifiers in this stage were trained by using original raw data without the intervention of any remediation technique for handling degradation problems. Table 6.2 shows a brief description of the implementation, which lists functions, packages and tuning parameters for the 14 classification algorithms. The following are the abbreviation names for the classifiers: NB (Naive Bayes), LOGREG (Logistic Regression), LDA (Linear Discriminant Analysis), QDA (Quadratic Discriminant Analysis), C5.0, CART (Classification and Regression Trees), k-NN (k-Nearest Neighbors), MLN (Multilayer Neural Network), SVM-G (Support Vector Machines with Gaussian kernel), SVM-P (Support Vector Machines with Polynomial kernel), RF (Random Forest), SGB (Stochastic Gradient Boosting), ADABOOST (AdaBoost Classification Trees), and RBFN (Radial Basis Function Network).

Finally, stage C experimental results derive from every combination of the 49 datasets, 14 classifiers, and 11 treatments (7343 trained models). Table 6.3 lists the treatments techniques and set parameters.

Table 6.1 Description of datasets

id	Name	Feat.	Instances	Miss.	Dupl.	IR train	IR test	% Train	%Test
1	Satimage *	36	6435	0	0	9.69	8.48	68.92	31.08
2	EEG	14	14980	0	0	1.23	1.23	70.51	29.49
3	Blocks *	10	5473	0	45	45.44	49.63	70.15	29.85
4	ILPD *	9	583	4	8	2.50	2.45	70.40	29.60
5	Glass *	9	214	0	1	1.88	2.60	66.20	33.80
6	QSAR	41	1055	0	1	2.11	1.71	68.41	31.59
7	Ozone	72	2534	687	0	13.76	12.67	71.14	28.86
8	Occupancy	5	17895	0	867	3.21	3.76	42.73	57.27
9	Vertebral	6	310	0	0	2.45	1.53	69.03	30.97
10	Haberman	3	306	0	7	3.02	2.13	68.56	31.44
11	Spam	57	4601	0	238	1.51	1.53	69.20	30.80
12	Gamma	10	19020	0	60	1.86	1.87	70.46	29.54
13	Blood	4	748	0	131	2.56	3.48	62.24	37.76
14	Seismic	11	2584	0	2	13.50	16.23	71.30	28.70
15	Wilt	5	4839	0	14	57.45	1.67	89.64	10.36
16	Abalone *	7	4177	0	0	19.89	24.46	69.52	30.48
17	Audit	8	776	1	26	1.98	1.57	68.49	31.51
18	Wireless *	7	2000	0	0	2.94	3.16	70.45	29.55
19	User-Knowledge *	5	403	0	0	9.75	4.58	64.02	35.98
20	Shuttle *°	9	17500	0	0	4.00	3.84	71.43	28.57
21	Image	18	2310	0	120	1.27	1.43	69.41	30.59
22	Happiness	6	143	0	9	1.08	0.72	58.96	41.04
23	Skin °	3	12500	0	3723	2.38	4.00	57.97	42.03
24	Seeds	7	210	0	0	1.82	2.45	67.14	32.86
25	Musk1	166	476	0	0	1.40	1.08	71.22	28.78
26	CTG *	20	2129	3	8	3.58	3.40	70.30	29.70
27	Sonar	60	208	0	0	1.18	1.06	69.23	30.77
28	Forest *	27	523	0	0	4.35	6.07	37.86	62.14
29	HTRU2	8	17898	0	0	9.90	9.97	70.45	29.55
30	Adult °	6	26281	0	24	3.13	3.23	37.99	62.01
31	Sports	59	1000	0	3	1.86	1.52	67.60	32.40
32	Banknote	4	1372	0	11	1.21	1.28	69.29	30.71
33	Electrical	13	10000	0	0	1.78	1.73	70.43	29.57
34	Wine *	13	178	0	0	2.94	2.20	73.03	26.97
35	Breast	9	699	0	153	1.03	0.49	66.79	33.21
36	nanoHUB	18	14110	0	142	39.63	38.02	70.39	29.61
37	Simulated	2	700	0	0	4.70	5.18	70.00	30.00
38	Weather +	17	266	0	0	4.52	4.55	68.64	31.36
39	Pima1+	7	532	0	0	2.11	1.76	71.99	28.01
40	Ionosphere +	32	351	0	1	1.76	1.96	79.71	20.29
41	Pima2 +	8	768	0	0	2.01	1.58	69.40	30.60
42	Ringnorm	20	7400	0	0	1.02	1.02	70.31	29.69
43	College +	17	777	0	0	3.05	2.03	69.24	30.76
44	Iris +*	4	150	0	0	1.74	2.56	62.00	38.00
45	MDRR +	342	528	0	2	1.36	1.16	71.67	28.33
46	Mammo	6	11183	0	2343	28.56	46.58	62.86	37.14
47	Phoneme	5	5404	0	32	2.39	2.48	70.16	29.84
48	Fraud °	30	10492	0	13	21.94	18.69	70.49	29.51
49	SDSS *	15	10000	0	0	11.28	9.74	69.39	30.61

Table 6.2 List of classification algorithms with tuning parameters.

Abbreviation names for classifiers: NB (Naive Bayes), LOGREG (Logistic Regression), LDA (Linear Discriminant Analysis), QDA (Quadratic Discriminant Analysis), C5.0, CART (Classification and Regression Trees), k-NN (k-Nearest Neighbors), MLN (Multilayer Neural Network), SVM-G (Support Vector Machines with Gaussian kernel), SVM-P (Support Vector Machines with Polynomial kernel), RF (Random Forest), SGB (Stochastic Gradient Boosting), ADABOOST (AdaBoost Classification Trees), and RBFN (Radial Basis Function Network)

Classifier	Function	Package	Tuning parameters
NB	nb	klaR	fL (Laplace Correction) usekernel (Distribution Type) adjust (Bandwidth Adjustment)
LOGREG	glm	R Stats	NA
LDA	lda	MASS	NA
QDA	qda	MASS	NA
C5.0	C50	C50, plyr	trials = c(1,5,10,15,20) winnow = c(T,F), model = tree
CART	rpart	rpart	method = clas, cp = 0, xval = 10
k-NN	knn	class	k (#Neighbors)
MLN	nnet	nnet	size (# Hidden Units) decay (Weight Decay) bag (Baggaing)
SVM-G	ksvm	kernlab	type = C-svc, kernel= rbfdot kpar = automatic, C=1
SVM-P	ksvm	kernlab	type = C-svc, kernel= polydot kpar = automatic, C=1
RF	rf	randomForest	mtry (# Randomly Selected Predictors)
SGB	gbm	gbm, plyr	n.trees (# Boosting Iterations) interaction.depth (Max Tree Depth) shrinkage (Shrinkage) n.minobsinnode (Min. Terminal Node Size)
ADABOOST	adaboost	fastAdaboost	nIter (#Trees), method
RBFN	rbfDDA	RSNNS	negativeThreshold (Activation Limit for Conflicting Classes)

Table 6.3 List of treatments with tuning parameters.
Abbreviation names for treatments: Raw (original set), Random (resampling), SMOTE (Synthetic Minority Oversampling Technique), B-SMOTE (Borderline-SMOTE), DBSMOTE (Density-base SMOTE), ADASYN (Adaptative Synthetic Sampling Approach for Imbalanced Learning), ENN (Edited Nearest Neighbors), NCL (Neighborhood Cleaning Rule NCL), OSS (One-Side Selection), and PSATE (**P**robabilistic **SA**mpling **TE**chnique)
Treatments for handling degradation problems

Name	Function	Package	Tuning parameters
Raw	-	-	-
Random	-	-	Over (Oversampling parameter) Under (Undersampling parameter)
SMOTE	SMOTE	DMwR	perc.over, perc.under k = 3
B-SMOTE	BLSMOTE	smotefamily	K = 3 (# Neighbors sampling process) C = 3 (# Neighbors safe-level process) dupSize (# Synthetic Positive instances) method = "type2"
DBSMOTE	DBSMOTE	smotefamily	dupSize (# Synthetic Positive instances)
ADASYN	ADASYN	smotefamily	K = 3 (# Neighbors sampling process)
ENN	ubENN	unbalanced	K = 3 (# Neighbors)
NCL	ubNCL	unbalanced	K = 3 (# Neighbors)
OSS	ubOSS	unbalanced	-
PSATE	-	-	Over (Oversampling parameter) Under (Undersampling parameter) k = 3 (# Neighbors sampling process) Subclasses Noisy labels MGD parameters method = (k-NN or MGD)

6.2 Stage A: Dataset profiling

As previously mentioned, different studies suggest class imbalance is not completely responsible for classifiers degradation and argue that the problem of small-disjuncts is even more severe [12], [22]–[26]. However, the small-disjunct concept itself can be interpreted as the imbalance problem between-class and within-class. Furthermore, one of the reasons why small-disjuncts show higher misclassification than large disjuncts is due to class imbalance [80], [81]. On the other hand, other studies state that misclassification is not solely caused by class imbalance, but also associated with the degree of overlapping among classes [9], [12], [27]. Keeping the above in mind, the diagnostic tool integrates those elements by building an output called "IRO matrix" (Refer Chapter 4), which creates a "diagnosis" or profile of the target dataset. In the following, a comparative example is presented by using Ozone and Wine datasets (Refer Table 6.1) to illustrate the benefits derived from the diagnostic model. Initially, Ozone dataset looks more complex since it has higher dimensionality (number of features) and class imbalance (IR). On the other hand, the low amount of data may be the major issue for the Wine dataset. Figure 6.2 compares the IRO matrix for those datasets. The diagonal of each matrix contains the number of instances per subclass (white), which allows for identifying small-disjuncts in both classes. The upper triangular part (blue) shows the IR between subclasses. Finally, the lower triangular part (red) of the matrix shows the separation index (degree of overlapping) between subclasses. The intensity of the colors helps to identify how critical is the problem based on each metric. It is important to mention that overlap between subclasses from the same class is not considered a problem. Overall, Wine has more subclasses (disjuncts), less amount of data, higher IR between subclasses, and lack of overlap. In contrast, the most relevant problem of Ozone is related to several regions with high overlap and some subclasses with few instances. Moreover, the segment chart below of each IRO matrix shows the performance of 14 classifiers for each dataset by using the G-mean performance metric. There is an evident low performance for most of the classifiers in the Ozone dataset case, suggesting a need to implement remediation treatments. Furthermore, even though Wine has more small-disjuncts and higher IR between subclasses, this dataset does not require any remediation treatment. Finally, Wine may be considered as no challenging dataset for classification task given lack of overlapping between subclasses from different classes. This latter statement will be validated in the next section.

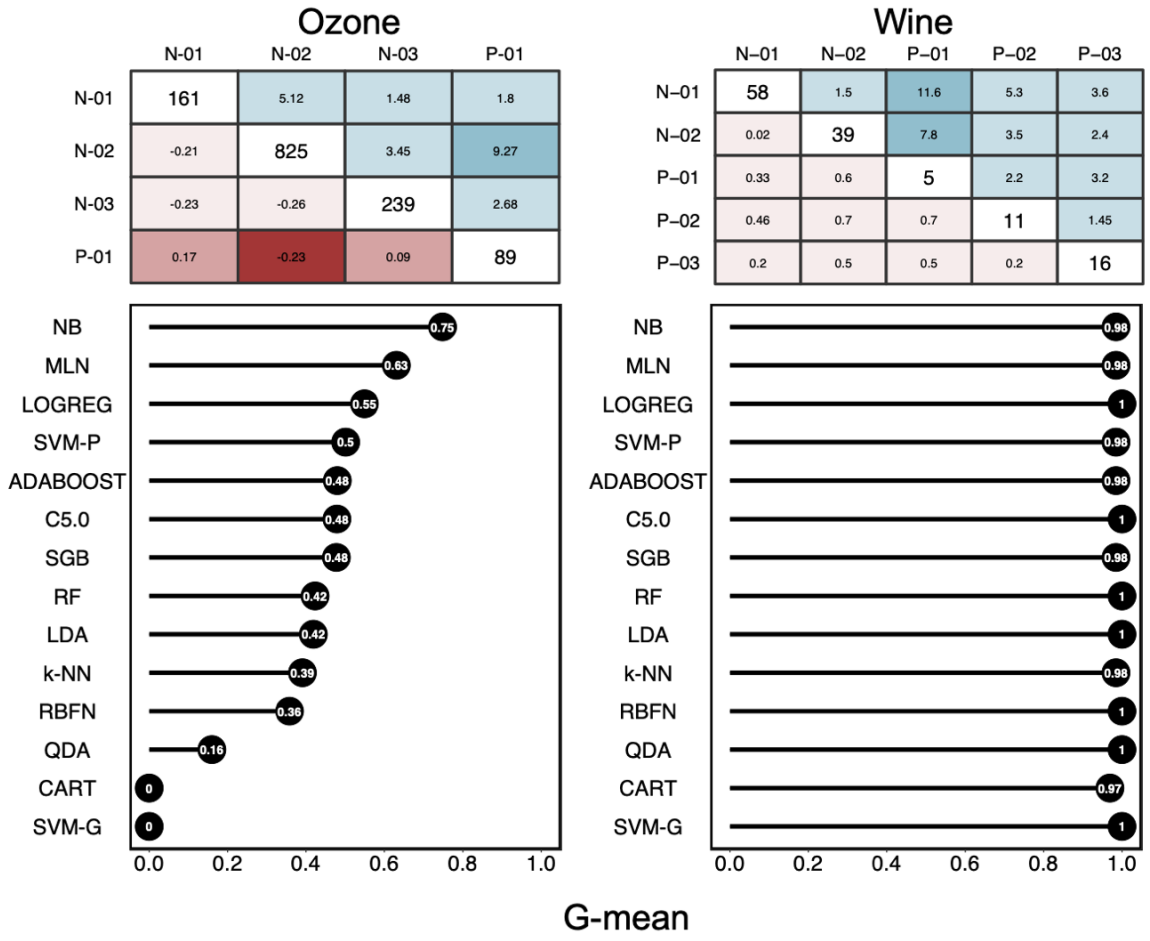


Figure 6.2 IRO matrix comparison and performance for multiple classifiers. The diagonal contains the number of instances per subclass (white), which allows for identifying small-disjuncts in both classes. The upper triangular part (blue) shows the IR between subclasses. Finally, the lower triangular part (red) of the matrix shows the separation index (degree of overlapping) between subclasses. The intensity of the colors identifies how critical is the problem based on each metric. The segment chart below of each IRO matrix shows the performance of 14 classifiers for each dataset by using the G-mean performance metric

In conclusion, the aim of the DDP (Dataset Degradation Profile) is to provide useful information for selecting or designing a tailored treatment. Dataset profiles are categorized according to levels of criticality. The DDP results are categorized as follows:

- **IR**: If IR is less or equal than 10 then level is "Low", otherwise level is "High"
- **Disjuncts**: If the number of Disjuncts (subclasses) is less or equal than 10 then level is "Low", otherwise level is "High"
- **Overlap**: If NOR (Noise Overlap Ratio) is less or equal to 0.1 then level is "Low", otherwise level is "High"

Table 6.4 shows a contingency table that summarized the results of the DDP process for the 49 datasets. The table describes the relationship between three degradation problems (IR, Disjuncts, and Overlap) in terms of frequencies of their levels of criticality (Low and High). The level “Unknown" from the Overlap problem refers to datasets for which it was not possible to find an optimal projection for estimating the *separation index*.

Table 6.4 Contingency table of dataset profiles.
Cross levels show the number of datasets for each criticality combination

			Overlap			
		IR	Low	High	Unknown	Total
Disjuncts	Low	Low	9	12	4	25
		High	1	-	-	1
	High	Low	7	4	4	15
		High	6	-	2	8
		Total	23	16	10	**49**

6.3 Stage B: Performance metrics selection

Performance metrics are used to evaluate how well classification algorithms conduct the discrimination task between classes. Several studies point out limitations of the global accuracy as a metric of performance in class imbalance situation [82]–[87] . In a nutshell, the global accuracy metric is heavily biased to favor the Negative class and may hide poor performance for the Positive class. Furthermore, Positive class is usually the class of interests in real world applications. Typically, the aim is to learn in detail the Positive class distribution in order to recognize its members as accurately as possible. For example, fraud detection is a well-known case in point, where the number of fraud transaction is much smaller than the legitimate ones and the goal is detect such transaction immediately in order to avoid financial losses. In conclusion, the main goal of classifiers is not only improving the accuracy for the Positive class, but also avoiding decreasing the accuracy for the Negative class during the process. Continuing with the same fraud example, financial institutions want to increase the detection of the number of "True Positive" cases (real fraud transactions) and avoid to increase the number of "False Positive" cases (legitime

transactions classified as fraud) because the latter means higher administrative costs related to respond to the false fraud alerts.

Practitioners have opted to use the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) [15] as metric of performance for supervised learning, since it is considered more appropriate for imbalance learning [88]–[90]. However, recent studies have raised that the AUC may provide an optimistic performance evaluation [16], [91] and draw wrong conclusions for imbalance datasets. Thereby, the first crucial contribution of this study in this context is to clarify how convenient is to use the AUC as performance metric for datasets in the presence of degradation problems. If the AUC metric is shown to be optimistic and misleading, it will be not only inconvenient for classifiers performance evaluation, but also for testing the effectiveness of the remediation techniques for degradation problems such as SMOTE, OSS, ENN, and so on. For instance, Figure 6.3 compare four performance metrics (AUC, Acc-, Acc+, and G-mean) associated with a Support Vector Machine with Gaussian kernel (SVM-G) implementation for Ozone and Wine datasets. As previously mentioned, Wine dataset is not a challenging case for classification. Thereby, these metrics agree in terms of the SVM-G performance for Wine dataset. Nevertheless, Ozone dataset shows discrepancy between some metric. Even though the AUC is above 80% and the accuracy of the Negative class (Acc-) is 100%, the accuracy of the Positive class (Acc+) is 0%. This could imply that the AUC is an inaccurate metric in relation to certain cases. For example, the low Acc+ may be associated with high level of global IR (13.76) in the Ozone dataset. In summary, there is one instance in the Positive class per 13.76 instances in the Negative class. Moreover, the separation index shows overlap problems between subclasses (refer IR and Overlap matrix from Figure 6.2). On the other hand, the G-mean is more conservative and intuitive metric because shows the balance between the accuracy from both classes. Since the Acc+ for the Ozone dataset is zero, the G-mean punishes the classifier performance because it is unable to classify one of classes correctly. Technically, the G-mean (also called G-measure) is the geometric mean of the Positive class accuracy (Acc+) and the Negative class accuracy (Acc-), which return a value between 0 and 1. Technical details of the evaluation metrics implemented in the present research such as AUC, F-Measure (F1-score), G-measure (G-mean), sensitivity, specificity, precision and other metrics can be found at the Appendix B. This example makes evident limitations of the AUC as performance metric for complex datasets. Since the ROC graphs are drawn by using True Positive Rates (TPR) and False Positive Rates (FPR), the relative

imbalance between classes directly affects the FPR, which tends to zero and generates misleading estimations of the AUC measure.

In general, most classifiers are not designed with the assumption of dealing with class imbalance [92] , and Figure 6.3 is a clear example of this by using Acc+ and the G-mean metrics, which are performance measures more conservative than the AUC. The SVM-G showed a low performance in term of Acc+ for the Ozone dataset, which is not surprising due to the high imbalance ratio. Figure 6.4 clearly shows the disagreement between AUC and G-mean metrics for Ozone dataset across 14 classification algorithms, where the AUC for all cases always overpass the G-mean performance.

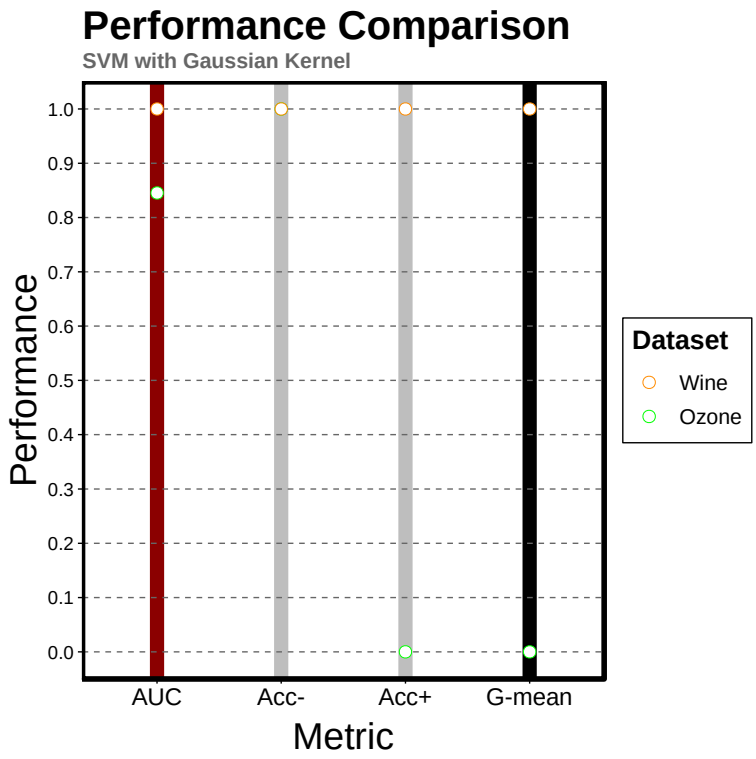


Figure 6.3 Performance metrics comparison for two datasets by using SVM-G classifier. The AUC and G-mean are consistent with the values of Acc- (Accuracy Negative class) and Acc+ (Accuracy Positive class) for the wine dataset. However, the AUC is an extremely optimistic metric for the Ozone dataset (85%) despite its low Acc+ performance (0%). On the other hand, the G-mean is more realistic performance metric based on the levels of misclassification

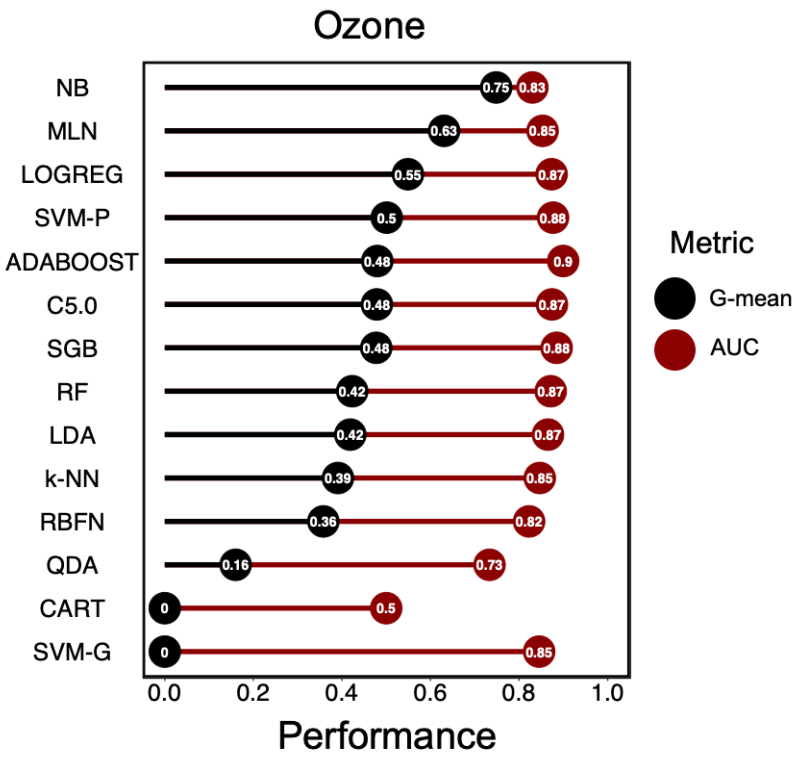


Figure 6.4 Performance metric comparison for Ozone dataset.
AUC and G-mean metrics for Ozone dataset across 14 classification algorithms, where the AUC for all classifiers get higher performance values (more optimistic) than the G-mean.

Another important finding related to the Wine dataset is that for well-separated subclasses the performance metric did not decreased even with high levels of imbalance ratio between subclasses. Thereby, the present experimental stage has two main purposes: the first one seeks to generalize the impropriety use of the AUC as performance metric for datasets under degradation problems. The second one refers to establish solid causality with statistical evidence between degradation problems and classifiers performance by using real-world datasets.

Figure 6.5 compares the distribution of the AUC and G-mean metrics for the 49 datasets across the 14 classification algorithms. AUC boxplot shows a lower variability and its values are closer to the high levels of prediction power. Moreover, AUC outliers below the lower quartile show better performance than the outliers from the G-mean. Finally, the measures of central tendency such as the median and the mean have larger values in the AUC than the G-mean distribution. Additionally, this figure shows the result of the paired samples Wilcoxon test [93]–[95], which is a nonparametric version of the paired t-test. The null hypothesis states that the AUC has values of prediction power equal or lower than the G-mean values. In contrast, the alternative hypothesis

claims that the AUC has values of prediction power higher than the G-mean values. As the *p-value* turns out to be 5.06e-88, and is less than the 0.05 significance level, it is possible to reject the null hypothesis. Therefore, for the experimental design defined in the present research, the Wilcoxon for paired samples test confirms the risky and inconvenience of using the AUC.

Table 6.5 illustrates the categorization in two levels of three degradation problems across the 49 datasets under study. This table is sub-table from the contingency table of dataset profiles mentioned from the previous section. Complete information can be found at Table 6.4. A fair question could be raised as to if the AUC limitation is the same through levels of criticality of the degradation problems.

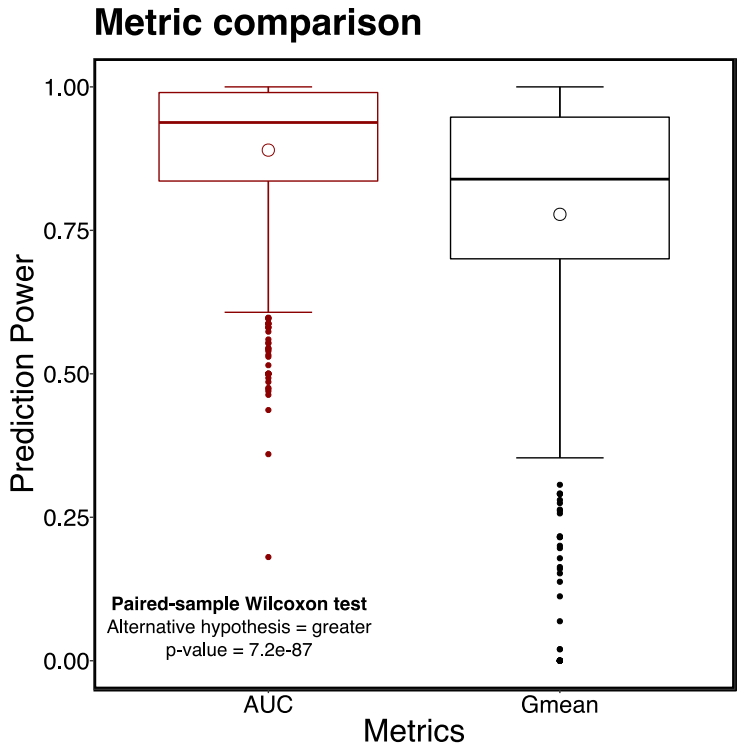


Figure 6.5 Boxplots for AUC and G-mean performance metrics.
It compares the distribution of the AUC and G-mean metrics for the 49 datasets across the 14 classification algorithms. The AUC shows noticeably more optimistic prediction power than the G-mean. Moreover, the paired samples Wilcoxon test rejects the hypothesis that states the AUC has values of prediction power equal or lower than the G-mean values.

Table 6.5 Number of datasets per criticality levels

Level of criticality

	Degradation problems		
Level	**IR**	**Disjuncts**	**Overlap**
Low	40	23	23
High	9	26	16

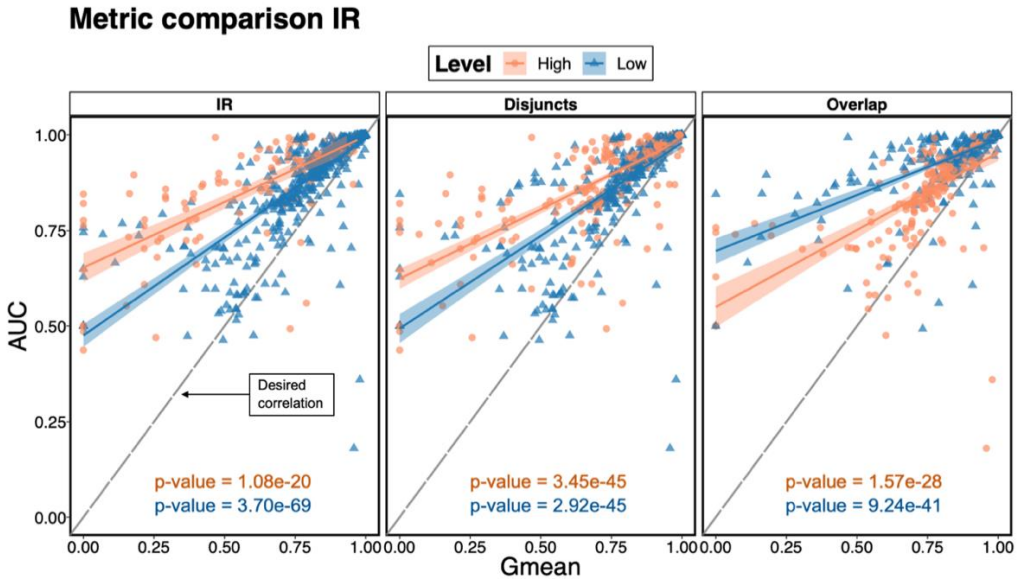


Figure 6.6 AUC and G-mean comparison across criticality levels of degradation problems. It compares the scatter between AUC and G-mean by levels of criticality of degradation problems for the 49 datasets across the 14 classification algorithms. The ideal scenario would be a high linear positive correlation between values for both performance metrics, even for the low and high values. In a nutshell, if the dots do not lie close to the 45-degree line, exist discrepancy between both metrics. Moreover, the paired samples Wilcoxon test rejects the hypothesis that states the AUC has values of prediction power equal or lower than the G-mean values

Figure 6.6 answers the above question by comparing the scatter between AUC and G-mean for levels of criticality of each degradation problem. The ideal scenario would be a high linear positive correlation between values for both performance metrics, even for the low and high values. In a nutshell, if the dots do not lie close to the 45-degree line, exist discrepancy between both metrics. It is evident that the AUC limitation persists through the different levels of criticality; however, for high levels of IR the AUC is more optimistic due to the low values of the FPR. For the next degradation problem, the dispersion of the disjunct levels does not show significant difference. Finally, the overlap problem shows an interesting behavior through its levels. High levels of

overlap make harder to learn the distribution for both classes. Then, the FPR is increased and the TPR is reduced, which decreases the AUC values. On the other hand, low levels of overlap avoid decreasing the "True Negative" value and follow a similar dispersion to the high level of IR.

Similarly, the paired samples Wilcoxon test was conducted for each level of criticality by separate. All *p-values* turns out to be less than the 0.05 significance level, it is possible to reject the null hypothesis. Therefore, for the experimental design defined in the present research, the Wilcoxon for paired samples test confirms the risky and inconvenience of using the AUC; even for different criticality levels of degradation problems.

6.4 Stage B: Classifiers robustness in the presence of degradation problems

Once the issue related to the most convenient performance metric is solved, the next aim of this research is to test classification algorithms robustness in the presence of degradation problems. The following results correspond to the same experimental design defined in the previous section. Basically, this section seeks to validate three recurrent statements in the literature, by incorporating a considerable real-world dataset, several classification algorithms, unbiased performance metrics, and statistical validation.

The three **statements** to valid are:

S1. The more class imbalance, the bigger classifier performance degradation.

S2. The greater number of disjuncts, the bigger classifier performance degradation.

S3. The larger overlap regions, the bigger classifier performance degradation.

Table 6.6 illustrates the results of an alternative version of the Wilcoxon signed-rank test, which is used to compare two related samples. In this example, results derive from the 49 datasets and SVM-G classifier by making use of three performance metrics (G-mean, F1-Score, and AUC). Basically, two tests were implemented in order to validate if exist statistical differences between the performance metric distribution across the levels of criticality for each degradation problem. In other words, those tests assess the truthfulness of the three above statements. Test A compares if the performance metric distributions between levels are equal and Test B compares if the performance metric distribution of the datasets with high levels of criticality have better prediction power than the performance metric distribution of the datasets with low levels of the criticality. Both tests have a 0.05 significance level. The *p-values* shown in the table are used to reject or not

reject the null hypothesis. However, an easier notation combines results from both statistical tests. Symbol "✓" means that exist statistical evidence to accept the **statement** on the other hand, symbol "✗" implies that there is no statistical evidence to accept the **statement**. Therefore, it is not possible to reject that the performance metric distributions across the levels are different.

Analyzing the outputs from Table 6.6, it is possible to state the following conclusions for the classifier SVM-G under the experimental framework defined in this research:

1. The more class imbalance, the bigger classifier performance degradation for metrics G-mean and F1-Score. On the other hand, AUC metric is not affected by the imbalance problem since it is an optimistic and misleading performance metric.
2. It is not true that the greater number of disjuncts, the bigger classifier performance degradation; however, disjuncts detection allows to know the imbalance ratio within classes [26], [80].
3. It is true that the larger overlap regions, the bigger classifier performance degradation.

Table 6.6 p-values of the Wilcoxon signed-rank tests for degradation problems.
Test A compares if the performance metric distributions between levels are equal and Test B compares if the performance metric distribution of the datasets with high levels of criticality have better prediction power than the performance metric distribution of the datasets with low levels of the criticality
p-values Wilconox test for SVM-G

	Statements for degradation problems (*p-values*)					
Metric	**S1 (IR)**		**S2 (Disjuncts)**		**S3 (Overlap)**	
G-mean						
Test A	0.00812	✓	0.86476	✗	0.07202	✓
Test B	0.00406		0.57548		0.03601	
F1-Score						
Test A	0.03484	✓	0.82594	✗	0.06566	✓
Test B	0.01742		0.41297		0.03283	
AUC						
Test A	0.15913	✗	0.984	✗	0.01767	✓
Test B	0.07956		0.51598		0.00883	

Table 6.7 Wilcoxon test for degradation problems across classifiers by using. The G-mean. The column "Tests" validates the statements about level of criticality and column "% Diff." shows the difference in percentage between the means of criticality levels

Classifier	Statements for degradation problems					
	S1 (IR)		**S2 (Disjuncts)**		**S3 (Overlap)**	
	Tests	**% Diff.**	**Tests**	**% Diff.**	**Tests**	**%Diff.**
NB	✗	0.07	✗	0.06	✓	0.05
LOGREG	✓	0.15	✗	0.03	✓	0.17
LDA	✗	0.13	✗	0.03	✓	0.17
QDA	✗	0.12	✗	-0.01	✗	0.02
C5.0	✓	0.17	✗	-0.05	✓	0.12
CART	✓	0.21	✗	-0.08	✓	0.14
k-NN	✓	0.22	✗	-0.03	✓	0.10
MLN	✗	0.16	✗	0.00	✓	0.14
SVM-G	✓	0.29	✗	-0.01	✓	0.09
SVM-P	✓	0.27	✗	0.05	✓	0.18
RF	✓	0.23	✗	0.01	✓	0.08
SGB	✓	0.18	✗	0.01	✓	0.09
ADABOOST	✓	0.23	✗	0.01	✓	0.08
RBFN	✓	0.24	✗	-0.02	✓	0.08

Table 6.7 shows same statistical analysis for the complete set of classifiers and datasets by using the G-mean as performance metric. The column "Tests" validates the statements about level of criticality (Test A and Test B) and column "% Diff." shows the difference in percentage between the means of the levels of criticality. Some important findings are:

- Ten out of fourteen classifiers present G-mean degradation in the presence of high levels of IR.
- Thirteen out of fourteen classifiers show G-mean degradation in presence of high levels of overlap.
- The G-mean for all classifiers is not affected by high levels of disjuncts.
- QDA classifier does not show statistical difference for the G-mean distribution between high and low levels of IR, disjuncts and overlap.

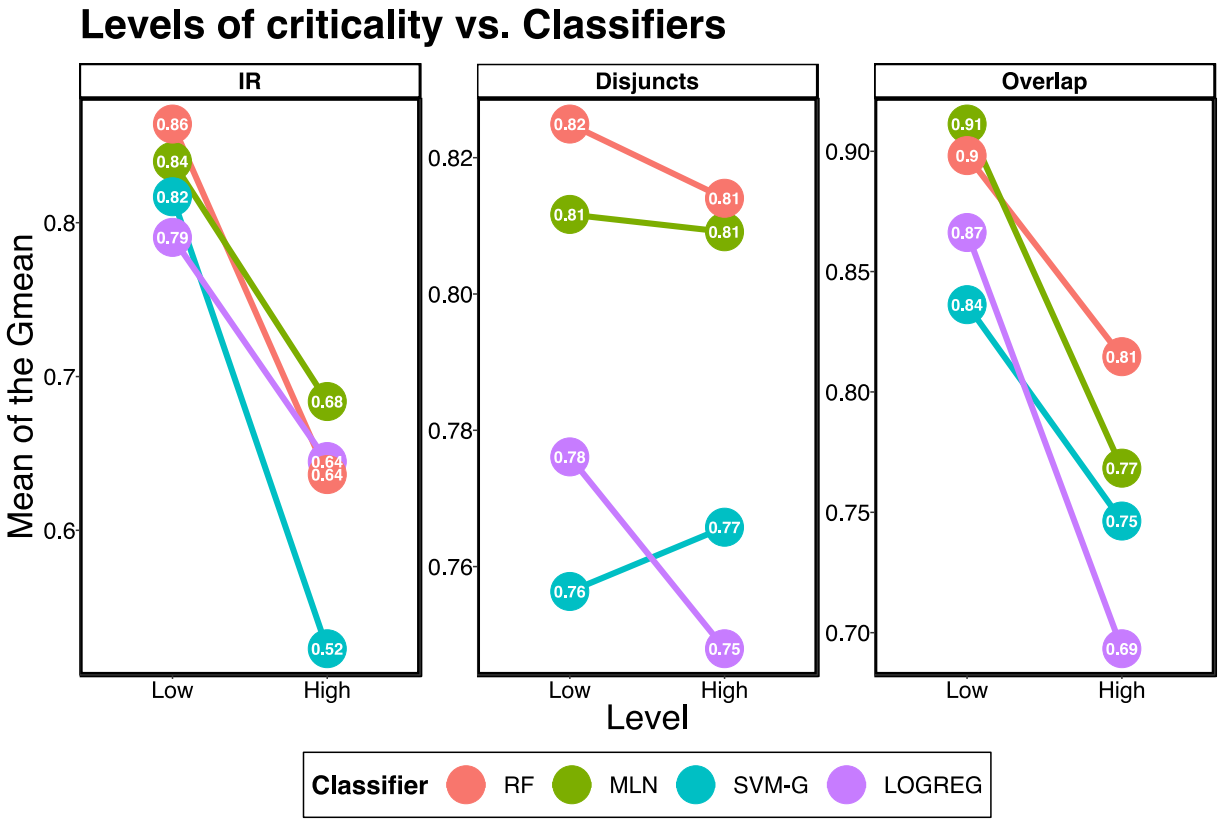


Figure 6.7 G-mean comparison between criticality levels of degradation problems. The mean of the G-mean is estimated across the level of criticality using bootstrap for the following selected classifiers: Random Forest (RF), Multilayer Neural Network (MLN), Supper Vector Machine with Gaussian kernel (SVM-G), and Logistic Regression (LOGREG)

Figure 6.7 illustrates some selected cases from Table 6.7. The mean of the G-mean is estimated across the level of criticality by using bootstrap for the following classifiers: Random Forest (RF), Multilayer Neural Network (MLN), Supper Vector Machine with Gaussian kernel (SVM-G), and Logistic Regression (LOGREG). Even though this chart is a visual and descriptive comparison between means, it is evident that there are no relevant differences between the Disjuncts levels. In contrast, the differences between means for the levels of IR and Overlap are significative.

6.5 Stage C: Statistical identification of the best treatment

To demonstrate that it is not possible to find one treatment that is the best in remediation for all degradation problems and classifiers, a ranking score was computed over three performance metrics (AUC, F1-Score, and G-mean) to each treatment (remediation technique) results across the 73443 trained models (Refer Stage C from Figure 6.1). Therefore, the ranking score for the best performing treatment is 11 and the worst performing treatment is 1. Then, the nonparametric Friedman test [96]–[98] and multiple comparison of treatments were applied to the ranking results. This test not only allows to detect differences between treatments results across multiple experimental related samples, but also implements the post hoc Friedman tests (multiple comparisons between treatments) by using the criterium Fisher's Least Significant Differences (LSD) [94], [95], [99], which groups the treatments according to similarities by using the Compact Letter Display method [100]. The null hypothesis for the Friedman test is that there are no differences between treatments results across multiple experimental attempts. If the null hypothesis is rejected (*p-value* less that the 0.05 significance level), it can be concluded that at least two of the treatments are significantly different from each other. Then, a multiple comparisons analysis can be executed in order to know which of the treatments are significantly different from which other treatments. In particular, the multiple comparisons results are presented by listing the treatments in order of decreasing average score and grouping the treatments that are not significantly different. Thus, the procedure uses a group (identified by a letter) as a union of different treatments that can contain one or more members and the members of these groups are the eleven different treatment in this research (Raw, Random, SMOTE, B-SMOTE, DBSMOTE, ADASYN, ENN, NCL, OSS, PSATE k-NN, and PSATE Gauss). Treatments with the same letter(s) are not statistically different. On the contrary, treatments that are statistically different get different letters. Treatments can have more than one letter to reflect overlap between the groups of treatments. Table 6.8 shows the Friedman test results for experimental definition on Stage C, which includes the 49 datasets. According to the *p-value* column, only the MLN classifier cannot reject the null hypothesis because its *p-value* is slightly higher to 0.05 (borderline result). That means, there are statistical differences between treatments for the remaining classifiers.

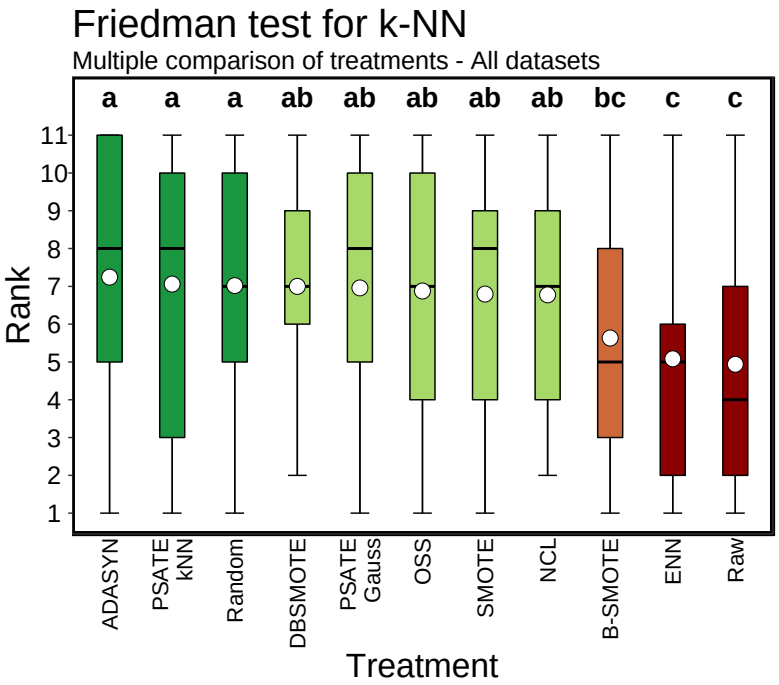
Table 6.8 Friedman test results for all datasets.

According to the *p-value* column, only the MLN (Multilayer Neural Network) classifier cannot reject the null hypothesis because its *p-value* is slightly higher to 0.05 (borderline result). That means, there is statistical differences between treatments for the remaining classifiers.

Means of the G-mean of rank

	Treatment											
Classifier	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSATE Gauss	**p-value**
NB	c	ab	a	c	cd	bc	abc	c	c	abc	d	2.0E-04
LOGREG	e	a	ab	bcd	ab	abc	de	de	e	abc	cd	1.7E-10
LDA	e	a	bc	bc	c	ab	e	de	de	bc	cd	9.2E-13
QDA	d	a	abc	cd	cd	ab	bcd	abc	abcd	abcd	abcd	4.6E-02
C5.0	d	ab	ab	a	abc	a	cd	bcd	abc	ab	ab	1.5E-03
CART	d	a	a	ab	ab	a	cd	ab	bcd	ab	abc	1.6E-04
k-NN	c	a	ab	bc	ab	a	c	ab	ab	a	ab	1.2E-05
MLN	c	ab	ab	ab	abc	a	bc	ab	bc	abc	abc	**5.2E-02**
SVM-G	e	a	ab	bc	cd	ab	e	cd	d	abc	abc	0.0E+00
SVM-P	f	ab	a	bcd	ab	abc	ef	bcd	de	abc	cde	2.0E-09
RF	e	bc	a	bc	bc	a	de	cd	cd	ab	abc	3.8E-08
SGB	e	a	ab	ab	bc	ab	de	bcd	cd	abc	cde	1.6E-07
ADABOOST	d	bc	ab	abc	ab	a	cd	bc	ab	ab	ab	2.4E-05
RBFN	e	cd	a	cd	abc	ab	de	bcd	bc	abc	abc	4.2E-07

For example, Figure 6.8 shows the boxplots of the G-mean rank distribution of each treatment and how they are grouped for the classifier k-NN. These boxplots are listed in decreasing order (from left to right) of the mean rank. In addition, the right table illustrates the G-mean rank position and mean rank value for each treatment. Even though ADASYN is the treatment with higher mean rank, it shares the same group "**a**" as PSATE k-NN and Random. On the contrary, Figure 6.9 shows a different order and grouping for the G-mean rank distribution of each treatment for the SVM-G classifier. In this case, Random has the highest mean rank and it does not share group with any other treatment. For both classifiers the worst treatments are ENN and Raw (original dataset).

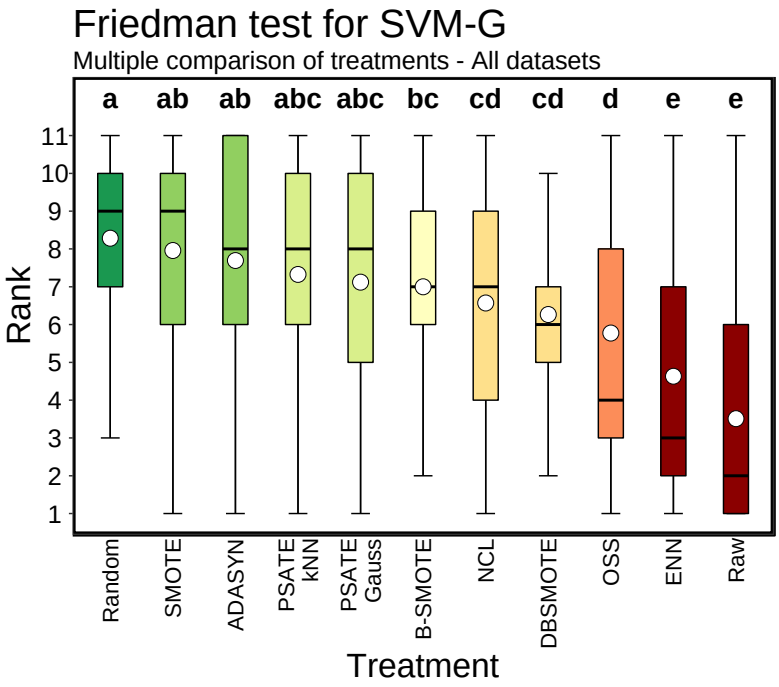


Ranking treatments for k-NN

Position	Treatment	Mean rank
11	**ADASYN**	**7.24**
10	PSATE kNN	7.06
9	Random	7.02
8	DBSMOTE	7.00
7	PSATE Gauss	6.96
6	OSS	6.88
5	SMOTE	6.80
4	NCL	6.78
3	B-SMOTE	5.63
2	ENN	5.08
1	Raw	4.94

Figure 6.8 Boxplots of the G-mean rank treatments for the classifier k-NN - All dataset
These boxplots are listed in decreasing order (from left to right) of the mean rank. In addition, the right table illustrate the G-mean rank position and the mean rank value for each treatment. Even though ADASYN is the treatment with higher mean rank, it shares the same group "**a**" as PSATE k-NN and Random treatments

These results call into question those in previous research on degradation problems (class imbalance, small-disjuncts, overlapping, sparseness, and noisy labels), which present successful techniques across few classifiers and datasets. More importantly, this analysis supports one of the hypotheses of the present research that states is not possible to find one treatment that is the best in remediation for all degradation problems, datasets or classifiers. The selection of the “best treatment” or even the most convenient classification algorithm depends on the available information and knowledge associated with the target dataset. For this reason, the diagnostic model for degradation problems has a direct relevance for the treatment selection in order to obtain successful classification outcomes and avoid degradation.



Ranking treatments for SVM-G

Position	Treatment	Mean rank
11	**Random**	**8.29**
10	SMOTE	7.96
9	ADASYN	7.69
8	PSATE kNN	7.33
7	PSATE Gauss	7.12
6	B-SMOTE	7.00
5	NCL	6.57
4	DBSMOTE	6.27
3	OSS	5.78
2	ENN	4.63
1	Raw	3.51

Figure 6.9 Boxplots of the G-mean rank treatments for the classifier SVM-G - All datasets. These boxplots are listed in decreasing order (from left to right) of the mean rank. In addition, the right table illustrate the G-mean rank position and the mean rank value for each treatment. Random has the highest mean rank and it does not share group with any other treatment

As previously mentioned in the Chapter 5, PSATE strategy focus on controlling the negative effects related to high levels of criticality for class imbalance and small-disjuncts. Datasets with such characteristics are marked as “Profile A”. Based on the information of Table 6.4, there are 8 datasets (Seismic, Wilt, Abalone, Blocks, nanoHUB, Fraud, Mammo, and SDSS) that cover such profile. Table 6.9 shows the Friedman test results for dataset with Profile A. In this case, there is statically differences between all treatments for each classifier because all *p-values* turns out to be much less than the 0.05 significance level.

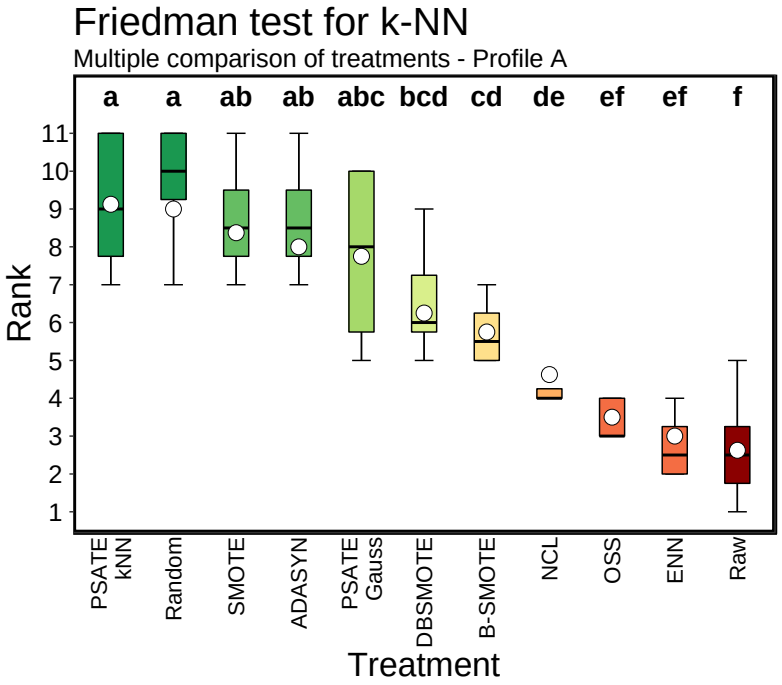
Table 6.9 Friedman test results for datasets with profile A.
There are statically differences between all treatments for each classifier because all *p-values* turns out to be much less than the 0.05 significance level.

Means of the G-mean of rank

Classifier	Treatment											**p-value**
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSAT Gauss	
NB	cd	ab	a	a	abc	ab	cd	cd	d	ab	bcd	1.4E-03
LOGREG	d	a	a	b	b	b	cd	c	cd	b	b	3.9E-09
LDA	c	a	ab	b	b	ab	c	c	c	ab	ab	6.2E-08
QDA	cd	a	ab	abc	cd	a	d	bcd	bcd	abc	bcd	6.0E-03
C5.0	d	ab	ab	bc	bc	a	d	cd	cd	ab	ab	6.3E-05
CART	cd	ab	a	c	c	a	d	cd	d	ab	b	7.0E-08
k-NN	f	a	ab	cd	bcd	ab	ef	de	ef	a	abc	3.3E-07
MLN	e	abcd	ab	abc	bcde	a	e	de	cde	abcd	a	6.2E-04
SVM-G	e	a	a	b	c	a	e	cd	d	a	a	4.5E-11
SVM-P	f	a	ab	abc	d	d	ef	e	ef	bcd	cd	5.1E-10
RF	d	bc	a	cd	ab	a	d	cd	d	a	a	1.7E-06
SGB	f	ab	ab	bc	cd	a	f	de	ef	ab	cd	4.8E-08
ADABOOST	e	de	ab	cd	bc	ab	de	de	e	a	ab	2.1E-07
RBFN	f	cd	ab	cd	bc	abc	ef	cde	def	ab	a	7.1E-06

Similarly, Figure 6.10 and Figure 6.11 show boxplots of the G-mean rank distribution for each treatment and how they are grouped for the classifiers k-NN and SVM-G respectively; however, these charts describe different behavior in terms of dispersion and grouping for treatments. Although ENN and Raw treatments remain the lowest ranking for both classifiers, the best treatments are PSATE k-NN for the k-NN classifier and PSATE Gauss for the SVM-G classifier. The next major aspect refers to the PSATE performance compared with other treatments. For this purpose, the Friedman's Aligned Rank post hoc test [37] was implemented. This pos hoc test allows to compare every treatment against a control, which in this case is PSATE. Therefore, the null hypothesis states that there is no statistical difference between PSATE and the compared treatment.

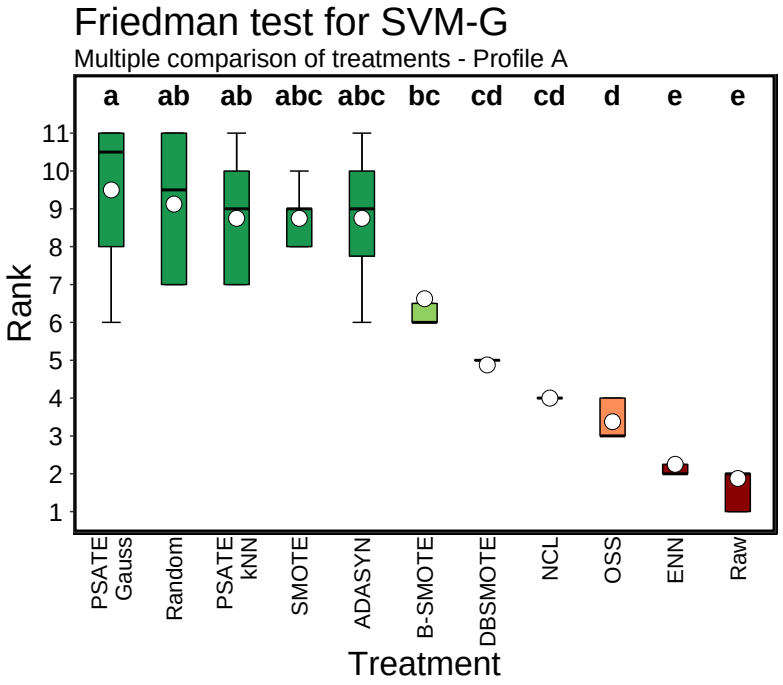
Complete outputs with *p-values* of this test can be found at the Appendix C; however, a summarized version of the results is shown in Table 6.10. The symbols "*" and "**" identify PSATE as the best treatment and the second-best treatment respectively. In general, for datasets with profile A (high class imbalance and high disjuncts), PSATE showed an outstanding performance for seven out of fourteen classifiers.



Ranking treatments for k-NN

Position	Treatment	Mean rank
11	**PSATE kNN**	**9.13**
10	Random	9.00
9	SMOTE	8.38
8	ADASYN	8.00
7	PSATE Gauss	7.75
6	DBSMOTE	6.25
5	B-SMOTE	5.75
4	NCL	4.63
3	OSS	3.50
2	ENN	3.00
1	Raw	2.63

Figure 6.10 Boxplots of the G-mean rank treatments for the k-NN - Profile A. These boxplots are listed in decreasing order (from left to right) of the mean rank. In addition, the right table illustrate the G-mean rank position and the mean rank value for each treatment. Even though PSATE kNN has the highest mean rank, it shares the same group "**a**" as Random treatment



Ranking treatments for SVM-G

Position	Treatment	Mean rank
11	**PSATE Gauss**	**9.50**
10	Random	9.13
9	PSATE kNN	8.75
8	SMOTE	8.75
7	ADASYN	8.75
6	B-SMOTE	6.63
5	DBSMOTE	4.88
4	NCL	4.00
3	OSS	3.38
2	ENN	2.25
1	Raw	1.88

Figure 6.11 Boxplots of the G-mean rank treatments for the SVM-G - Profile A
These boxplots are listed in decreasing order (from left to right) of the mean rank. In addition, the right table illustrate the G-mean rank position and the mean rank value for each treatment. **PSATE Gauss** has the highest mean rank and it does not share group with any other treatment

The feasibility of conducting similar analysis for different profiles of criticality between degradation problems resides in the fact of having enough datasets for such configuration. Based on the information from Table 6.4, it is possible to stablish a second profile (Profile B), which compares the treatments across classifiers for datasets with high levels of overlap. For this new Profile, there are 16 datasets (Blood, College, Glass, ILPD, Ionosphere, MDRR, Phoneme, Pima1, Pima2, Ringnorm, Satimage, Sonar, Spam, Sports, Vertebral, and Weather). The nonparametric test results for this case can be found at Appendix C. Table 6.11 shows a heat map of the predict success of treatments across classifiers for two dataset profiles in terms of the mean rank for the G-mean metric:

- Profile A: High levels of class imbalance and disjuncts
- Profile B: High levels of overlap and low levels of class imbalance

Table 6.10 Summary of PSATE performance for datasets with Profile A. The symbols "*****" and "******" identify PSATE as the best treatment and the second-best treatment respectively.

Performance ranking PSATE

Classifier	**Mean rank**	
	PSATE kNN	PSATE Gauss
NB	8.00	5.63
LOGREG	7.50	7.38
LDA	8.25	**8.75** **
QDA	6.88	5.50
C5.0	7.50	7.63
CART	8.25	7.25
k-NN	**9.13** *	7.75
MLN	7.25	**8.13** **
SVM-G	8.75	9.50 *
SVM-P	7.75	7.50
RF	8.25	**8.88** **
SGB	8.63	5.88
ADABOOST	**9.50** *	8.13
RBFN	8.50	**9.50** *

Table 6.11 Prediction of success of treatments across classifiers for two critical cases. High mean rank values of the G-mean tend to increase the steepness of the color gradient to green. On the contrary, low mean rank values tend to decrease the steepness of the color gradient to red. Profile A: High levels of class imbalance and disjuncts. Profile B: High levels of overlap and low levels of class imbalance

Classifier	Profile	Treatment										
		Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSAT Gauss
NB	**A**											
	B											
LOGREG	**A**											
	B											
LDA	**A**											
	B											
QDA	**A**											
	B											
C5.0	**A**											
	B											
CART	**A**											
	B											
k-NN	**A**											
	B											
MLN	**A**											
	B											
SVM-G	**A**											
	B											
SVM-P	**A**											
	B											
RF	**A**											
	B											
SGB	**A**											
	B											
ADABOOST	**A**											
	B											
RBFN	**A**											
	B											

The following list shows some relevant findings from Table 6.11:

- In general, training sets treated with remediation techniques showed better classification performance than Raw training sets.
- Data cleaning (ENN, NCL, and OSS) techniques showed low performance for profiles A and B.
- For NB (Naive Bayes) classifier the best remediation technique for datasets with Profile B (High levels of overlap and low levels of class imbalance) is ENN.
- SMOTE treatment takes top places in terms of increasing the classification performance.
- BDSMOTE technique takes top places associated with Profile B (High levels of overlap and low levels of class imbalance) for LOGREG (Logistic Regression), CART (Classification and Regression Trees), and SVM-P (Support Vector Machines with Gaussian Polynomial).
- PSATE Gauss showed low performance associated with Profile B (High levels of overlap and low levels of class imbalance) for NB (Naive Bayes) and SGB (Stochastic Gradient Boosting).
- PSATE treatment takes top places associated with Profile A (High levels of class imbalance and disjuncts) for classifiers k-NN (k-Nearest Neighbors), LDA (Linear Discriminant Analysis), MLN (Multilayer Neural Network), RF (Random Forest), ADABOOST (AdaBoost Classification Trees), and RBFN (Radial Basis Function Network).

7. SUMMARY AND OUTLOOK

The performance of supervised learning algorithms is hindered by the presence of degradation problems and lack of diagnosis for training sets before building classifiers. This thesis describes how the introduction of diagnostic methods help to minimize degradation in classification performance. Moreover, a new technique called “PSATE” (**P**robabilistic **SA**mpling **TE**chnique) for handling degradation problems is proposed, which is directly related to the diagnostic model outputs and focus on mitigating problems related to class imbalance and small-disjuncts.

In this research, an experimental framework was proposed to describe the complete diagnostic method and the new technique, where experimental results derive from statistical analysis over every combination of the 49 datasets, 14 classifiers, and 11 treatments (techniques for handling degradation problems).

Experimental results in this research allowed to implement statistical validation related to performance degradation in supervised learning. 1) The more class imbalance, the bigger classifier performance degradation for metrics G-mean and F1-Score. On the other hand, AUC metric is not affected by the imbalance problem since it is an optimistic and misleading performance metric. 2) It is not true that the greater number of disjuncts, the bigger classifier performance degradation; however, disjuncts detection allows to know the imbalance ratio within classes. 3) It is true that the larger overlap regions, the bigger classifier performance degradation.

Contrary to previous studies [101], [102], this research found empirical evidence that rebalancing the classes artificially have a positive effect on the predictive performance of classification algorithms. Furthermore, synthetic data generation may be useful strategy when scarcity of training samples is a problem due to characteristics of the phenomena studied itself or cost associated with data collection. For those cases, the goal is to minimize the probability of discarding instances located on overlap regions or rare cases, rather mislabeling instances.

In general, SMOTE treatment takes top places in terms of increasing the classification performance even overpassing its most recent extensions such as Borderline-SMOTE and DBSMOTE. However, SMOTE based techniques ignore the problem of small-disjuncts, more specifically subclass imbalance (imbalance within classes). When training sets have high imbalance and large number of disjuncts, PSATE has an outstanding performance for seven out of fourteen classifiers.

Even though data cleaning methods are designed to handling overlap problems, experiments over datasets with Profile B (high levels of overlap) showed poor classification performance. In general, results for the three data cleaning techniques used in this research (ENN, NCL, and OSS) were overpassed by sampling techniques. This can be explained by the wrong generalization of the concept of "Noisy Labels" or "mislabeling". In other words, instances located in overlap regions not necessarily have wrong labels. Thereby, these strategies of removing or label reallocation increase the problem of sparseness and reduce the classification performance. This latter result is a generalization of the Quinlan [47] results applied to decision trees.

In the light of the previous situation, PSATE conducts data cleaning procedures if, and only if, noisy instances are detected between two well-separated subclasses. Otherwise, those instances are retained and associated with overlapping effects instead of a mislabeling causes. Consistent with this view, PSATE leaves the overlap problem on charge of the classification algorithm.

Results from datasets with high levels of class imbalance and disjuncts support that classification performance is significantly improved by increasing the number of synthetic Positive instances. In contrast, cleaning techniques such as ENN, NCL, and OSS showed adverse effects on classification performance.

An important goal for the future is thus to implement the diagnostic model and PSATE as functional components of a new package in R open source software.

It would also be worthwhile to study the classifier and treatments performance for different profiles associated with levels of the degradation problems.

APPENDIX A. DIAGNOTIC MODEL DETAILS

Gaussian Mixture Models (GMM)

The Multivariate Gaussian distribution can be defined over a D-dimensional vector x of contiguous variables, which is given by

$$\mathcal{N}(\mathrm{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\Sigma|^{1/2}}\exp\left\{-\frac{1}{2}(\mathrm{x}-\mu)^{T}\Sigma^{-1}(\mathrm{x}-\mu)\right\} \qquad (6)$$

where μ is the mean vector of dimension D, Σ is the covariance matrix of dimension $D \times D$, and $|\Sigma|$ denotes the determinant of Σ.

Then, Gaussian Mixture Models can be written as a linear superposition of K Multivariate Gaussian distributions [64], [65] in the form

$$p(\mathrm{x}|\pi,\mu,\Sigma) = \sum_{k=1}^{K}\pi_k\,\mathcal{N}(\mathrm{x}|\mu_k,\Sigma_k) \qquad (7)$$

Where $\pi_k = \{\pi_1, \dots \pi_K\}$ is the mixing coefficient vector (probability vector of membership) and $\sum_{k=1}^{K}\pi_k = 1$ in order to be valid probabilities. Moreover, $\mu_k = \{\mu_1, \dots \mu_K\}$ and $\Sigma_k = \{\Sigma_1, \dots \Sigma_K\}$.

For a given training set $\mathbf{X} = \{\mathrm{x}_1, \dots, \mathrm{x}_N\}$, $\hat{\mathcal{L}}$ is the Maximum Likelihood Estimator (LME) and its log-likelihood function is expressed by Equation 9, which estimates the probability of membership of K components. The EM algorithm is used to maximize this function and estimate the mixture parameters [66].

$$\hat{\mathcal{L}}(\boldsymbol{X};\theta) = p(\boldsymbol{X}|\pi,\mu,\Sigma) \qquad (8)$$

$$log\, p(\boldsymbol{X}|\pi,\mu,\Sigma) = \sum_{n=1}^{N} log\left\{\sum_{k=1}^{K}\pi_k\mathcal{N}(x|\mu_k,\Sigma_k)\right\} \qquad (9)$$

Determining the optimal number of components K continues to be a problem for empirical cluster techniques such as k-means. However, probabilistic approaches based on mixture models can solve this problem using a penalty function. If the number of components K is increased in the Equation 9, it results in an increase in the dimensionality of the model, causing a monotonous increase in its likelihood. This is particularly a problem because it is not useful to obtain as many components as instances. Therefore, the best GMM is the one that maximizes the Bayesian Information Criterion (BIC) [67], [68], which seeks to balance the increase in likelihood and the complexity of the model by introducing a penalty term for each parameter. BIC is defined by

$$BIC(\boldsymbol{X};\theta) = log\, p(\boldsymbol{X}|\pi,\mu,\Sigma) - \frac{1}{2}\eta(\theta)log(N) \qquad (10)$$

where $\eta(\theta)$ is the number of free parameters in the model that represents the complexity of the model.

GMM build ellipsoidal subclasses (clusters), centered at the means μ_k and with covariance Σ_k. The covariance matrix can be determined by geometric characteristics. For instance, Banfield et al. [103] develop a covariance matrix model in terms of its eigenvalues decomposition in the term form

$$\Sigma_k = \lambda_k D_k A_k D_k^T \qquad (11)$$

where D_k is the orthogonal matrix of eigenvectors and defines the orientation, A_k is a diagonal matrix whose elements are proportional to the eigenvalues of Σ_k and determines the shape of the density contours, and λ_k is a scalar that defines the volume of the corresponding ellipsoid. This covariance model is the generalization of several earlier proposals based on GMM. Table A. 1 and Figure A. 1 are taken for the "*mclust*" [104], [105] package in R open source software.

Table A. 1 shows names, models, and geometric interpretation of the covariances model implemented for this research. Figure A. 1 illustrates the parametrization concepts related to the covariance models.

Table A. 1 Parametrization of the covariance matrix for Gaussian models

Covariance Matrix Models. Source [104]

Model	Σ_k	Distribution	Volume	Shape	Orientation
EII	$\lambda \mathrm{I}$	Spherical	Equal	Equal	—
VII	$\lambda_k \mathrm{I}$	Spherical	Variable	Equal	—
EEI	$\lambda \mathrm{A}$	Diagonal	Equal	Equal	Coordinate axes
VEI	$\lambda_k A$	Diagonal	Variable	Equal	Coordinate axes
EVI	λA_k	Diagonal	Equal	Variable	Coordinate axes
VVI	$\lambda_k A_k$	Diagonal	Variable	Variable	Coordinate axes
EEE	λDAD^T	Ellipsoidal	Equal	Equal	Equal
EVE	$\lambda DA_k D^T$	Ellipsoidal	Equal	Variable	Equal
VEE	$\lambda_k DAD^T$	Ellipsoidal	Variable	Equal	Equal
VVE	$\lambda_k DA_k D^T$	Ellipsoidal	Variable	Variable	Equal
EEV	$\lambda D_k AD_k^T$	Ellipsoidal	Equal	Equal	Variable
VEV	$\lambda_k D_k AD_k^T$	Ellipsoidal	Variable	Equal	Variable
EVV	$\lambda D_k A_k D_k^T$	Ellipsoidal	Equal	Variable	Variable
VVV	$\lambda_k D_k A_k D_k^T$	Ellipsoidal	Variable	Variable	Variable

Ellipses of isodensity for covariance models. Source [104]

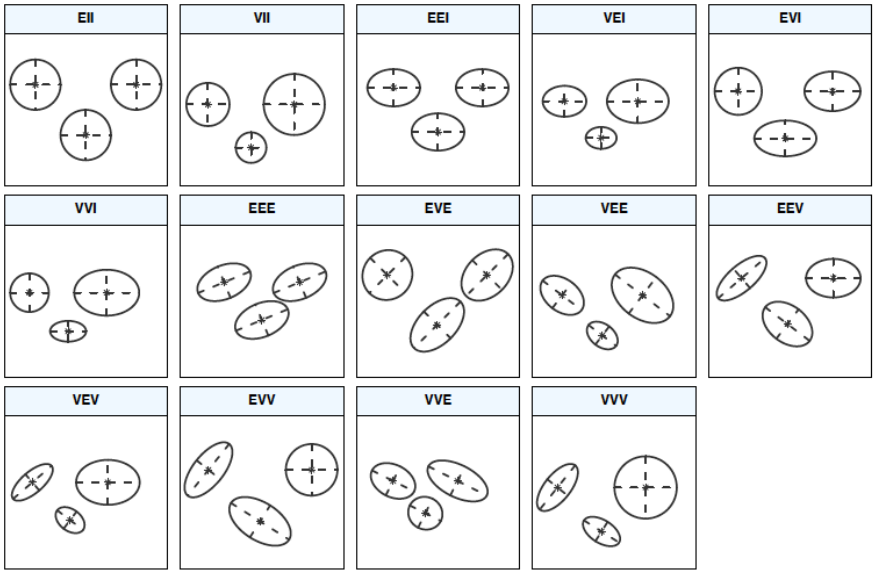


Figure A. 1 Ellipses of isodensity for covariance models obtained by eigen-decomposition in case of tree groups in two dimensions.

Separation Index

The separation index (J^*) was proposed by Qiu et al. [69], [70]. This index measures the magnitude of the gap between pairs of subclasses. It has a value between -1 and $+1$, where Negative values indicate subclasses are overlapped, zero means subclasses are touching, and Positive values indicate subclasses are separated. Initially, it is necessary to find the optimal projection in one-dimension space in which two subclasses have the maximum separation.

The initial projection is selected between two possible methods: $(\Sigma_1 + \Sigma_2)^{-1}(\mu_2 - \mu_1)$ and $(\mu_2 - \mu_1)$. Then, the Newton-Raphson method is used to search the optimal projection. Therefore, it requires that both covariances matrices from subclasses must be positive definite. This assumption is covered because all subclasses follow a multivariate Gaussian distribution. In a nutshell, the covariance matrix must be a symmetric positive definite matrix in order to the Equation 12 makes sense. Let J_{12}^* be the optimal separation index between subclasses 1 and 2. Then, $J_{12}^* = J_{12}(a^*)$, where a^* is the optimal projection direction witch maximizes J_{12}. Finally, the Separation index J^* is estimated as follow

$$J(a^*) = \frac{a^T(\mu_2 - \mu_1) - q_{\alpha/2}\left(\sqrt{a^T\Sigma_1 a} + \sqrt{a^T\Sigma_2 a}\right)}{a^T(\mu_2 - \mu_1) + q_{\alpha/2}\left(\sqrt{a^T\Sigma_1 a} + \sqrt{a^T\Sigma_2 a}\right)} \qquad (12)$$

where μ_1, μ_2, Σ_1, and Σ_2 are the probabilistic parameters of the two subclasses, $\alpha \in (0{,}05)$ is a tuning parameter indicating the percentage of data in the extremes to downweigh, $q_{\alpha/2} = z_{\alpha/2}$, where $z_{\alpha/2}$ is the upper $\alpha/2$ quantile of the univariate standard normal distribution.

APPENDIX B. PERFORMANCE METRICS AND DATASETS

Performance Metrics

The confusion matrix is used to describe the performance of a classification algorithm on a set of test data for which the true classes are known. Table B. 1 shows the confusion matrix elements for a binary classification problem. Traditionally, the overall accuracy (Acc) is the most important metric to evaluate the performance of a classifier, which can be calculated by using Equation 13.

Table B. 1 Confusion matrix for a binary classification task

		Prediction	
		Positive (P)	**Negative (N)**
Observations	**Positive (P)**	TP	FN
	Negative (N)	FP	TN

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

$$Sensitivity = \frac{TP}{TP + FN} = Acc^{+} \tag{14}$$

$$Specificity = \frac{TN}{TN + FP} = Acc^{-} \tag{15}$$

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

Sometimes the overall accuracy can be misleading [106] and models with a lower overall accuracy could have better predictive power. For instance, in case of imbalance data a classifier can reach high accuracy levels for the Negative class; however, the accuracy for the Positive class is low. Therefore, the overall accuracy has a bias toward the Negative class in imbalance data. To solve this situation, more specific metrics are used such as sensitivity (recall, True Positive Rate, or *Acc+*), specificity (True Negative Rate or Acc^{-}) and precision (Positive Predictive Value).

The Receiver Operating Characteristic (ROC) curve is a well-accepted technique for summarizing classifier performance [15].The area under the ROC curve (AUC) is a measure of how well a classifier can distinguish between two classes [88], [107].

$$AUC = \frac{1}{P \times N} \sum_{i=1}^{P} \sum_{j=1}^{N} 1_{p_i > p_j} \qquad (17)$$

Where i runs over all Positive instances with true class 1, and j runs over all Negative instances with true class 0; p_i and p_j denote the probability score assigned by the classifier to instance i and j, respectively. 1 is the indicator function: it outputs 1 if, and only if, the condition ($p_i > p_j$) is satisfied. In simple words, the ROC curve is generated by plotting the "True Positive Rate" (TPR) against the "False Positive Rate" (FPR). However, This AUC metric may provide an overly optimistic performance evaluation [16]. Thereby, performance evaluation metrics such as Precision-Recall curves, F1-score (F-measure), the geometric mean score (G-mean or G-measure), and the correlation coefficient (Phi) are considerate more convenient in the class imbalance context.

$$TPR = \frac{TP}{TP + FN} \qquad (18)$$

$$FPR = \frac{FP}{FP + TN} \qquad (19)$$

In Statistics, F1-score (also F-score or F-measure) uses the precision and sensitivity providing a single accuracy measurement for a classifier [86]. Formally, it is the harmonic average of the precision and recall, which return a value between 0 and 1. Where 1 indicates perfect precision and sensitivity.

$$F1 - score = \ 2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity} \qquad (20)$$

The geometric mean score G-mean (also G-measure) is the geometric mean of specificity and sensitivity, which return a value between 0 and 1 [38], [108].

$$G - mean = \sqrt{Sensitivity \times Specificity} \qquad (21)$$

$$G - mean = \sqrt{Acc^{+} \times Acc^{-}} \qquad (22)$$

In Statistics, the phi coefficient (also phi correlation coefficient or mean square contingency coefficient) measure of the degree of association between two binary variables, which are considered positively associated if most of the data falls along the diagonal cells of the confusion matrix. Phi coefficient return a value between -1 and 1, where values between 0.7 to 1 indicate strong association.

$$\phi = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (FN + TN) \times (TP + FN) \times (FP + TN)}} \qquad (23)$$

Datasets information

Table B. 2 Datasets with sampling parameters and IR

id	Name	IR train	IR test	Over%	Under%	New IR train
1	Satimage	9.69	8.48	500	120	1.00
2	EEG	1.23	1.23	20	600	1.00
3	Blocks	45.44	49.63	2000	105	1.00
4	ILPD	2.50	2.45	100	200	1.00
5	Glass	1.88	2.60	100	180	1.11
6	QSAR	2.11	1.71	110	190	1.00
7	Ozone	13.76	12.67	1200	108	1.00
8	Occupancy	3.21	3.76	200	150	1.00
9	Vertebral	2.45	1.53	200	120	1.00
10	Haberman	3.02	2.13	200	150	1.00
11	Spam	1.51	1.53	50	300	1.00
12	Gamma	1.86	1.87	100	185	1.00
13	Blood	2.56	3.48	200	125	1.20
14	Seismic	13.50	16.23	1000	110	1.00
15	Wilt	57.45	1.67	2000	105	1.00
16	Abalone	19.89	24.46	1000	110	1.00
17	Audit	1.98	1.57	100	198	1.00
18	Wireless	2.94	3.16	100	200	1.00
19	User-Knowledge	9.75	4.58	900	105	1.07
20	Shuttle	4.00	3.84	300	133	1.00
21	Image	1.27	1.43	30	420	1.00
22	Happiness	1.08	0.72	20	530	1.22
23	Skin	2.38	4.00	100	200	1.00
24	Seeds	1.82	2.45	100	182	1.11
25	Musk1	1.40	1.08	50	280	1.06
26	CTG	3.58	3.40	200	170	1.13
27	Sonar	1.18	1.06	20	590	1.00
28	Forest	4.35	6.07	400	109	1.16
29	HTRU2	9.90	9.97	800	110	1.00
30	Adult	3.13	3.23	200	150	1.00
31	Sports	1.86	1.52	100	185	1.08
32	Banknote	1.21	1.28	20	600	1.00
33	Electrical	1.78	1.73	100	177	1.13
34	Wine	2.94	2.20	200	146	1.00
35	Breast	1.03	0.49	10	1034	1.14
36	nanoHUB	39.63	38.02	2000	105	1.00
37	Simulated	4.70	5.18	300	130	1.00
38	Weather	4.52	4.55	400	113	1.11
39	Pima1	2.11	1.76	100	200	1.00
40	Ionosphere	1.76	1.96	76	233	1.00
41	Pima2	2.01	1.58	200	150	1.00
42	Ringnorm	1.02	1.02	2	5100	1.00
43	College	3.05	2.03	200	152	1.00
44	Iris	1.74	2.56	76	228	1.00
45	MDRR	1.36	1.16	40	339	1.00
46	Mammo	28.56	46.58	1000	110	1.00
47	Phoneme	2.39	2.48	100	200	1.00
48	Fraud	21.94	18.69	1000	110	1.00
49	SDSS	11.28	9.74	1000	110	1.00

The following is a list with complete names of the datasets and references:

1. Landsat Satellite (*Satimage*)
2. EEG Eye State (*EEG*)
3. Page Blocks Classification (*Blocks*)
4. Indian Liver Patient Dataset (*ILPD*)
5. Glass Identification (*Glass*)
6. QSAR biodegradation (*QSAR*) [109]
7. Ozone Level Detection-eighth (*Ozone*)
8. Occupancy Detection (*Occupancy*) [110]
9. Vertebral Column (*Vertebral*)
10. Haberman's Survival (*Haberman*)
11. Spambase (*Spam*)
12. MAGIC Gamma Telescope (*Gamma*)
13. Blood Transfusion Service Center (*Blood*) [111]
14. Seismic-bumps (*Seismic*) [112]
15. Wilt (*Wilt*) [113]
16. Abalone (*Abalone*)
17. Audit (*Audit*) this research work is supported by Ministry of Electronics and Information Technology Govt.of India
18. Wireless Indoor Localization (*Wireless*) [114]
19. User Knowledge Modeling (*User-Knowledge*) [115]
20. Shuttle (*Shuttle*)
21. Image Segmentation (*Image*)
22. Somerville Happiness Survey (*Happiness*) [116]
23. Skin Segmentation (*Skin*)
24. Seeds (*Seeds*)
25. Musk-Version1 (*Musk1*)
26. Cardiotocography (*CTG*)
27. Connectionist Bench (*Sonar*)
28. Forest type mapping (*Forest*) [117]
29. HTRU2 (*HTRU2*) [118]
30. Adult (*Adult*)
31. Sports articles for objectivity analysis (*Sports*) [119]
32. Banknote authentication (*Banknote*)
33. Electrical Grid Stability Simulated (*Electrical*)
34. Wine Recognition (*Wine*) [120]
35. Breast Cancer Wisconsin-Diagnostic (*Breast*) [121]
36. Simulation Patterns nanoHUB (*nanoHUB*),
37. Simulated dataset (*Simulated*)
38. Rain in Australia (*Weather*) [122]
39. Pima Indians Diabetes (*Pima1*) [123]
40. Johns Hopkins University Ionosphere (*Ionosphere*) [124]
41. Pima Indians Diabetes 2 (*Pima2*)
42. Leo Breiman's Ringnorm (*Ringnorm*) This dataset was taken from Delve (Data for Evaluating Learning in Valid Experiments).
43. U.S. News and World Report's College (*College*) this dataset was taken from the StatLib library which is maintained at Carnegie Mellon University. The dataset was used in the ASA Statistical Graphics Section's 1995 Data Analysis Exposition.
44. Edgar Anderson' s Iris data (*Iris*) [125]
45. Multidrug Resistance Reversal Agent (*MDRR*) [126]
46. Microcalcifications in Mammography (*Mammo*) [127]
47. Phoneme (*Phoneme*) It has been obtained from the ELENA Project
48. Credit Card Fraud Detection (*Fraud*) [128]–[130]
49. Sloan Digital Sky Survey DR14 (*SDSS*) [131]

APPENDIX C. STATISTICAL OUTPUTS

Table C. 1 Mean ranks of G-mean for all datasets

G-mean: mean of the ranks - All datasets

Classifier	Treatment											p-value
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSAT Gauss	
NB	6.78	7.69	7.86	6.20	5.67	6.43	7.02	6.35	6.06	6.61	4.63	2.0E-04
LOGREG	4.67	8.39	8.00	6.78	7.86	6.92	5.92	5.76	5.39	7.02	6.29	1.7E-10
LDA	4.92	8.79	7.33	6.79	6.58	7.77	5.19	5.54	5.42	7.19	6.31	9.2E-13
QDA	5.75	7.50	7.15	5.78	5.98	7.55	6.53	7.43	6.85	6.70	6.25	4.6E-02
C5.0	5.27	7.12	7.29	7.51	6.53	7.35	5.61	6.20	6.49	6.73	6.69	1.5E-03
CART	5.18	7.39	7.33	6.84	6.80	7.22	5.73	7.04	6.12	7.20	6.41	1.6E-04
k-NN	4.94	7.02	6.80	5.63	7.00	7.24	5.08	6.78	6.88	7.06	6.96	1.2E-05
MLN	5.33	7.06	6.73	7.16	6.51	7.57	6.10	6.73	6.18	6.31	6.27	**5.2E-02**
SVM-G	3.51	8.29	7.96	7.00	6.27	7.69	4.63	6.57	5.78	7.33	7.12	0.0E+00
SVM-P	4.61	7.84	8.22	6.55	7.57	7.08	5.45	6.80	5.92	6.78	6.08	2.0E-09
RF	5.00	6.88	7.86	6.73	6.69	8.06	5.31	6.43	6.37	7.65	7.31	3.8E-08
SGB	5.00	8.06	7.67	7.67	6.71	7.49	5.61	6.59	6.02	6.84	5.49	1.6E-07
ADABOOST	4.73	6.45	7.24	6.82	7.18	7.71	5.94	6.65	6.96	7.20	7.08	2.4E-05
RBFN	4.56	6.44	7.65	6.13	6.81	7.63	5.42	6.54	6.33	7.13	7.21	4.2E-07

For Tables C. 1, C. 2, C. 3, and C. 4, high mean rank values of the G-mean tend to increase the steepness of the color gradient to green. On the contrary, low mean rank values tend to decrease the steepness of the color gradient to red.

Table C. 1 shows the Friedman test results for experimental definition on Stage C, which includes the 49 datasets. According to the *p-value* column, only the MLN classifier cannot reject the null hypothesis because its *p-value* is equal to 0.05 (borderline result). That means, there is statistical differences between treatments for the remaining classifiers. However, the means of the G-mean ranks do not show improvement for most of the treatments. Therefore, global analysis across all datasets without using diagnostic profiles does not show any benefit.

Table C. 2 Mean ranks of G-mean for Profile A

G-mean: mean of the ranks - Profile A

Classifier	Treatment											p-value
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSAT Gauss	
NB	5.00	7.13	8.50	8.50	6.00	7.38	5.00	3.88	3.38	8.00	5.63	1.4E-03
LOGREG	2.00	9.75	9.63	7.00	7.13	7.50	2.63	4.13	3.63	7.50	7.38	3.9E-09
LDA	3.50	9.25	7.75	6.75	6.88	7.88	3.63	3.38	3.88	8.25	8.75	6.2E-08
QDA	4.88	9.00	7.88	6.50	3.88	8.63	4.13	5.75	5.88	6.88	5.50	6.0E-03
C5.0	3.25	7.63	8.50	6.13	6.25	8.88	3.00	4.25	3.88	7.50	7.63	6.3E-05
CART	3.75	8.13	10.00	5.00	5.13	9.38	3.13	4.88	3.25	8.25	7.25	7.0E-08
k-NN	2.63	9.00	8.38	5.75	6.25	8.00	3.00	4.63	3.50	9.13	7.75	3.3E-07
MLN	2.88	6.63	7.50	7.38	5.25	9.00	3.38	4.63	4.88	7.25	8.13	6.2E-04
SVM-G	1.88	9.13	8.75	6.63	4.88	8.75	2.25	4.00	3.38	8.75	9.50	4.5E-11
SVM-P	2.50	9.75	9.38	9.00	6.75	7.13	2.88	3.88	3.50	7.75	7.50	5.1E-10
RF	3.00	6.00	9.25	4.13	7.38	8.88	3.63	4.75	3.63	8.25	8.88	1.7E-06
SGB	2.50	8.63	9.13	7.13	5.50	9.25	2.25	4.75	3.25	8.63	5.88	4.8E-08
ADABOOST	2.75	3.88	9.13	5.63	7.25	9.00	4.38	4.13	3.38	9.50	8.13	2.1E-07
RBFN	2.13	6.13	8.88	5.88	6.88	7.63	3.25	5.38	3.63	8.50	9.50	7.1E-06

Table C. 2 shows the Friedman test results associated with datasets from Profile A (high class imbalance and large number of disjuncts), where all *p-values* are less than the level of significance (0.05). Therefore, there is statistical differences between treatments across classifiers. Moreover, the means of the G-mean ranks show differences between treatments. Table C. 3 shows the same test by using the AUC metric; however, most of the treatments do not shows statistical differences across classifiers (those marked in bold) because their *p-value*s are larger than the significance level. This result makes sense due the optimistic characteristic associated with the AUC metric.

Table C. 3 Mean ranks of AUC for Profile A

AUC: mean of the ranks - Profile A

Classifier	Treatment											p-value
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSAT Gauss	
NB	9.00	5.88	6.50	6.50	2.63	7.00	8.88	7.50	8.25	5.25	4.00	1.6E-02
LOGREG	7.88	9.25	8.00	5.38	6.75	7.38	7.88	8.00	6.50	6.63	3.88	**3.5E-01**
LDA	6.75	9.00	7.38	6.13	5.75	7.75	6.75	7.25	5.75	6.13	5.25	**5.2E-01**
QDA	8.38	9.75	6.00	4.13	1.50	6.38	9.00	7.88	8.13	5.50	5.75	3.8E-05
C5.0	5.50	5.50	7.38	6.00	8.13	6.13	5.63	4.38	5.88	7.88	6.63	**4.4E-01**
CART	3.13	5.13	9.00	6.50	7.63	7.75	2.50	4.88	4.88	8.25	8.38	4.5E-06
k-NN	4.88	4.88	7.63	6.13	7.38	7.00	4.25	5.00	4.75	6.63	10.25	2.1E-03
MLN	7.13	3.50	6.38	7.50	5.75	5.13	6.75	5.75	7.38	5.13	9.00	**8.4E-02**
SVM-G	6.00	7.00	6.50	6.00	4.75	5.38	5.88	5.75	5.88	8.50	8.75	**2.7E-01**
SVM-P	6.50	9.50	8.63	7.00	6.13	7.75	5.88	6.50	6.63	7.00	5.63	**3.8E-01**
RF	7.63	4.38	5.50	8.00	7.00	5.13	6.50	7.63	6.25	6.50	6.63	**5.2E-01**
SGB	5.38	9.00	6.88	6.38	5.75	6.88	5.88	6.25	5.38	7.50	5.38	**3.2E-01**
ADABOOST	3.75	3.50	5.88	6.88	6.50	6.25	6.25	5.75	5.63	8.50	8.50	**6.0E-02**
RBFN	5.88	6.25	5.63	7.63	4.63	4.50	6.38	6.25	7.75	8.00	6.50	**4.0E-01**

Table C. 4 Mean ranks of G-mean for Profile B

G-mean: mean of the ranks - Profile B

Classifier	Treatment											p-value
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN	PSAT Gauss	
NB	7.94	7.88	6.63	5.56	4.81	6.63	8.63	6.81	6.25	5.81	4.69	1.4E-03
LOGREG	4.31	7.44	7.19	6.56	7.38	6.13	6.00	4.44	5.44	7.13	6.63	3.9E-09
LDA	4.67	8.47	7.20	6.93	5.93	7.07	4.53	5.13	5.73	6.33	6.80	6.2E-08
QDA	5.58	6.75	5.75	4.00	7.00	6.83	7.00	7.92	7.17	7.33	6.92	6.0E-03
C5.0	4.88	6.06	6.88	7.81	6.44	5.81	6.38	5.19	6.31	7.75	7.44	6.3E-05
CART	5.81	8.13	5.88	5.06	7.94	5.13	6.38	7.25	7.31	7.25	6.38	7.0E-08
k-NN	4.50	6.44	5.56	5.13	6.75	6.06	5.06	7.38	8.19	6.69	6.38	3.3E-07
MLN	5.31	6.81	6.56	6.81	6.50	6.44	5.69	6.25	5.88	5.44	7.25	6.2E-04
SVM-G	2.88	8.25	7.63	7.19	6.81	8.06	3.94	7.31	5.56	7.19	6.94	4.5E-11
SVM-P	3.94	7.13	7.38	5.88	7.75	7.13	4.88	7.00	6.13	6.50	5.69	5.1E-10
RF	4.94	7.50	8.00	7.19	5.31	8.25	4.81	6.06	5.88	7.56	6.81	1.7E-06
SGB	5.19	8.44	7.06	8.13	7.13	5.81	5.44	6.44	6.56	6.38	5.38	4.8E-08
ADABOOST	4.50	6.75	7.19	7.00	6.50	7.25	6.06	5.94	6.19	6.50	7.94	2.1E-07
RBFN	3.56	5.88	8.56	5.00	6.44	7.63	4.31	6.38	6.25	7.69	6.19	7.1E-06

Table C. 4 shows the Friedman test results associated with datasets from Profile B (high overlapping levels), where all *p-values* are less than the level of significance (0.05). Therefore, there is statistical differences between treatments across classifiers. Moreover, the means of the G-mean ranks show differences between treatments.

Table C. 5 Friedman's Aligned Rank post hoc for PSATE Gauss

G-mean: Post hoc test - Profile A

Control: PSATE Gauss

Classifier	Treatment									
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE k-NN
	p-values									
NB	5.8E-01	3.3E-01	6.9E-02	6.7E-02	8.6E-01	2.7E-01	5.8E-01	1.8E-01	1.4E-01	1.4E-01
LOGREG	3.2E-04	1.2E-01	1.4E-01	8.1E-01	8.8E-01	9.2E-01	1.5E-03	3.3E-02	1.1E-02	9.5E-01
LDA	1.9E-04	7.5E-01	5.5E-01	1.9E-01	2.2E-01	5.8E-01	2.1E-04	2.1E-04	7.5E-04	7.5E-01
QDA	6.5E-01	2.5E-02	1.3E-01	5.2E-01	2.8E-01	4.4E-02	3.2E-01	9.1E-01	8.3E-01	4.0E-01
C5.0	4.8E-03	1.0E+00	5.6E-01	3.5E-01	3.8E-01	4.2E-01	2.7E-03	3.3E-02	1.7E-02	9.3E-01
CART	1.6E-02	6.3E-01	8.5E-02	1.3E-01	1.4E-01	1.9E-01	4.7E-03	1.1E-01	5.5E-03	5.7E-01
k-NN	6.5E-04	4.5E-01	6.6E-01	2.2E-01	3.6E-01	8.6E-01	1.4E-03	5.9E-02	5.6E-03	3.6E-01
MLN	8.8E-04	3.3E-01	6.8E-01	6.2E-01	6.9E-02	5.8E-01	2.7E-03	2.5E-02	3.7E-02	5.7E-01
SVM-G	5.6E-07	8.1E-01	6.1E-01	7.5E-02	4.3E-03	6.2E-01	1.7E-06	4.8E-04	5.9E-05	6.2E-01
SVM-P	4.8E-04	1.5E-01	2.6E-01	3.3E-01	7.1E-01	8.3E-01	7.5E-04	1.9E-02	5.5E-03	8.8E-01
RF	1.9E-04	6.0E-02	8.4E-01	2.5E-03	3.1E-01	9.3E-01	7.5E-04	6.7E-03	5.6E-04	6.1E-01
SGB	3.1E-02	8.0E-02	4.0E-02	4.3E-01	7.9E-01	3.3E-02	2.1E-02	4.8E-01	9.0E-02	8.0E-02
ADABOOST	4.1E-04	5.5E-03	5.4E-01	1.1E-01	5.8E-01	5.9E-01	1.4E-02	9.5E-03	2.2E-03	3.9E-01
RBFN	3.2E-06	2.8E-02	6.5E-01	1.9E-02	8.8E-02	2.2E-01	8.2E-05	7.6E-03	1.9E-04	5.2E-01

Table C. 6 Friedman's Aligned Rank post hoc for PSATE k-NN

G-mean: Post hoc test - Profile A

Control: PSATE k-NN

Classifier	Treatment									
	Raw	Random	SMOTE	B-SMOTE	DBSMOTE	ADASYN	ENN	NCL	OSS	PSATE Gauss
	p-values									
NB	4.3E-02	6.1E-01	7.3E-01	7.2E-01	1.9E-01	7.1E-01	4.3E-02	5.0E-03	3.0E-03	1.4E-01
LOGREG	2.5E-04	1.4E-01	1.6E-01	7.5E-01	8.2E-01	9.7E-01	1.2E-03	2.8E-02	8.9E-03	9.5E-01
LDA	6.5E-04	5.2E-01	7.8E-01	3.3E-01	3.6E-01	8.1E-01	7.1E-04	7.0E-04	2.3E-03	7.5E-01
QDA	1.9E-01	1.6E-01	4.9E-01	8.5E-01	5.6E-02	2.4E-01	6.7E-02	4.7E-01	5.3E-01	4.0E-01
C5.0	6.1E-03	9.3E-01	5.1E-01	3.9E-01	4.2E-01	3.7E-01	3.5E-03	4.1E-02	2.1E-02	9.3E-01
CART	2.8E-03	9.3E-01	2.5E-01	3.8E-02	4.1E-02	4.6E-01	6.8E-04	3.1E-02	8.2E-04	5.7E-01
k-NN	1.5E-05	8.6E-01	6.3E-01	3.1E-02	6.4E-02	4.5E-01	3.6E-05	4.9E-03	2.2E-04	3.6E-01
MLN	5.8E-03	6.9E-01	8.7E-01	9.5E-01	2.1E-01	2.6E-01	1.5E-02	9.3E-02	1.3E-01	5.7E-01
SVM-G	6.4E-06	8.0E-01	9.8E-01	2.0E-01	1.8E-02	1.0E+00	1.8E-05	2.7E-03	4.3E-04	6.2E-01
SVM-P	2.7E-04	1.9E-01	3.3E-01	4.1E-01	6.0E-01	7.1E-01	4.3E-04	1.2E-02	3.4E-03	8.8E-01
RF	1.3E-03	1.7E-01	4.8E-01	1.2E-02	6.1E-01	6.7E-01	4.2E-03	2.8E-02	3.3E-03	6.1E-01
SGB	9.4E-05	1.0E+00	7.7E-01	3.3E-01	4.4E-02	7.0E-01	4.9E-05	1.4E-02	5.6E-04	8.0E-02
ADABOOST	1.2E-05	2.9E-04	8.1E-01	1.4E-02	1.6E-01	7.5E-01	9.4E-04	5.7E-04	8.9E-05	3.9E-01
RBFN	5.9E-05	1.2E-01	8.5E-01	9.0E-02	2.9E-01	5.5E-01	9.7E-04	4.2E-02	2.0E-03	5.2E-01

Table C. 5 and Table C. 6 show Friedman's Aligned Rank post hoc for PSATE Gauss and PSATE k-NN respectively. This pos hoc test allows to compare every treatment against a control, which in this case is PSATE. Therefore, the null hypothesis states that there is no statistical difference between PSATE and the compared treatment. *p-values* marked in green reject the null hypothesis. Then, this information is subsequently contrasted with the means of the G-mean ranks from Table C. 2 to determine the PSATE position (Refer Table 6.10).

REFERENCES

[1] Institute of Medicine and National Academies of Sciences, Engineering, and Medicine, *Improving Diagnosis in Health Care*. Washington, DC: The National Academies Press, 2015.

[2] B. J. Bordini, A. Stephany, and R. Kliegman, "Overcoming Diagnostic Errors in Medical Practice," *J. Pediatr.*, 2017.

[3] M. L. Graber, "The incidence of diagnostic error in medicine," *BMJ Quality and Safety*. 2013.

[4] F. B. Felix Last, Georgios Douzas, "Oversampling for Imbalanced Learning Based on K-Means and SMOTE," 2017.

[5] A. Fernández, S. García, F. Herrera, and N. V. Chawla, "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary," *Journal of Artificial Intelligence Research*. 2018.

[6] W. Lu, Z. Li, and J. Chu, "Adaptive Ensemble Undersampling-Boost: A novel learning framework for imbalanced data," *J. Syst. Softw.*, vol. 132, pp. 272–282, 2017.

[7] A. X., W. J., S. V.S., Z. P., and V. S. . O. http://orcid. org/000.-0003-4960-174X A. O.-Z. Cui Z. AO - Sheng Pengpeng; ORCID: http://orcid.org/0000-0002-3099-4523, "Immune centroids oversampling method for binary classification," *Comput. Intell. Neurosci.*, 2015.

[8] C. Beyan and R. Fisher, "Classifying imbalanced data sets using similarity based hierarchical decomposition," *Pattern Recognit.*, 2015.

[9] V. García, J. Sánchez, and R. Mollineda, "An Empirical Study of the Behavior of Classifiers on Imbalanced and Overlapped Data Sets," in *Progress in Pattern Recognition, Image Analysis and Applications*, 2007, pp. 397–406.

[10] T. M. Khoshgoftaar, C. Seiffert, J. Van Hulse, A. Napolitano, and A. Folleco, "Learning with limited minority class data," in *Proceedings - 6th International Conference on Machine Learning and Applications, ICMLA 2007*, 2007.

[11] X. Y. Liu, J. Wu, and Z. H. Zhou, "Exploratory under-sampling for class-imbalance learning," in *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2006.

[12] R. Prati, G. Batista, and M. Monard, "Learning with class skews and small disjuncts," *Lect. notes Comput. Sci.*, 2004.

[13] Nitesh V. Chawla et. al, "SMOTE," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[14] N. V Chawla, N. Japkowicz, and P. Drive, "Editorial: Special Issue on Learning from Imbalanced Data Sets," *ACM SIGKDD Explor. Newsl.*, 2004.

[15] J. A. Swets, "Measuring the accuracy of diagnostic systems.," *Science*, vol. 240, no. 4857, pp. 1285–1293, 1988.

[16] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, 2006.

[17] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, 2002.

[18] G. M. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *J. Artif. Intell. Res.*, 2003.

[19] A. Estabrooks, T. Jo, and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," *Comput. Intell.*, 2004.

[20] P. Domingos, "MetaCost: A General Method for Making Classifiers Cost-Sensitive," *Proc. fifth ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, 1999.

[21] C. Elkan, "The foundations of cost-sensitive learning," in *IJCAI International Joint Conference on Artificial Intelligence*, 2001, pp. 973–978.

[22] R. C. Holte, L. Acker, and B. Porter, "Concept Learning and the Problem of Small Disjuncts.," *Ijcai*, pp. 813–818, 1989.

[23] N. Japkowicz, "Concept-Learning in the Presence of Between-Class and Within-Class Imbalances," in *Advances in Artificial Intelligence: 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2001 Ottawa, Canada, June 7--9, 2001 Proceedings*, E. Stroulia and S. Matwin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 67–77.

[24] A. S. Nickerson, N. Japkowicz, and E. Milios, "Using Unsupervised Learning to Guide Resampling in

Imbalanced Data Sets," *Proc. Eighth Int. Work. AI Statitsics*, p. 5, 2001.
[25] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," *ACM SIGKDD Explor. Newsl.*, vol. 6, no. 1, p. 40, 2004.
[26] N. Japkowicz, "Class imbalance: Are we focusing on the right issue?," *Proc. II Work. Learn. from Imbalanced Data Sets, ICML Conf.*, pp. 17–23, 2003.
[27] R. C. Prati, G. E. Batista, and M. C. Monard, "Class imbalances versus class overlapping: an analysis of a learning system behavior," *MICAI 2004 Adv. Artif. Intell.*, 2004.
[28] C. E. Brodley and M. A. Friedl, "Identifying Mislabeled Training Data," *J. Artif. Intell. Res.*, 1999.
[29] D. Opitz and R. Maclin, "Popular Ensemble Methods: An Empirical Study," *J. Artif. Intell. Res.*, 1999.
[30] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine*. 2006.
[31] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, 2010.
[32] Committee on Diagnostic Error in Health Care, *Improving Diagnosis in Health Care*. 2015.
[33] L. E. Beutler, K. Someah, S. Kimpara, and K. Miller, "Selecting the most appropriate treatment for each patient," *Int. J. Clin. Heal. Psychol.*, vol. 16, no. 1, pp. 99–108, 2016.
[34] J. Larry Jameson and D. L. Longo, "Precision Medicine-Personalized, Problematic, and Promising," *Obstet. Gynecol. Surv.*, 2015.
[35] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," in *Neural Networks*, 2003, vol. 16, no. 5–6, pp. 555–559.
[36] D. H. Wolpert, "The Supervised Learning No-Free-Lunch Theorems," in *Soft Computing and Industry*, 2011.
[37] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci. (Ny).*, 2010.
[38] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from Imbalanced Data Sets*. 2018.
[39] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and regression trees Regression trees*. Monterey, CA: Wadsworth and Brooks, 1984.
[40] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One Sided Selection," in *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.
[41] R. C. Holte, L. Acker, and B. Porter, "Concept Learning and the Problem of Small Disjuncts.," *IJCAI Int. Jt. Conf. Artif. Intell.*, pp. 813–818, 1989.
[42] C. Drummond and R. C. Holte, "C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," in *Proceedings of the International Conference on Machine Learning (ICML 2003). Workshop on Learning from Imbalanced Data Sets II*, 2003, pp. 1–8.
[43] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proceedings of the International Joint Conference on Neural Networks*, 2008.
[44] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "DBSMOTE: Density-based synthetic minority over-sampling technique," *Appl. Intell.*, 2012.
[45] G. M. Weiss and H. Hirsh, "A Quantitative Study of Small Disjuncts," in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, 2000, pp. 665–670.
[46] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "Balancing Strategies and Class Overlapping," in *Advances in Intelligent Data Analysis VI: 6th International Symposium on Intelligent Data Analysis, IDA 2005, Madrid, Spain, September 8-10, 2005. Proceedings*, 2005.
[47] J. R. Quinlan, "Induction of Decision Trees," *Expert Syst.*, 1986.
[48] J. R. Quinlan, "Learning from noisy data," in *Proceedings of the Second International Machine Learning Workshop*, 1983.
[49] J. R. Quinlan, "The Effect of Noise on Concept Learning," in *Machine Learning, An Artificial Intelligence Approach Volume II*, Morgan Kaufmann, 1986, pp. 149–166.
[50] P. Hart, "The Condensed Nearest Neighbor Rule (Corresp.)," *IEEE Trans. Inf. Theor.*, vol. 14, no. 3, pp. 515–516, Sep. 2006.
[51] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. Inf. Theory*, 1967.
[52] T. M. Cover, "Estimation by the Nearest Neighbor Rule," *IEEE Trans. Inf. Theory*, 1968.
[53] D. L. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," *IEEE Trans. Syst. Man Cybern.*, 1972.

[54] J. Laurikkala, “Improving Identification of Difficult Small Classes by Balancing Class Distribution,” in *Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine*, 2001, pp. 63–66.
[55] I. Tomek, “An Experiment with the Edited Nearest-Neighbor Rule,” *Syst. Man Cybern. IEEE Trans.*, vol. SMC-6, no. 6, pp. 448–452, 1976.
[56] K. C. Gowda and G. Krishna, “The Condensed Nearest Neighbor Rule Using the Concept of Mutual Nearest Neighborhood,” *IEEE Trans. Inf. Theory*, 1979.
[57] C. E. Brodley and M. A. Friedl, “Identifying and Eliminating Mislabeled Training Instances,” *Comput. Eng.*, 1996.
[58] X. Ester, M., Kriegel, H. P., Sander, J., & Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” *Kdd*, 1996.
[59] N. Japkowicz, “Concept-Learning in the Presence of Between-Class and Within-Class Imbalances,” in *Advances in Artificial Intelligence: 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2001 Ottawa, Canada, June 7--9, 2001 ProceedingsArtificial Intelligence*, 2001.
[60] X. Ai, J. Wu, Z. Cui, X. Xian, and Y. Yao, “Enrich the data density of cluster for imbalanced learning using immune representatives,” in *2016 IEEE International Conference on the Science of Electrical Engineering, ICSEE 2016*, 2017.
[61] D. Boley, “Principal direction divisive partitioning,” *Data Min. Knowl. Discov.*, 1998.
[62] J. MacQueen, “Some Methods for classification and Analysis of Multivariate Observations,” in *5th Berkeley Symposium on Mathematical Statistics and Probability 1967*, 1967.
[63] S. P. Lloyd, “Least Squares Quantization in PCM,” *IEEE Trans. Inf. Theory*, 1982.
[64] K. Pearson, “Contributions to the Mathematical Theory of Evolution,” *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, 2006.
[65] D. M. Titterington, A. F. M. Smith, and U. E. Makov, *Statistical Analysis of Finite Mixture Distributions*. Wiley, New York, 1985.
[66] D. B. R. A. P. Dempster N. M. Laird, A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data via the EM Algorithm,” *J. R. Stat. Soc. Ser. B*, 1977.
[67] G. Schwarz, “Estimating the dimension of a model,” *Ann. Stat.*, vol. 6, no. 2, pp. 461–464, 1978.
[68] C. Fraley and A. Raftery, “How Many Clusters ? Which Clustering Method ? Answers Via Model-Based Cluster Analysis,” *Comput. J.*, 1998.
[69] W. Qiu and H. Joe, “Separation index and partial membership for clustering,” *Comput. Stat. Data Anal.*, 2006.
[70] W. Qiu and H. Joe, “Generation of random clusters with specified degree of separation,” *Journal of Classification*. 2006.
[71] M. J. Anderson, “Distance-based tests for homogeneity of multivariate dispersions,” *Biometrics*, 2006.
[72] M. J. Anderson, K. E. Ellingsen, and B. H. McArdle, “Multivariate dispersion as a measure of beta diversity,” *Ecol. Lett.*, 2006.
[73] M. J. Anderson, “PERMDISP: a FORTRAN computer program for permutational analysis of multivariate dispersions (for any two-factor ANOVA design) using permutation tests,” 2004.
[74] M. B. Brown and A. B. Forsythe, “Robust tests for the equality of variances,” *J. Am. Stat. Assoc.*, 1974.
[75] D. Dua and C. Graff, “{UCI} Machine Learning Repository.” 2017.
[76] M. Kuhn, “caret Package,” *J. Stat. Softw.*, 2008.
[77] M. Kuhn, “Building predictive models in R using the caret package,” *J. Stat. Softw.*, 2008.
[78] M. Kuhn, “A Short Introduction to the caret Package,” *R Found. Stat. Comput.*, 2017.
[79] M. Kuhn, “caret Package: Classification and Regression Training,” *https://Cran.R-Project.Org/*, 2017.
[80] G. M. Weiss, “The Effect of Small Disjuncts and Class Distrubution on Decision Tree Learning,” Rutgers University, 2003.
[81] N. Japkowicz, “Class imbalances: are we focusing on the right issue,” in *ICML Workshop on Learning from Imbalanced Data Sets*, 2003.
[82] G. M. Weiss, “Mining with Rarity: A Unifying Framework,” *SIGKDD Explor. Newsl.*, 2004.
[83] M. Maloof, “Learning When Data Sets are Imbalanced and When Costs are Unequal and Unknown,” *Analysis*, 2003.
[84] F. Provost and T. Fawcett, “Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions,” *KDD*, 1997.
[85] F. Provost, T. Fawcett, and R. Kohavi, “The Case Against Accuracy Estimation for Comparing Induction Algorithms,” in *Proceedings of the Fifteenth International Conference on Machine Learning1*, 1997.

[86] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, 2009.
[87] G. Weiss and F. Provost, “The effect of class distribution on classifier learning: an empirical study,” 2001.
[88] A. P. Bradley, “The use of the area under the ROC curve in the evaluation of machine learning algorithms,” *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, 1997.
[89] T. Fawcett, “ROC Graphs: Notes and Practical Considerations for Researchers,” *HP Labs Tech Rep. HPL-2003-4*, 2004.
[90] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006.
[91] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets,” *PLoS One*, 2015.
[92] N. V Chawla, N. Japkowicz, and A. Kołcz, “Editorial: Special Issue on Learning from Imbalanced Data Sets,” *ACM SIGKDD Explor. Newsl.*, 2004.
[93] H. B. Mann and D. R. Whitney, “On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other,” *Ann. Math. Stat.*, 1947.
[94] M. Hollander and D. Wolfe, *Nonparametric Statistical Methods, 2nd Edition*. 1999.
[95] W. J. Conover, *Practical Nonparametric Statistics.*, 3rd ed. New York, 1999.
[96] M. Friedman, “The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance,” *J. Am. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, 1937.
[97] M. Friedman, “A Correction: The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance,” *J. Am. Stat. Assoc.*, vol. 34, no. 205, 1939.
[98] M. Friedman, “A Comparison of Alternative Tests of Significance for the Problem of m Rankings,” *Ann. Math. Stat.*, vol. 11, no. 1, pp. 86–92, 1940.
[99] R. A. Fisher, *The design of experiments*. London: Oliver and Boyd, 1935.
[100] H. P. Piepho, “An algorithm for a letter-based representation of all-pairwise comparisons,” *J. Comput. Graph. Stat.*, 2004.
[101] F. Provost, “Machine learning from imbalanced data sets 101,” *Proc. AAAI’2000 Work. …*, 2000.
[102] C. Drummond and R. C. Holte, “Exploiting the cost (in)sensitivity of decision tree splitting criteria,” *Int. Conf. Mach. Learn.*, 2003.
[103] J. D. Banfield and A. E. Raftery, “Model-Based Gaussian and Non-Gaussian Clustering,” *Biometrics*, no. 49, pp. 803–821, 1993.
[104] L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery, “mclust 5: Clustering, Classification and Density Estimation Using Gaussian Finite Mixture Models.,” *R J.*, 2016.
[105] C. Fraley and A. Raftery, “MCLUST: Software for model-based cluster and discriminant analysis,” 1998.
[106] F. Provost, T. Fawcett, and R. Kohavi, “The case against accuracy estimation for comparing induction algorithms,” in *15th International Conference on Machine Learning*, 1998.
[107] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. 2000.
[108] N. Japkowicz, “Assessment Metrics for Imbalanced Learning,” in *Imbalanced Learning*, 2013.
[109] K. Mansouri, T. Ringsted, D. Ballabio, R. Todeschini, and V. Consonni, “Quantitative structure-activity relationship models for ready biodegradability of chemicals,” *J. Chem. Inf. Model.*, 2013.
[110] L. M. Candanedo and V. Feldheim, “Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models,” *Energy Build.*, 2016.
[111] I. C. Yeh, K. J. Yang, and T. M. Ting, “Knowledge discovery on RFM model using Bernoulli sequence,” *Expert Syst. Appl.*, 2009.
[112] M. Sikora and L. Wrobel, “Application of rule induction algorithms for analysis of data collected by seismic hazard monitoring systems in coal mines,” *Arch. Min. Sci.*, 2010.
[113] B. A. Johnson, R. Tateishi, and N. T. Hoan, “A hybrid pansharpening approach and multiscale object-based image analysis for mapping diseased pine and oak trees,” *Int. J. Remote Sens.*, 2013.
[114] J. G. Rohra, B. Perumal, S. J. Narayanan, P. Thakur, and R. B. Bhatt, “User localization in an indoor environment using fuzzy hybrid of particle swarm optimization & Gravitational search algorithm with neural networks,” in *Advances in Intelligent Systems and Computing*, 2017.
[115] H. T. Kahraman, S. Sagiroglu, and I. Colak, “Developing intuitive knowledge classifier and modeling of users’ domain dependent data in web,” *Knowl. Based Syst.*, vol. 37, pp. 283–295, 2013.
[116] W. W. Koczkodaj *et al.*, “How to reduce the number of rating scale items without predictability loss?,” *Scientometrics*, 2017.
[117] B. Johnson, R. Tateishi, and Z. Xie, “Using geographically weighted variables for image classification,” *Remote Sens. Lett.*, 2011.

[118] B. W. Stappers, S. Cooper, J. M. Brooke, and J. D. Knowles, “Fifty years of pulsar candidate selection: From simple filters to a new principled real-time classification approach,” *Mon. Not. R. Astron. Soc.*, 2016.
[119] N. Hajj, Y. Rizk, and M. Awad, “A subjectivity classification framework for sports articles using improved cortical algorithms for Feature Selection,” *Neural Computing and Applications*, 2018.
[120] M. Forina, R. Leardi, C. Armanino, and S. Lanteri, “PARVUS: An Extendible Package for Data Exploration, Classification and Correlation,” *J. Chemom.*, 1990.
[121] W. H. Wolberg and O. L. Mangasariant, “Multisurface method of pattern separation for medical diagnosis applied to breast cytology (linear programniing/pattern recognition/expert systems/cancer diagnosis),” 1990.
[122] K. S. Mwitondi, “Data mining with Rattle and R,” *J. Appl. Stat.*, 2012.
[123] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes, “Using the ADAP learning algorithm to forecast the onset of diabetes mellitus,” in *Proceedings of the Symposium on Computer Applications and Medical Care*, 1988.
[124] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker, “Classification of radar returns from the ionosphere using neural networks,” *Johns Hopkins APL Tech. Dig. (Applied Phys. Lab.*, 1989.
[125] E. Anderson, “The irises of the Gaspe Peninsula,” *Bull. Am. Iris Soc.*, vol. 59, pp. 2–5, 1935.
[126] V. Svetnik, A. Liaw, C. Tong, J. Christopher Culberson, R. P. Sheridan, and B. P. Feuston, “Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling,” *J. Chem. Inf. Comput. Sci.*, 2003.
[127] L. P. Woods, Kevin S. Solka, Jeffrey L. Priebe, Carey E. Doss, Chris C. Bowyer, Kevin W. Clarke, “Comparative evaluation of pattern recognition techniques for detection of microcalcifications,” in *SPIE 1905, Biomedical Image Processing and Biomedical Visualization*, 1993.
[128] A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, “Calibrating probability with undersampling for unbalanced classification,” in *Proceedings - 2015 IEEE Symposium Series on Computational Intelligence, SSCI 2015*, 2015.
[129] A. Dal Pozzolo, O. Caelen, Y. A. Le Borgne, S. Waterschoot, and G. Bontempi, “Learned lessons in credit card fraud detection from a practitioner perspective,” *Expert Syst. Appl.*, 2014.
[130] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, “Credit card fraud detection: A realistic modeling and a novel learning strategy,” *IEEE Trans. Neural Networks Learn. Syst.*, 2018.
[131] M. ~R. Blanton *et al.*, “Sloan Digital Sky Survey IV: Mapping the Milky Way, Nearby Galaxies, and the Distant Universe,” *aj*, vol. 154, p. 28, Jul. 2017.