UNIVERSIDAD
DE GRANADA

NEW APPROACHES TO IMPROVE THE PERFORMANCE OF MACHINE LEARNING AND DEEP LEARNING ALGORITHMS IN SOLVING REAL-WORLD PROBLEMS: COMPANIES FINANCIAL FAILURE FORECASTING

Thesis submitted by

HUTHAIFA RIYAD ALJAWAZNEH

To obtain the Ph.D. degree as part of the

PROGRAMA DE DOCTORADO EN TECNOLOGÍAS DE LA INFORMACIÓN Y LA COMUNICACIÓN

Supervisors

PEDRO ÁNGEL CASTILLO VALDIVIESO
ANTONIO MIGUEL MORA GARCÍA

October 15, 2021

Editor: Universidad de Granada. Tesis Doctorales
Autor: Huthaifa Riyad Aljawazneh
ISBN: 978-84-1117-208-0
URI: http://hdl.handle.net/10481/72340

DECLARATION OF AUTHORSHIP

I declare that I have done this Doctoral Thesis under the direction of my supervisors. Therefore, I assume the authorship of the contents except those mentions of other authors' works, which are duly referenced.

I also declare that the work is unpublished and is not plagiarism, in whole or in part, of any other research carried out by other persons.

Likewise, I assert that the results and data exposed in this document have not been falsified and that any error that may exist has not been introduced intentionally.

Granada, October 15, 2021

Sgd. Huthaifa Riyad Aljawazneh

To my father, the one who sets the stage for me..

ACKNOWLEDGEMENTS

First of all, I would love to express my gratitude to my parents and my brothers as this work wouldn't have been done without their unconditional love and usual support. My sincere gratitude for my father, the one who is always aiming to set the stage for me, and he is the reason behind who I am now. Further, I would like to thank my uncle Youssef for encouraging me alongside my Ph.D. Journey and his wife Dr. Erika for her absolute help and assistance. In addition, sincerest thanks to my supervisors, Dr. Pedro Ángel Castillo Valdivieso and Dr. Antonio Miguel Mora García for their thoughtful comments, suggestions and recommendations which added much value to the dissertation. In addition, couldn't forget to thank Dr. Pablo García Sánchez for his insightful contribution as well in this very intense academic year. I´m grateful to the effort they made for me and really It was an honor working with all of them. Last but not least, many thanks to miss Nouran for her permanent interest and kindness, thanks for everything..

ACRONYMS

A

AdaBoost Adaptive Boosting
ADASYN-I Adaptive Synthetic Sampling
AHC Aglomerative Hierarchical Clustering
AI Artificial Intellegence
ANS Adaptive Neighbor Synthetic minority oversampling
AND Automatic Neighborhood size Determination
AUC Area Under ROC Curve

C

CART Classification And Regression Trees
CE-SMOTE Cluster Ensemble SMOTE
Cluster-SMOTE Clustering-Based Implementation of SMOTE
CNN Convolutional neural networks
CGIs Corporate Governance Indicators
CSC Cost-Sensitive Classification

D

DT Decision Trees Classifier
DBN Deep Belief Networks
DL Deep Learning
DLR Delay Line Reservoir
DBBI- SMOTE Distance-Based Border Instances SMOTE

E

EWSs Early Warning Systems
ENN Edited Nearest Neighbor
EMIS Emerging Markets Information Service
EXGB Extreme Gradient Boosting

F

FA Factor Analysis
FN False Negatives
FNR False Negative Rate
FP False Positive
FPR False Positive Rate
FRs Financial Ratios

G

GMBoost Geometric Mean based Boosting
G-SMOTE Partially guided hybrid sampling approach

H

HVDM Heterogeneous Value Distance Metric

I

ISMOTE Improved SMOTE
IPF Iterative-Partitioning Filter

K

KNN K-Nearest Neighbor

L

LVQ Learning Vector Quantization
LVQ-SMOTE Learning Vector Quantization based SMOTE
LDA Linear Discriminant Analysis
LN Local Neighbourhood
LLE Locally Linear Embedding
LR Logistic Regression
Logit Logistic Regression
Logit Logistic Regression
LSTM Long-Short Term Memory

M

Max_Features	Number of Features
Max_Samples	Number of Samples
ML	Machine Leaning
MSE	Mean Square Error
MSMOTE	Modified SMOTE
MLP-6L	Multilayer Perception model of 6 layers
MLP	Multi-layer Perceptron
MDA	Multivariate Discriminate Analysis
MWMOTE	Majority Weighted Minority Oversampling TEchnique

N

NCR	Neighbor Cleaning Rule
NTSMOTE	Neighborhood Triangular SMOTE
NT-SMOTE	Neighborhood Triangular SMOTE
NNs	Neural Networks
N_estimators	Number of Estimators

P

PD	Polish Dataset
PCA	Principal Components Analysis

Q

QDA	Quadratic Discriminant Analysis

R

RBF	Radial Basis Function
RF	Random Forest
ROS	Random Oversampling
ROC	Receiver Operating Characteristics
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Networks
REPT	Reduced Error Pruning Tree

RBM Restricted Boltzmann Machines
RST Rough Set Theory

S

SD Spanish Dataset
SDSMOTE Support Degree SMOTE
SMOTE-IPF SMOTE with Iterative-Partitioning Filter
SMOTE-ENN SMOTE with Edited Nearest Neighbor
SMOTE-PSO SMOTE and Particle Swarm Optimization
SMOTE-RS$B*$ SMOTE with Rough Sets theory
SMOTE Tomek SMOTE with Tomek Links
SD Spanish Dataset
Stdev Standard Deviation
SDA Stepwise Discriminant Analysis
SVM Support Vector Machine
SN-SMOTE Surrounding Neighborhood-based SMOTE
SMOTE Synthetic Minority Oversampling TEchnique
SIEDS Systems and Information Engineering Design Symposium

T

TD Taiwanese dataset
TN True Negatives
TP True Positives

X

XGBoost eXtreme Gradient Boosting

ABSTRACT

Companies' financial failure prediction is one of the most crucial real-world problems. This is because many companies are interested in forecasting their incoming financial status in order to adapt to the current financial and business environment to avoid bankruptcy. In addition, commercial banks are interested in gaining this prior information about the future financial status of companies as a requirement that supports the decision of providing loans to companies in some cases, and even the investors are interested in this information. On this basis, the main objective of this doctoral thesis is to improve the performance of Machine learning and Deep Learning algorithms in predicting companies' financial failure. The main challenge of predicting the companies' financial failure using these kinds of algorithms is the scarcity of companies' bankruptcy occurrence in the real-world, making the real companies' financial datasets extremely imbalanced. In other words, the inconsistent distribution of the financial data dramatically affects the overall performance and reliability of the classifiers. To carry out this main objective, new simple resampling approaches have been proposed in this study to solve the data balancing problem. These simple approaches aim mainly to avoid the overfitting that arises as a consequence of the 'simple' replication of the minority instances (bankrupt companies records) in order to balance the dataset, and also to prevent losing some important information that happens as a consequence of eliminating some majority instances (solvent companies records) to solve the data balancing problem. Accordingly, the simple balancing approaches are based mainly on splitting the imbalanced dataset into several balanced subsets processed by the classifiers individually. Afterwards, a comprehensive analysis of the impact of using several different balancing methods on the performance of classical classification algorithms in predicting companies' financial failure has been done. The selected balancing methods are analyzed bearing in mind the existing types in the literature.

This analysis aims to conclude the most appropriate balancing technique to solve the financial data inconsistency distribution. Furthermore, in order to improve the performance of several classical classifiers in predicting companies' bankruptcy, cascading technique have been used to create hybrid classifiers, showing better performance than using stand-alone ones. A further step in this doctoral thesis is to customize Deep Learning algorithms by identifying a specific number of hidden layers and hyperparameter values to maintain the highest performance in predicting bankrupt and solvent companies. Finally, a novel data balancing technique baptized as Distance Based Border Instances SMOTE (DBBI-SMOTE) has been developed to solve the inconsistent distribution of the financial data. This novel method avoids some drawbacks in the existing balancing methods procedures, such as generating the new minority instances in the majority region. In addition, it outperforms many other balancing methods addressed in the literature. Thus, the novel approach is considered as a preprocessing stage of several standard and ensemble classifiers, yielding significant improvements in their reliability and overall performance.

RESUMEN

La predicción de la quiebra de empresas se considera como un problema crítico en el mundo real, ya que muchas empresas necesitan tener previsiones de su situación financiera futura para adaptarse al entorno financiero y empresarial del momento y evitar la quiebra. Tanto los bancos como los inversores están interesados en disponer de información sobre el estado financiero futuro de las empresas, como requisito para apoyar la decisión de conceder préstamos a las empresas en algunos casos. Así pues, el objetivo principal de esta tesis doctoral es mejorar el rendimiento de diversos algoritmos de Machine Learning y Deep Learning ante el problema de la predicción de la quiebra de empresas. El principal reto es la escasez de patrones de quiebra de empresas en el mundo real, lo que hace que los conjuntos de datos financieros de las empresas reales estén extremadamente desbalanceados. En otras palabras, hay una distribución inconsistente de los datos financieros en los conjuntos de datos que afecta al rendimiento general y a la fiabilidad de los clasificadores. Para solventar este problema se han propuesto nuevos enfoques sencillos de 'remuestreo' de datos para resolver el problema del balanceo de los conjuntos de datos. Los métodos propuestos tienen como objetivo principal evitar el sobreajuste que surge como consecuencia de replicar las instancias minoritarias (registros de empresas en quiebra) para equilibrar el conjunto de datos, lo cual se hace en algunos métodos simples, y también para evitar la pérdida de información importante que se produce como consecuencia de la eliminación de algunas instancias de la clase mayoritaria (registros de empresas solventes), que se realiza en otros métodos, para hacer frente al problema del desbalanceo de los datos. Así, los enfoques simples de equilibrado propuestos se basan principalmente en dividir el conjunto de datos desequilibrado en varios subconjuntos equilibrados que son procesados por los clasificadores de forma individual. Junto con esto, se ha realizado un análisis exhaustivo del impacto que tiene el uso

de varios métodos de balanceado de datos en el rendimiento de diferentes algoritmos de clasificación clásicos para la predicción de la quiebra de empresas. Este análisis tiene como objetivo obtener la técnica de balanceado más adecuada para resolver el problema de distribución desbalanceada de datos financieros. Además, con el fin de mejorar el rendimiento de los clasificadores clásicos en la predicción de la quiebra de empresas, se han utilizado técnicas en cascada para crear clasificadores híbridos que ofrezcan un mejor rendimiento que el obtenido utilizando los clasificadores independientes más sencillos. Una aportación adicional de esta tesis doctoral es el estudio de la configuración óptima de diferentes algoritmos de Deep Learning, estableciendo un número adecuado de capas ocultas, así como de los valores de los hiperparámetros de dichos métodos, a fin de para obtener el máximo rendimiento en la predicción de la quiebra de empresas. Finalmente, se ha desarrollado una nueva técnica de balanceo de datos denominada Distance Based Border Instances SMOTE (DBBI-SMOTE) para resolver el desbalanceo de datos financieros. Este nuevo método evita algunos problemas, de los métodos de balanceo de datos existentes actualmente, como por ejemplo generar nuevas instancias minoritarias en la región mayoritaria. Asimismo, el nuevo método supera en rendimiento a muchos otros mencionados en la literatura. La aplicación de este nuevo método es una etapa de preprocesamiento que se aplica a varios clasificadores estándar y agrupados, obteniendo así una mejora claramente significativa en su fiabilidad y rendimiento general.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

CONTENTS

Nowadays, the demand on using the computer in our life is increasing excessively due to its utility in performing a wide range of applications could make the human life easier, and even safer. One of the most important parts of computer science is Artificial intelligence (AI)[1], which is the science to construct smart machines, in particular intelligent computer applications [2]. Accordingly, the main factors that give the AI this importance are the reliability, precision and speed in solving issues or providing services that could affect a wide range of aspects of human life. The issues solved, or the applications provided by the AI of the computer can be classified in a range of normal to very critical issues that could catastrophically affect the human life, such as a failure in autopilot system. Particularly, one of the major fields that basically rely on AI to solve are the real-world problems. Many researchers devoted using a part of AI (i.e., Machine Learning)[3] to solve a wide range of real-world problems such as Image Processing [4], Computer Vision [5], Financial Forecasting [6], Cars Self-Driving [7], Spam Detection [8], Fraud Detection [9], Speech Recognition [10], Cyber security [11], and many more. Thus, as a critical example of real-world problems solving, Machine Learning and more recently Deep Learning have been applied to a great extent in solving the problem of predicting companies financial failure because it has a crucial impact on the future of business. Accordingly, this doctoral thesis studies the problem of predicting companies' financial failure, and proposed new approaches aiming to improve the reliability and the overall performance of Machine learning and Deep learning algorithms in solving it. Noteworthy, the main obstacle in solving this problem is the scarcity of the accruing companies' bankruptcy , which makes it a great challenge for the classifiers to predict the correct financial case of companies, (i.e., bankrupt or solvent) and guaranteeing a considerable reliability in that sense.

1.1 CONTEXT AND MOTIVATIONS

The relevance of companies' financial failure prediction problem is evident in today's world due to its effects on banks, businesses, a large number of stakeholders, including workers, creditors and suppliers, and eventually, even entire countries. There might be huge financial losses encountered due to bad judgment and analysis.

Thus, providing a specific approach to deal with financial data is a challenging task due to the usual high imbalanced distribution in it, where the number of insolvent companies is much less than those successful ones; the occurrence of companies' financial failure is infrequent compared to the solvency in the real-world. In other words, the companies' real financial data is extremely imbalanced. Accordingly, the data inconsistent distribution badly affects the reliability and the overall performance of classifiers in predicting companies' financial failure, and strongly raises the need for using resampling techniques in order to solve the problem. A classifier is a software tool or programming representation of algorithms that process the data observations or samples in order to determine the class of new ones [12]. On the other hand, the resampling can be defined as a procedure of increasing or decreasing the amount of samples in the dataset in order to solve the data balancing problem [13]. Thus, the classifiers tend to select the easiest way guaranteeing a very high accuracy at the expense of reliability, which is assigning the new instances to the majority instances class (solvent companies) and avoiding the other class. Thus, the first and easiest technique to solve the data inconsistent distribution problem is the random oversampling, which is based mainly on replicating the minority instances (bankrupt companies records) randomly until achieving a balanced state for the dataset. This technique can simply cause overfitting in the classifier. This means that the classifier learns the minority instances very well due to the highly replicating of them, but in the validation on new data or test data, the classifier moderately mispredicts the bankrupt companies.

Accordingly, this doctoral thesis studies and analyses existing methods, trying to find the best data processing and hybridisation of techniques, as well as their best configuration to maximize their performance in this problem. Thus, new simple resampling approaches are provided and supposed to avoid the random replicating of the minority instances that cause the overfitting case. In addition, this thesis aims to analyze the impact of many of the state of the art balancing techniques on the performance of the classifiers and conclude the appropriate ones to deal with the financial data and leads the classifiers to the minimum bankrupt companies mispredicting. Besides, it aims to improve the performance of the classical classifiers in predicting bankrupt companies by the hybridizing them. Furthermore, due to the importance of the prior knowledge about companies' financial failure, this thesis studies the use of Deep Learning methods as robust tools that could yield very high performance in predicting companies' financial failure compared to standard Machine Learning methods. Furthermore, the final aim of this thesis is to develop a novel balancing technique that can fill the gap and avoid some drawbacks that could be happened by the existing techniques such as generating the new minority instances in the majority region, or using certain categories of the minority instances to generate the new instances and avoid the others.

1.2 OBJECTIVES

Taking into account the main motivations described in Section 1.1, the main objective of this thesis is to make relevant contributions in improving the performance of Machine Learning and Deep Learning algorithms in predicting companies financial failure. The specific main objectives are detailed below.

1.2.1 *Analysing classical classification algorithms with new simple resampling techniques to predict companies financial failure*

The first specific objective is to test new simple resampling approaches to solve the inconsistent data distribution problem, and avoiding the occurrence of the overfitting that happened by applying the random oversampling technique. Moreover, the new resampling approaches are based mainly on splitting the original companies dataset into several balanced subsets, each subset comprised of the whole bankrupt companies record and a certain portion of the majority instances. Then, use these subsets to train three well-known classifiers in order to predict companies financial failure. Finally, the final results will be obtained by finding the average of the outcomes yielded by each subset.

1.2.2 *Studying classical classification algorithms with advanced balancing techniques to predict companies' financial failure*

The second objective of this thesis is to analyze the impact of using several different balancing techniques on the performance and reliability of classifiers in predicting companies financial failure. The outcome of this analysis is to conclude the best balancing technique and its best configuration to prepare the companies dataset before using it to train the classical classification algorithms.

1.2.3 *Improving the performance of classical classification algorithms in predicting companies financial failure*

The third objective of this thesis is to hybridize classical classification algorithms using cascading technique, and then use the hybrid models to predict companies financial failure. This, in turn, will have a significant impact on the overall performance of the classification algorithms in solving the same real-world problem.

1.2.4 *Applying Deep Learning algorithms with advanced data balancing methods to predict companies' financial failure*

The fourth objective of this thesis is to optimize the configuration of Deep Learning algorithms in order to achieve the highest performance and reliability in predicting companies financial failures, and outperforming other machine learning algorithms in solving the same problem. In other words, the fourth objective is presenting the customized Deep learning methods as solid alternatives that could outperform all of the standard machine learning and statistical methods in forecasting the companies' financial status.

1.2.5 *Designing and developing a novel advanced data balancing technique to solve the financial data inconsistent distribution problem*

The final specific objective of this thesis is to design and develop a novel balancing technique filling the gap and avoiding some drawbacks of several existing balancing techniques procedures. The novel balancing technique will be used to balance several datasets of companies belonging to different markets, and vary with respect to the data type and balancing ratios. Then, the generated balanced datasets by applying the novel balancing technique will be used to train several classifiers in order to evaluate the feasibility and reliability of the novel method. In addition, studying the impact of many other balancing techniques covering wide range of the existing types in the literature regarding solving the same problem, and compare it with the novel technique.

1.3 THESIS STRUCTURE

This section presents a brief description of the chapters that make up this thesis. Thus, the structure of the thesis is as follows:

- **Chapter 1. Introduction**: it introduced this thesis, and presented the motivations and the objectives of it.

- **Chapter 2. Comprehensive literature review**: This chapter presents a comprehensive review about the statistical and the artificial intelligence methods that had been used in the literature to predict companies' financial failure.

- **Chapter 3. Methodology**: this chapter presents in details the Machine Learning and Deep Learning algorithms that have been used along this doctoral thesis to predict companies financial failure.

- **Chapter 4. Using classical classification algorithms with new simple sampling approaches to predict companies' financial failure**: this chapter presents new simple resampling approaches used in order to solve the financial data inconsistent distribution problem, and avoiding the occurrence of the overfitting happened by applying the random oversampling.

- **Chapter 5. Using simple classification algorithm with several different advanced balancing techniques to predict companies' financial failure**: this chapter presents a comprehensive analysis about the impact of several different advanced balancing techniques on the performance of classical classifiers in predicting companies financial failure.

- **Chapter 6. Improving the performance of classical machine learning algorithms in predicting companies' financial failure**: this chapter present a new approach to improve the performance of classical classification algorithms in predicting companies financial failure by hybridizing them.

- **Chapter 7. Using Deep Learning algorithms with advanced data balancing techniques to predict companies' financial failure**: this chapter presents several customized Deep Learning methods as a solid alternative to predict companies' financial failure, and outperforming several high performance ensemble-based Machine Learning classifiers.

- **Chapter 8. A novel data balancing technique: DBBI-Smote (Distance-Based Border Instances SMOTE)**: This chapter presents a novel technique to solve the financial data inconsistent distribution problem, and outperforms several balancing techniques addressed in the literature.

- **Chapter 9. Conclusions**: this chapter summarizes the conclusions drawn from the results obtained and the contributions of this doctoral thesis. In addition, the future work is also exposed.

2 COMPREHENSIVE LITERATURE REVIEW

CONTENTS

This chapter presents a literature review about companies financial failure prediction (bankruptcy), focusing on the most relevant statistical and artificial intelligence methods addressed in the literature to solve the problem.

Financial failure prediction is a critical matter that occupies the efforts of many researchers, since an inaccurate decision about the companies' financial status could cause costly financial losses. Mostly, the prediction of companies' financial status could be done using statistical techniques such as Linear Discriminant Analysis (LDA), Multi-Discriminant Analysis (MDA) and Logistic Regression (LR or Logit); or by Artificial intelligence algorithms[14, 15].

2.1 STATISTICAL METHODS

Actually, many researchers adopted the statistical methods to predict companies' financial failure. In the sixties, Altman [16] used MDA to predict companies' financial status using their financial statements. Later, Martin [17] proposed using logit to predict banks financial failure. Furthermore, in the eighties, the use of MDA in bankruptcy models was reduced [18]. Ohlson [19] adopted Logit to predict companies' financial failure. More recently, Kolari et al. [20] developed two Computer-based Early Warning Systems (EWSs) to predict large US banks financial failures, based mainly on logit and trait recognition, respectively. However, the system that based of trait recognition outperformed the other one based on logit with respect of predicting both classes. In addition, Jones and Hensher [21] proposed a mixed Logit model, and compared it with a standard Logit model in predicting companies financial distress, proving that the mixed Logit model yields better results than the standard one. One year later, Montgomery et al. [22] discussed the financial circumstances could causes a financial failure in Japanese and Indonesians domestic banks using Logit. Canbas et al. [23] constructed an integrated early warning system to predict the financial status of Turkish commercial banks. The authors combined discriminant analysis, Logit and probit models to construct the model which provide a considerable solution to avoid the wrong estimations about the future of the banks in Turkey.

Furthermore, Lanine and Vennet [24], developed two models to predict the financial status of Russian commercial banks. The first model based on a parametric logit, whereas the second model is a modified nonparametric trait recognition model. Both models showed a good performance in predicting the bankrupt banks, but the modified trait recognition model obtained better results compared to logit and the traditional trait recognition model. Also, In [25], the authors devoted Logit analysis to forecast Russian banks bankruptcy. Later, Berg [26] proposed Logit-based Generalized Additive models (GAM) to predict Norwegian firms financial status. Thus, dataset of Norwegian firms between the period (1996–2000) adopted to train GAM-Logit approach. The authors reported that GAM-Logit significantly outperformed other models addressed in the same study. Minussi et al. [27], adopted Logit in order to predict the financial status of Brazilian companies. Later, in [28], the authors analyzed the impact of several feature selection approaches on Logit and three other algorithms in predicting the financial failures of US banks. Then, Bhunia [29] used MDA to predict the financial status of Indian companies with different financial, business and operating conditions. The proposed model showed high accuracy (between 86% and 96%). Thus, according to the accuracy, MDA proved that it is still a considerable alternative to predict companies financial failure and verging the other predictors used in the literature. In 2016, Brozyna et al. [30] used LDA and Logit to predict the financial status of Polish and Slovak companies. Most recently, Horváthová and Mokrišová [31], compared the performance of Logit to BCC DEA model in predicting business failure in Slovakia. The acronym BCC refers to the first letters of the names of its three founders (i.e., Banker, Chames and Cooper), and it is a model to measures the efficiency of technical estimations. In addition, the acronym DEA refers to Data Envelopment Analysis. Thus, the authors reported that BCC DEA model significantly outperforms the Logit model with respect to predicting both cases; the success and the failure. Also, In [32], the authors made a careful analysis about improving the knowledge of bankruptcy prediction of large European companies using the discriminant analysis.

2.2 ARTIFICIAL INTELLIGENCE METHODS

Many studies adopted using the Artificial Intelligence Methods to predict companies' financial failure. the Artificial Intelligence Methods can be grouped into two categories: Machine Learning and Deep Learning methods.

2.2.1 *Machine learning methods*

In the case of using classical machine learning algorithms, Baek and Cho [33], made a training trick in order to improve the performance of a proposed auto-associative neural network in predicting companies financial failure. In other words, they use a sample dataset contain only solvent companies to train auto-associative neural network, and then used a sample dataset contained records for bankrupt and solvent companies. The new training approach relatively enhanced the performance of the proposed method. Later, Hui and Sun [34] depended on SVM to do an empirical study about the financial status of Chinese companies. Also, Bose and Pal [35] made a study comparing several methods to forecast the financial status. They proved that neural networks accuracy is better than SVM for this aim. Afterwards, Li and Sun [36] in their study improved the financial status prediction accuracy by using a straightforward wrapper approach in order to complement SVM. Later, Heo and Yang[37] compared the performance of AdaBoost, artificial neural network, SVM, decision tree, and Z-score in predicting the financial failure of large Korean construction companies. Noteworthy, the proposed Korean construction companies dataset was extremely imbalanced. Thus, AdaBoost showed the best performance in predicting companies financial failure compared to the other classifiers addressed in that study. In [38], the authors studied the impact of combining the Financial Ratios (FRs) and Corporate Governance Indicators (CGIs) on the classifiers' performance in predicting Taiwanese companies' financial status. The problem of the inconsistent data distribution was solved by selecting a balanced subset using stratified sampling method. The selected subset contained 239 records for bankrupt companies and another 239 records for solvent companies. Thus, five well-known classifiers were compared, i.e., SVM, KNN, CART, MLP and Naïve Bayes. Then, combining the FRs and the CGIs improved the performance of the classifiers, Stepwise Discriminant Analysis (SDA) Feature Selection method with SVM obtained the best results. In addition, Zięba et al. [39], compared several classifiers performance with a novel approach that applies EXtreme

Gradiant Boosting (EXGB) for learning an ensemble of decision trees in order to predict companies' financial status. In their study, they introduced a new concept named synthetic feature in order to obtain higher-order statistics in data. An extremely imbalanced Polish companies' dataset, collected during 2007 to 2013 for bankrupt companies, and 2000 to 2012 for still operating ones was used to evaluate the novel approach performance. Thus, the proposed approach obtained significant results with respect to the referenced methods they applied such as J48, RF, SVM and AdaBoost. In addition, Karas and Reznakova[40], developed a customized model based on Classification and Regression Trees (CART) to predict construction companies financial failure. Also, Le et al.[41] discussed the impact of using the balancing techniques on the Korean companies' bankruptcy status prediction performance. Thus, four classification models were used to predict the financial status, namely; RF, Decision tree, MLP and SVM. The dataset used was extremely imbalanced, so 6 balancing techniques were utilized to solve this problem; (SMOTE, BL-SMOTE, SMOTE-ENN, SMOTE-Tomek and ADASYN). Furthermore, the classification models applied on the data before and after balancing, RF outperform the other models in both cases, but using RF with SMOTE-ENN obtained the best results.

2.2.2 *Deep learning methods*

Actually, there are not many studies that apply DL methods to predict companies' financial failure using companies' real data [42]. However, some researchers used financial data as a graphical representation. Yeh et al. [43], predicted companies financial status using DBN, the return of stock markets for solvent and bankrupt companies were presented as binary images and then were utilized in order to train the models. They proved that DBN outperforms SVM classical classification method. Also, Hosaka [44] proposed a method based on Convolutional Neural Networks (CNN) to predict bankruptcy using Japanese stock market data represented as a grayscale image. Moreover, the proposed method obtained the optimum results compared to classical and other DL classification methods. On the other hand, following classification using tabular data, Wyrobek [45] compared the performance of convolutional neural networks (CNN) to several machine learning (i.e., SVM, RF, gradient boosting decision trees, neural network with one hidden layer (NN), NB) and statistical methods (i.e., discriminant analysis (DA) and logit) in forecasting the financial failure of Polish companies. Thus, the gradient boosting decision trees model showed best performance compared to the other classifier including the deep learning one. Jang et al.[46] compared LSTM, Feed-forward neural network and SVM regarding predicting Business Failure relying on listed US construction contractors. The same authors [47] also proposed a model based on LSTM to predict the business failure probability from one to three years using accounting, construction market and macroeconomic variables. Moreover, the SMOTE-Tomek balancing technique was used as a data preprocessing stage in both works, obtaining better results than using only the accounting variables. In addition, Vochozka et al. [48], proposed using LSTM to predict the financial status of construction companies in Czech Republic.

2.3 COMPARISON BETWEEN MACHINE LEARNING AND STATISTICAL METHODS

In order to identify the most appropriate classification algorithm to solve the problem of companies financial failure prediction, many researcher compared the performance of different classifiers in solving the problem. For instance, Pompe and Feelders [49] compared the performance of LDA with classification trees and neural networks in this problem, and proved that neural networks outperform the rest of methods. Min and Lee [50] compared SVM, MDA, Logit and three-layer fully connected back-propagation neural networks regarding bankruptcy prediction, with SVM obtaining the best results. Also, Xu and Wang [51], compared MDA to Logitic and SVM in predicting the financial failure of cooperates listed in Shanghai stock exchange. In the three models the authors considered using efficiency as a predictor variable. Lin [52], discussed in details the performance of several methods in predicting the finical failures of public industrial firms in Taiwan. Thus, the methods addressed in that study were Multiple discriminate analysis (MDA), logit, probit, and artificial neural networks (ANNs). A dataset of Taiwanese public industrial companies in the period of 1998-2005 obtained from Taiwan Economic Journal adopted to train the predictors. The the probit, logit, and ANN models achieved very good results with respect to the prediction accuracy. The probit model showed the best and stable performance outperforming the remaining predictors addressed in that study. More recently, Mansouri et al. [53], compared the performance of three layers artificial neural network with Logit in predicting the financial failure of companies listed in Tehran stock exchange (TSE) in Three, two and one year in advance. Thus, the authors reported that the three layers artificial neural network outperformed the Logit model in the case of predicting the companies financial failure three, two and one year in advance with respect to the accuracy, respectively, (93% / 86.4%), (88% / 84%) and (94% / 84%). Later, More recently, several researchers have compared the statistical techniques with ML techniques on forecasting companies' financial failure. Also, Islam et al.[54] compared 13 classification models regarding predicting bankruptcy status using extremely imbalanced dataset. SMOTE was utilized to resample the data as a preprocessing stage, showing an improvement on the performance of the classification algorithms according the evaluation metrics.

Most recently, Bateni and Asghari [55] compared the performance of using Logit to a genetic algorithm with respect to predict the financial failure of Iranian companies. A balanced dataset comprising of 174 records belong to solvent Iranian companies and other 174 records belong to the bankrupt companies in the same market in the time period of 2006–2014 had been used to train both methods. Nevertheless, the genetic algorithm model significantly outperformed the logit model in predicting the financial status of the Iranian companies with a huge gap in the obtained accuracy.

2.4 COMBINED METHODS

In an attempt to provide robust classification model outperforming the existing standard ones, several researchers combined classification methods, and use those combinations to predict the financial status of companies. For instance, Hu et al. [56] proposed three combinations of methods to predict the financial failure of Korean companies. The proposed models were MDA + neural networks, decision tree + neural networks, and Self Organizing Feature Map (SOFM) + neural networks. The authors reported that the hybrid models of neural networks yielded promised performance in predicting companies financial status. Later, Min et al. [57] devoted the genetic algorithm to improve the performance of SVM in predicting Korean companies financial status. Mainly, the genetic algorithms had been used to improve the feature subset and the SVM method parameters. The new method showed an optimum performance compared to stand alone SVM and Logit. Also, Wu et al. [58] developed genetic-based SVM to predict the financial distress in Taiwanese companies. The performance of the new model compared to two artificial methods (i.e., SVM and neural networks) and three statistical methods (i.e., MDA, Logit and Propit). Thus, the genetic-based SVM showed better performance in predicting and mispredicting the companies financial failure compared to the other methods. Hu [59], proposed a novel multilayer perceptron + Choquet fuzzy integral and applied it in order to analyse financial distress . The main idea of the proposed model is replacing the activation function of the traditional MLP by the Choquet fuzzy integral. In addition, Sun and Li [60] considered combining several statistical and artificial intelligence classifiers using weighted majority voting, namely: MDA, Logit, decision tree, neural network and SVM, and reported that using the combined model shows better performance that using sand alone classifiers. In addition, Chandra et al. [61] combined several powerful classifiers (i.e., RF, SVM, MLP, Logit and CART) to predict the financial failure of dotcom companies.

Most recently, Ghatasheh et al. [62] combined ensemble methods with cost-sensitive methods in order to predict companies financial status. They compared the performance of combining 3 cost sensitive methods, ie., cost-sensitive learning, cost-sensitive classification and MetaCost with several ensemble classifiers. The combination of RF with cost-sensitive classification method outperformed the other ensemble and cost-sensitive methods addressed in that study, the accuracy, type I and type II errors obtained by that combination were, respectively, 91.172%, 9.4% and 8.8%. Also, Faris et al. [63] proposed a hybrid approach to predict companies' financial status, that approach based mainly on several stages, ie., data normalization, resampling, feature selection and the final stage is classification. In their study, they compared the performance of using basic and ensemble classifiers. They found that the combination of SMOTE and Adaboost ensemble methods utilizing reduced error pruning tree guaranteed promising results compared with the other basic and ensemble classifiers. The accuracy, type I and type II errors obtained by that approach were, respectively, 98.3%, 0.6% and 45%.

3 METHODOLOGY

CONTENTS

This chapter presents in details the standard and ensemble machine learning algorithm used along this thesis in order to predict companies financial failure (Section 3.1) . As well as, the Deep Learning algorithm that have been customized to solve the same problem in the optimum way (Section 3.1.3).

3.1 MACHINE LEARNING ALGORITHMS

In this section, the Standard, ensemble machine learning and deep learning algorithms used in this study are presented in details.

3.1.1 *Standard Classifiers*

Several standard (classical) classification algorithms shows a promising performance in forecasting some events in the future. Thus, This subsection presents in detail the standard (classical) classifiers that have been utilized along this thesis to predict companies financial failure.

3.1.1.1 *Decision Trees Classifier (DT)*

Is a basic iterative model used normally for classification and regression tasks, and also it is the base algorithm for several machine learning algorithms[64, 65]. DT comprised mainly of three components, namely:

1. Root Node: which represent the data input point; it doesn't have an input edge.

2. Test Nodes: which split the attribute space into two or more sub-spaces according to the possibility (probability) of the attributes values.

3. Leaf Nodes: That consist of the objects that belong to the same class.

Accordingly, most of the decision trees are made up of two essential procedures [65]:

- Growth procedure: which is a recursive partitioning of the data during building the decision trees till constructing a tree contains leaves nodes represent a single class, or a values less or more than certain threshold.

- Pruning procedure: the steps of this procedure referred generally as *pre-pruning*, which normally applied during the growth procedure in order to avoid the training data over-fitting. Thus, avoiding the over-fitting situation could happen by preventing the splits that incompatible with a certain threshold such as minimum number of instances for each split, or minimum number of labels for each leaf node.

Moreover, during constructing the decision trees, several iterations should be addressed in order to build the ideal decision tree that leads to the optimum solution, in each iteration the attribute space split based on the output of discrete functions. Thus, after evaluate the discrete function the dataset split into two (or more) subsets (i.e, sub-trees) in each iteration; each subset belongs to one class (pure), or to two or more classes (impure) [65]. Accordingly, to present a simple example of decision tree construction, small sample of the nominal attributes has been selected randomly from the Spanish companies' dataset. In other words, as it shown in Table 3.1, 10 instances selected from the original dataset, each instance comprised of six nominal attributes, namely: *Social code* (LTD, Co,or Other), *Linked Group* (Yes, or No), *Delay Account* (Yes, or No), *Audited* (Yes, or No) and *Auditors Opinion* (positive, negative, minor, or nothing), and the class attribute *'Bankrupt'* (Yes, or No). Figure 3.1 illustrates the decision tree that constructed based on the selected sample. Thus, as it can be seen in the figure, one of the possible attributes values series that could lead to the *solvency* case (i.e, class label= No) is: *Social code* = Co, *Linked Group*= No, *Delay Account*= Yes, and *Auditors Opinion*= Minor.

Moreover, several splitting measures could be used to construct the decision trees, each measure might fit certain problems or

Table 3.1: Random small subset selected from the Spanish companies' dataset nominal attributes.

Social Code	Linked Group	Delay Account	Audited	Auditors Opinion	Bankrupt
LTD	Yes	Yes	Yes	Positive	No
Co	No	Yes	Yes	Minor	No
Co	Yes	No	Yes	Positive	No
LTD	Yes	No	No	Nothing	No
Co	No	Yes	Yes	Negative	Yes
Other	No	No	Yes	Negative	Yes
LTD	Yes	Yes	Yes	Negative	Yes
Other	Yes	Yes	Yes	Negative	Yes
Co	No	No	Yes	Positive	No
Co	Yes	No	Yes	Minor	No
LTD	Yes	No	Yes	Minor	No

dataset better than the others [64, 65]. Specificity, in our work we have used two well-known splitting measures, which are:

- *Information gain*: it is a measure of the impurity in several instances based on the *entropy* criterion. It discover the usefulness of each feature for classification; the feature with more information gain is more useful to split during the decision tree construction. Accordingly, the *entropy* measures the degree of the impurity. In other words, if the splitted subset's instances belong to more than one class, then the subset is impure and the *entropy* value would be more than zero, but if all of the instances belong to the same class, then the *entropy* value would be zero. Equations 3.1 and 3.2 present the *Information gain* and *entropy* formulas, respectively [66].

$$Information\,Gain(L, f) = Entropy(L) - \sum_{v=1}^{V} \frac{|L^V|}{|L|} \,.\, Entropy(L^V) \tag{3.1}$$

$$Entropy(L) = -\sum_{i=1}^{j} p_i \,.\, \log_2(p_i) \tag{3.2}$$

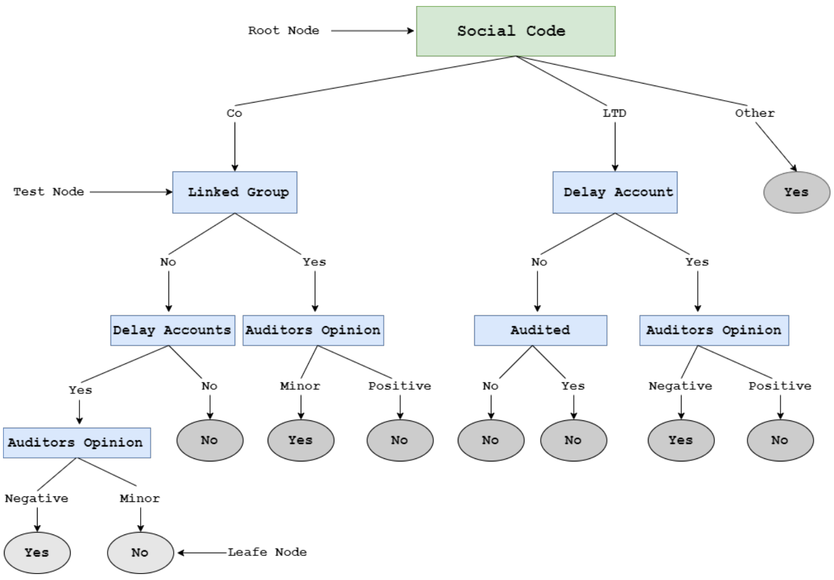

Figure 3.1: Simple decision tree constructed using the random subset that selected from the Spanish companies' dataset nominal attributes.

where L is the splitted subset using the feature f, V represent amount of the various values of the feature f, $|L^V|$ is the subset of L with $f = v$, and p is the probability of each observation value or label.

- *Gini index*: is an impurity-based criterion; better split yields more pure data. Equation 3.3 defines the Gini formula [66].

$$Gini(L) = 1 - \sum_{i=1}^{j} p_i^2 \tag{3.3}$$

Where L is a dataset, j is the amount of the various class values, p_i is the relative frequency if class i in L. Thus, if the dataset split into 2 subsets (i.e, $L1$ and $L2$) with 2 different sizes (i.e, $N1$ and $N2$) based on the attribute A values, Gini founded as mentioned in the equation 3.4 [66].

$$Gini_A(L) = \frac{N_1}{N} \,.\, Gini(L_1) + \frac{N_2}{N} \,.\, Gini(L_2) \tag{3.4}$$

Furthermore, the reduction in the impurity is founded as mentioned in the Equation 3.5[66].

$$\Delta Gini(A) = Gini(L) - Gini_A(L) \tag{3.5}$$

3.1.1.2 *C4.5 Classifier*

C4.5 is very common classifier proposed by Ross Quinlan in 1993 [67], and based on ID3 algorithm [68] (proposed by the same author in 1986) with enhancements on handling missing values and continuous attribute value ranges with the ability to choose an appropriate attribute selection measure. ID3 is a very simple decision tree constructed by splitting the training data relying on the *Information Gain* splitting criteria, thus, the growth procedure remains until all of the instances have the same value, or the the information gain values not exceeding zero. Moreover, ID3 algorithm is not processing numeric values (only categorical or discrete), it doesn't handle the messing values normally, and also it is sensitive to the features with large number of different values. Accordingly, C4.5 proposed to overcome these limitations, it process both numerical and categorical values, handle the messing values, and also eliminate the sensitivity of processing the features with large amount of different values by using *Gain Ratio* splitting criterion instead of the *Information Gain* [69]. Equation 3.6 presents the *Gain Ratio* splitting criterion formula [64].

$$Gain\,Ratio(a_i, S) = \frac{Information\,Gain(a_i, S)}{Entropy(a_i, S)}, \tag{3.6}$$

where S is a training set, and a_i is an attribute value. Moreover, C4.5 provide the the pruning procedure with the dept-first constructing strategy. For each nominal attribute each value used to generate several sub-trees as many as the number of the different values in that attribute. On the other hand, the numeric values of the attributes are sorted, and then the *Gain Ratio* calculated based on for each split that based on each fixed unique value in the sorted values [69].

3.1.1.3 *J48 Classifier*

is a Java implementation of the C4.5 algorithm in the WEKA data mining tool [70], which is an extension of ID3 decision tree algorithm. The main objective of J48 classifier is to implement the training dataset into a decision tree based on the number of attributes in it. While J48 classifier creates the decision tree it ignores all of the missing values because they are valueless. The procedure of J48 predicting stands on the known attributes values [70], and handle the discrete and the contiguous data, then pruning the decision tree if there are some branches which do not help in order to reach the leaf node [71]. J48 algorithm runs according to particular steps:

- The first step is if all of the records in the dataset belong to the same class, so the tree is a leaf, this leaf will be labeled with the same class.

- The second step is calculating the information of the attributes given by applying a test on it depending on the probability of the attribute value in each record, then the information Gain calculation relaying on the information given by applying the tests.

- The last step is to select the best attributes regarding to the information gain calculated in the previous step.

The final touch on the decision tree after the full creation of it, and before performing the classification, is to remove a discordant information, which is far away from the majority of data and adversely affect data classification. This process called pruning, it is very important to improve the accuracy of the prediction while many datasets may contain this type of unuseful data [71].

3.1.1.4 *Naïve Bayes Classifier*

Is a probabilistic method used to assign the class of each record relaying on calculating the probability of each attribute independently from the other attributes [72]. In other words, Naïve Bayes assume that the effect of each attribute value is detached from other attributes values on predicting the class. This classification method presents high accuracy and efficiency when applied on large training set by calculating the frequencies and combining the values to make a good decision about the predicted class for each records. Furthermore, Naïve Bayes classifier is easy to implement, as well as follows simple processing procedure; no complicated iterative parameter scheme, thus, it is strongly recommended to process big data [73]. For more details about the Naïve Bayes classifier procedure, suppose that we have n attributes $E = (x_1, x_2, x_3, ..., x_n)$, where x_i is the value of X_i, C is the classification variable, c is the variable of C. In addition, in the consideration of that we have binary class variables (*True*, *False*), and relying on the Bayesian theorem, the probability of $E = (x_1, x_2, x_3, ..., x_n)$ is defined in Equation 3.7 [74].

$$p(c|E) = \frac{p(c|E)\ .\ p(c)}{p(E)} \tag{3.7}$$

Thus, E is classified as c in the case of

$$f_b(E) = \frac{p(c = True|E)}{p(c = False|E)} \tag{3.8}$$

where $f_b(E)$ is the Bayesian classifier. Moreover, assuming that each attribute independent according the value of class attribute, then the probability of E given that the class is c defined as in the Equation 3.9 [74].

$$f_b(E|c) = p(x_1, x_2, x_3, ..., x_n|c) = \prod_{i=1}^{n} p(x_i|c) \tag{3.9}$$

Thus, the output classifier is defined in the Equation 3.10.

$$f_{nb}(E) = \frac{p(c = True)}{p(c = False)} \prod_{i=1}^{n} \frac{p(x_i|c = True)}{p(x_i|c = False)} \tag{3.10}$$

The function $f_{nb}(E)$ is called the Naïve Bayes classifier. Figure 3.2 illustrates the structure of a simple Naïve Bayes classifier [74].

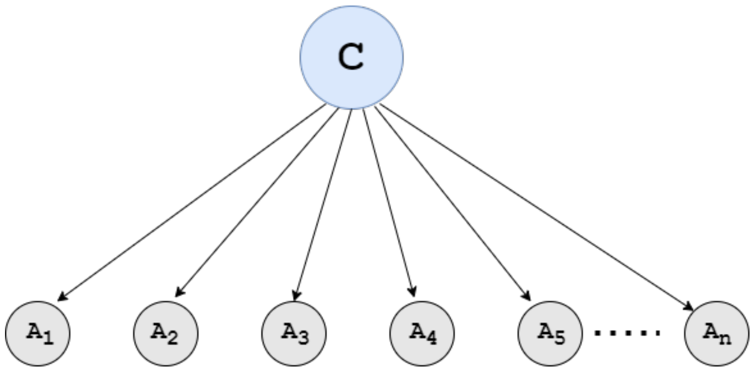

Figure 3.2: The structure of Naïve Bayes classifier.

3.1.2 *Ensemble Classifiers*

Actually, the ensemble classifiers developed as a further step from the standard one. They shows a promising performance compared to the standard ones. Thus, this section presents the ensemble-based classifiers used along this thesis to predict companies financial failure.

3.1.2.1 *Random Forest Classifier*

Random forest (RF) is very powerful ensemble classification method developed by Breiman [75] in 2001. It is based on the creation of different decision trees from different subsets of the original dataset. Training these subsets creates several decision trees constructing the random forest, each instance's class in the test set predicts independently in each decision tree. The final results of the instances class relay on the majority voting of these decision trees. As an initial step in Random forest classifier bagging that based on *bootstrapping* method is used to create these datasets and then split the original dataset into training and test set by taking a partition of data as a training and the remaining is the test set. In other words, some instances will be selected from the original dataset by sampling and replacement method to be a training dataset and the remaining instances that defined as Out-Of-Bag considered to be a test set. Creating a decision tree for each subset basically depend on C4.5 algorithm which is a benchmark decision tree algorithm that stands mainly on entropy and gained values, but in the random forest the *Gini Index* is the splitting criterion that used to split the data. The last mission of Random Forest classifier is to gather the subtrees with each other to create the forest, the classification result is the average of class probabilities obtained from all the training trees [76].

Figure 3.3 describes the architecture of RF model.

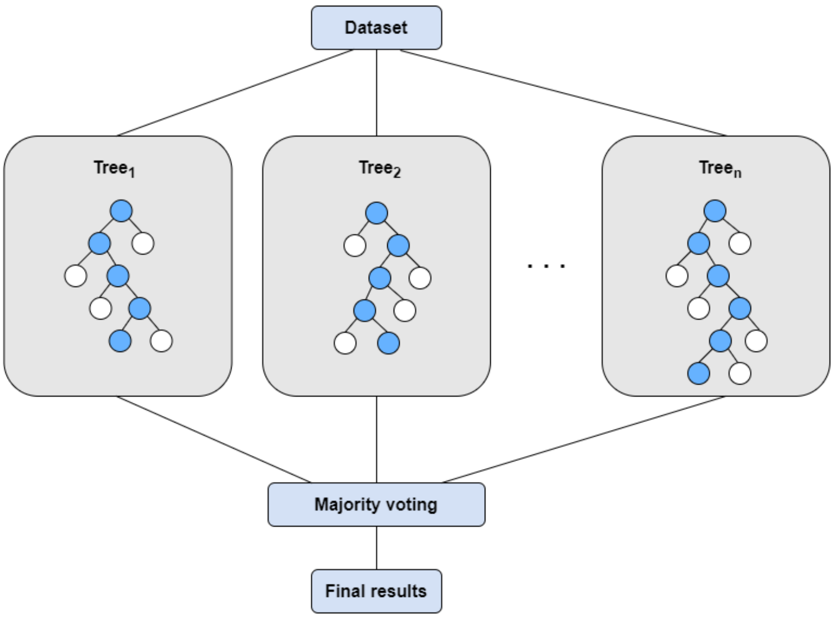

Figure 3.3: The illustration of RF classifier structure.

3.1.2.2 *SVM Classifier*

SVM is one of the most popular supervised ML algorithms proposed mainly for binary classification and regression problems by Cortes and Vapnik in 1995 [77]. It can be used in several different application such as Data mining, Image processing, Speech processing, Time-series prediction, Automotive, Security, Bioinformatics, and Power systems [78]. Basically, as shown in Figure 3.4, it finds the proper separating hyperplane that maximizes the margin in the features space between the two classes. Thus, for training the data that belong to two classes, given that the training data comprised of n instances $(x_i, y_i), i = 1, 2, 3, .., n$ and $x_i = (x_i^{(1)}, x_i^{(2)}, x_i^{(3)}, ..., x_i^{(n)})^T \in R^n$, whereas $y = 1\ or\ -1$, thus fining the optimal hyperplane can be done by applying the formula mentioned in Equation 3.11 [78–80]. Figure 3.5 shows the optimal hyperplane that separates the data while the data belong to two classes and linearly separable.

$$w\ .\ x\ + b\ =\ 0 \tag{3.11}$$

where w in the weight victor, and b is the bias victor. Accordingly, the optimal classification function could be reached by the constrained minimization shown in Equation 3.12 [79, 80]..

$$\begin{aligned} &E(W) = \frac{1}{2} \cdot ||W^2|| \\ &Subject\ to \\ &y_i[W\ .\ X_i + b] \geq 1,\ i = 1, 2, 3, ..., n. \end{aligned} \tag{3.12}$$

Moreover, Equation 3.13 represent the SVM classification function [79, 80].

$$D(x) = \sum_{i=1}^{n} y_i\ \alpha_i\ k\ (X, X_i) \tag{3.13}$$

Where α_i are all automatically determined as a result of quadratic optimization. In addition, the Equation 3.14 represent the formula of finding the optimal hyperplane from a linear SVM model [79].

$$D(x) = sign(\sum_{i=1}^{n} y_i \alpha_i X.X_i + b) \tag{3.14}$$

On the other hand, it is not mandatory that the data is linearly separable (see Figure 3.6), thus, to avoid some complex calculations (i.e, the case illustrated in Figure 3.7) , kernel functions (such as Linear, Polynomial, Sigmoid and Gaussian Radial Basis function (RBF)) are used as a hyperparameter aiming to allocate the separating hyperplanes [81].

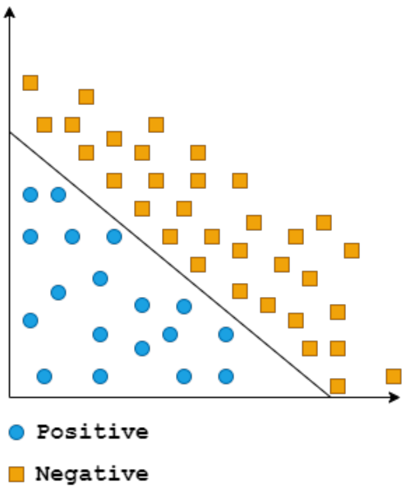

Figure 3.4: Example of linearly separable data.

Equation 3.15 shows the formula of RBF kernel function; which is the most frequently used function due to its effectiveness.

$$k(x,y) = exp\left(- \frac{|x-y|^2}{2\sigma^2} \right) \tag{3.15}$$

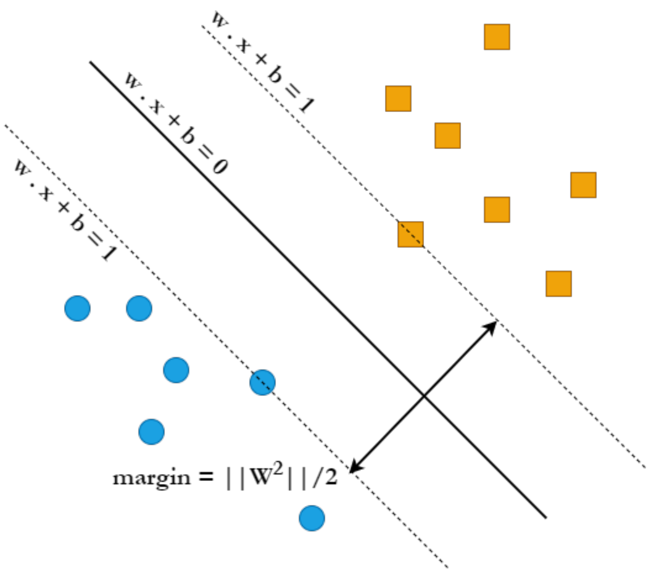

Figure 3.5: The hyperplane separating the data in the case of Linearly separable data.

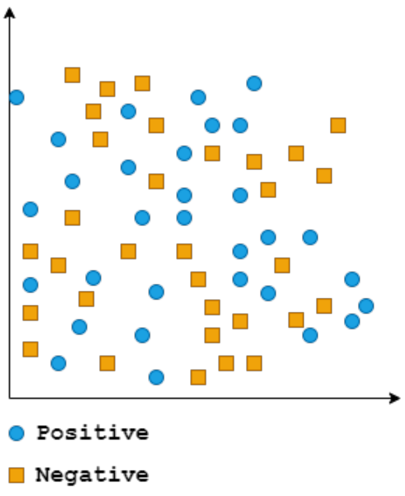

Figure 3.6: Example of linearly inseparable data.

Thus, Equation 3.16 shows the corresponding decision function [79].

$$D(x) = sign\left(\sum_{i=1}^{n} \alpha_i \, exp\left(-\frac{|X - X_i|^2}{2\sigma^2}\right) + b\right) \tag{3.16}$$

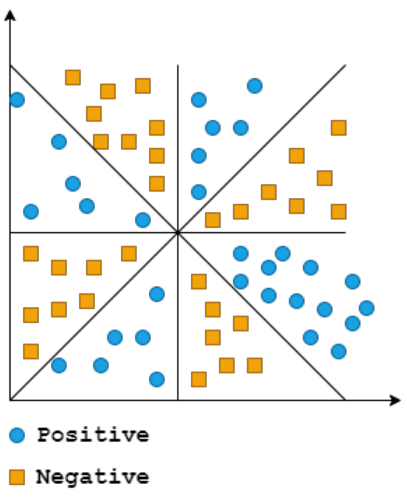

Figure 3.7: Example of several calculated hyperplanes to separate the data (complex calculations).

In addition, to improve the classification performance, the SVM has considered in this study as an ensemble model applying *bagging* [82]. This uses the *bootstrapping* method in order to create several subsets from the original dataset, and implements the model several times independently and aggregating the ensemble model's final results using majority voting. Figure 3.8 shows the bagging technique procedure.

3.1.2.3 *KNN Classifier*

KNN is another widely used non-parametric ML algorithm proposed by Cover and Hart in 1967 [83]. It decides about the class label of each sample according to the similarity (distance) with its closest neighbors. Thus, the distances between each instance and the rest of instances are calculated using several distance measures. In addition, K is a parameter associated with the KNN method, representing the number of nearest neighbors to select for each sample in the dataset. Selecting the K parameter value is a bit tricky, depending on the dataset, low K value would increases the impact of the noise on the classifier's results which

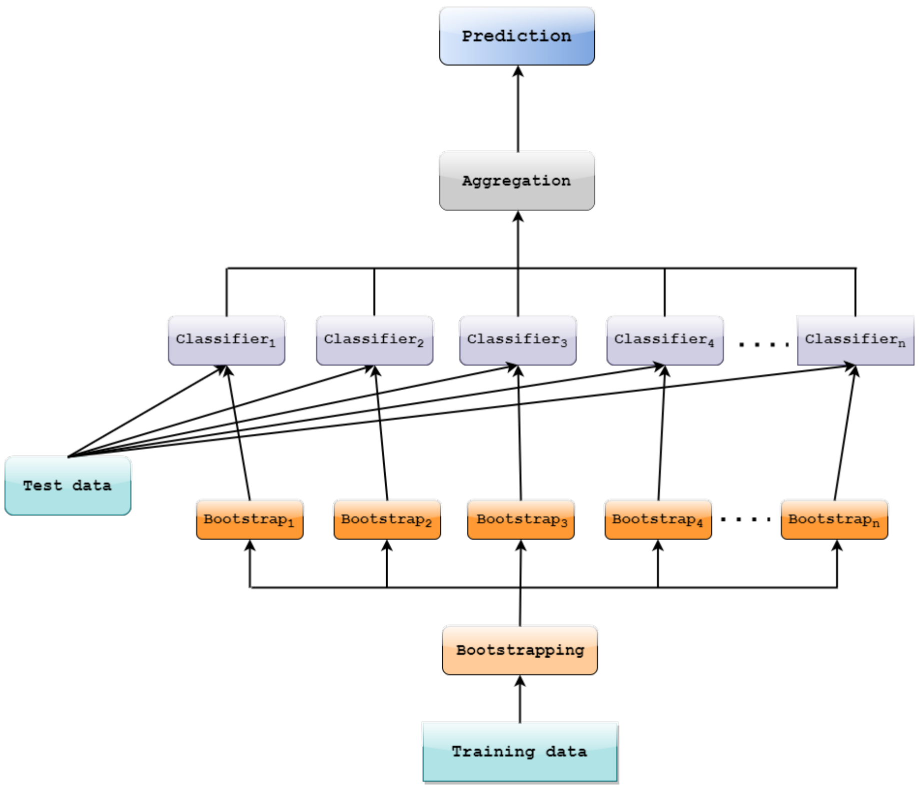

Figure 3.8: The procedure of bagging technique.

affects the decision's accuracy, whereas not reasonable high K value increases the computation time and the memory usage. Moreover, to avoid any confusion during making the decisions (predictions), K parameter value set normally to an odd number; to avoid having any chance of equality between the nearest instances that belong to the different classes (e.g., if $K = 6$, and three of the neighbors belong to the class label 'Yes', and the remaining three neighbors belong to the class label 'No'). Thus, according to the K value and the majority class of the nearest neighbors, the test instances class set as the majority class. For example, consider we have some instances of companies, and each instance belongs to one out of two classes (i.e, Bankrupt and Solvent), whereas K set to three ($K = 3$), then if two or more of the neighbor instances belong to 'Bankrupt' class, the test instance will be set to the majority class label which is 'Bankrupt'.

Accordingly, Figure 3.9 illustrates several instances of companies belong to two classes.

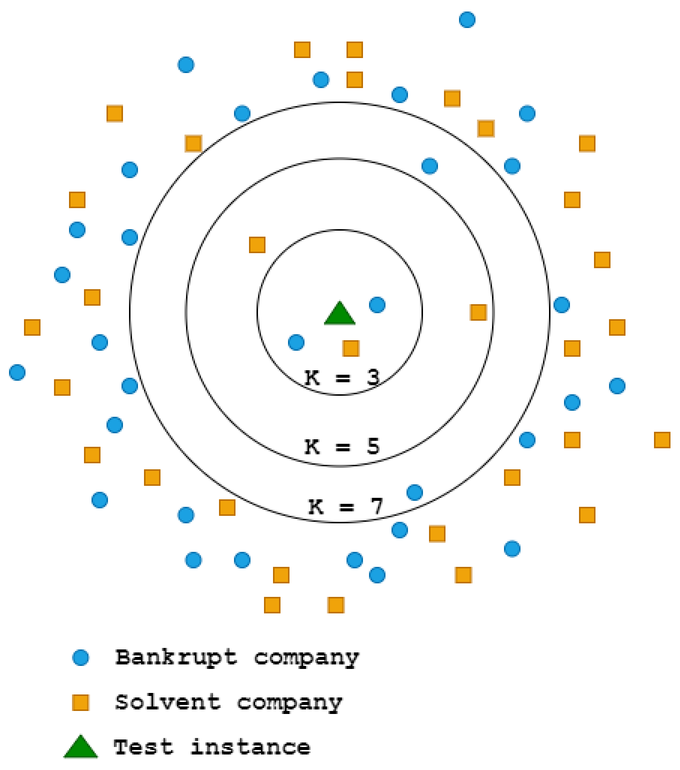

Figure 3.9: The nearest neighbors instances to the test instances; each circle contains K neighbors.

Thus, as it can be seen in the figure, if K preset to three, the test instance would set to 'Bankrupt' because it has two neighbors belong to 'Bankrupt' class, and one instance belongs to 'Solvent' class. Moreover, if K preset to five, the test instance will assigned with the 'Solvent' class because it has three neighbors belong to 'Solvent' class and the remaining two instances belong to the other class. Furthermore, if $K = 7$, the test instance will be assigned with the 'Bankrupt' class because it has four instances belong the 'Bankrupt' class, whereas the remaining three instances belong to the other class. The firm conclusion that we can extract from this example is that the value of the parameter K has a major impact on the classification results. Besides, KNN is a powerful and robust classifier, but in the expense of the computation time and memory; it calculate the distances between each test instance with each instance in the training dataset and store it in the memory during the classification stage.

Basically, the KNN classification procedure comprised of two stages, the first one is the *training stage*, in which the training data is stored with no missing or categorical values. The second on is the *classification stage*, in which the distance between each test instance and all instances in the stored dataset are calculated and stored in the memory, then according to the K parameter value, the nearest neighbors will be selected. Finally, the test instance will be assigned to the majority class in the selected nearest neighbors [84].

Several distance algorithms could be used to measure the similarity (distance) between samples such as, e.g., Euclidean, Mahalanobis, Minkowski and Hamming [85]. Minkowski distance is a generalization form of some other distance metrics, it is a metric that measures the similarity (distance) between two points in the data according to the following formula:

$$D(x,y) = \left(\sum_{i=1}^{n} |x_i - y_i|^{\frac{1}{p}} \right)^{p}, \tag{3.17}$$

where x_i and y_i are data points, and p is an integer. The value of p could make Minkowski metric equal to other metrics, if $p = 1$ (City block distance), $p = 2$ (Euclidean distance) and $p = \infty$ (Chebyshev distance) [86]. On the other hand, as in the previous method, KNN has been used in this study as an ensemble model using the *bagging* method.

3.1.2.4 *AdaBoost Classifier*

AdaBoost is an iterative ensemble ML algorithm that applies its base classifiers in a sequence based on *boosting*, a technique that combines a set of 'weak' learners applied sequentially for developing a 'strong' learner [87]. In other words, AdaBoost applies the base classifier (normally a *Decision Tree*) several times iteratively. In the first iteration of the model, the weights are set equally to all samples, and in the remaining iterations, the weights increase for the misclassified samples and decrease for the correctly classified samples in the previous iterations in order to improve the overall performance of the model. The final results are the combination of the ensemble classifiers predictions using weighted majority voting [88]. Figure 3.10 illustrates the structure of the AdaBoost classification algorithm.

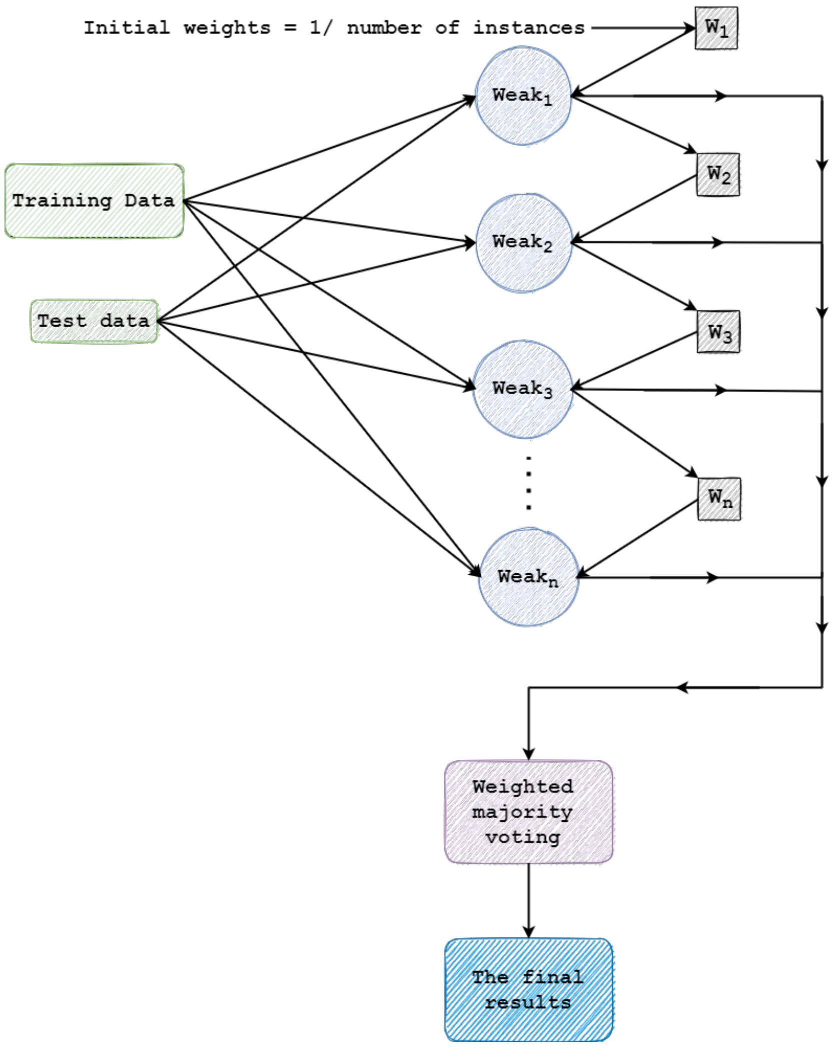

Figure 3.10: The structure of the AdaBoost classifier.

Accordingly, consider that we have training dataset contains n instances $(x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_n, y_n)$, while x_i is in domain $X(x_i \in X)$, while y belongs to two classes $(y \in Y = -1, 1)$. As aforementioned, the AdaBoost algorithm is based mainly on repeating applying the 'weak' learners iteratively T times.

The procedure of AdaBoost algorithm is described in the following steps [89, 90]:

- In the first iteration, the weak learner process the training data normally with weights set initially to $1/n$ for all instances.

- Starting from the second iteration $(t = 2, 3, 4, .., T)$, the weights are updated depending on the weak classifier prediction in the previous iteration; increasing for the misclassified instances and decreased for the correctly classified instances, in order to force the classifier to focus on the errors and try to minimize it in the next iterations.

- The weights are recorded as D_t, and for each stored D_t the weak classifier find the weak hypothesis $h_t = X \to (-1, 1)$.

- The error rate computed in each iteration as stated in the following equation: $h_t = \varepsilon_t = \sum_{i=1}^{n} D_t(x_i)[h_t(x_i) \neq y_i]$.

- The new weights for the weak learners is computed relying on the error rate founded in the previous step as stated in the following equation: $a_t = 1/2\left(ln\Big((1 - \varepsilon_t/)\varepsilon_t\Big)\right)$

- Update each sample weight:
  $D_{t+1}(x_i) = \Big(D_t(x_i)/Z_t\Big)exp\Big(-a_t y_i h_t(x_i)\Big)$, where Z_t is the factor to meet the probability of each instance.

- The last step is to combine the T weak learners to construct a strong classifier and obtain the final results H:
  $H(x) = sign\left(\sum_{t=1}^{T} a_t h_t(x)\right)$

3.1.2.5 XGBoost Classifier

XGBoost is a relatively new ensemble *boosting* ML method proposed by Chen and Guestrin in 2016[91]. The procedure of XGBoost is based mainly on Gradient Boosting but with further steps in order to improve the performance of predictions regarding the computation speed, generalization and scalability by controlling the overfitting using regularization. The base learner used in XGBoost is Classification And Regression Trees (CART). The final result is the sum of the CARTs' scores.

The general model for estimating the XGBoost function is shown in Equation 3.18 [92].

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_t^{(t-1)} + f_t(x_i) \tag{3.18}$$

Where $\hat{y}_i^{(t)}$ and $f_t(x_i)$ is the prediction and the weak learner at step t, respectively, and x_i is the input variable. Moreover, using the regularization could affect the data fitting to the model by reducing the preprocessing step during the additive expansion. equation 3.19 shows the formula devoted to eliminate the overfitting [92].

$$\begin{gathered} F_{obj}(\theta) = \sum_{k=1}^{n} L(\theta) + \sum_{k=1}^{t} \Omega(\theta) \\ given\ that: \\ L(\theta) = L(\hat{y}_i, y_i) \\ \Omega(\theta) = \gamma T + \frac{1}{2}(\lambda ||w||^2) \end{gathered} \tag{3.19}$$

Where $L(\theta)$ is the loss function that devoted to find the difference between the actual value (y_i) and the prediction ($\hat{y}_i$), $\Omega(\theta)$ is a regularized term that penalize the complex model, γ is the minimum loss needed to further partitioning for the leaf nodes, T is the number of leaves in the CART tree, Ω is a regulized parameter to scale penalty, γT is the tree pruning spanning, and w is the weights for the leaves.

Furthermore, the second order expansion addressed in the Equation 3.20, could be used to in order to apply other loss functions[92].

$$J^{(t)} \approx \sum_{i=1}^{n}\left(g_i w_{q(x_i)} + \frac{1}{2}\left(h_i w_{q(x_i)}^2\right)\right) + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{3.20}$$

Where g_i and g_i is the first and the second derivative of the loss function, respectively.

Moreover, the formula mentioned in Equation 3.21 used to obtain the weights for the leaf node j.

$$w_j = \frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{3.21}$$

In addition, the formula motioned in Equation 3.22 used to calculate the loss value for each leaf node.

$$J^{(t)} \approx \sum_{j=1}^{T}\left(\left(\sum_{i \in I_j} g_i\right) w_j + \frac{1}{2}\left(\sum_{i \in I_j} h_i + \lambda\right)\left(w_j^2\right)\right) + \gamma T \tag{3.22}$$

Where I_j are all instances in the leaf node j [92].

3.1.3 *Deep learning Algorithms*

Deep Leaning algorithms are neural networks with two or more hidden layers used widely to solve real-world complex problems. In addition, they show very high performance compared to the classical machine learning algorithms. This section presents the Deep Learning algorithms adopted along this doctoral thesis to predict companies' financial failure.

3.1.3.1 *Deep Belief Networks (DBN)*

DBN is a stochastic DL method proposed by Hinton et al.[93] in 2006, consisting of several-stacked Restricted Boltzmann Machines (RBM). RBM is an energy-based generative model composed of two layers, visible units and hidden units, where all units are fully bidirectional connected with symmetric weights between layers. As shown in Figure 3.11, a DBN consists of several stacked RBMs, where the hidden layer of the lower RBM represents the visible layer of the upper RBM, the links between the top two layers are undirected and the links between the remaining layers are directed. In addition, DBN trains greedily; each RBM trains unsupervised on a time. Therefore, the results of each RBM represent the input of the higher RBM, and the final results are fine-tuned with supervised learning.

Every hidden layer is modeled as h^i a binary random vector with elements h^i_j. Thus, Equations 3.23 and 3.24 parameterize the whole DBN model and each hidden layer probabilities, respectively,

$$P(V, h^1, ...h^\ell) = P(V|h^1)P(h^1|h^2)..P(h^\ell - 1|h^\ell) \tag{3.23}$$

$$P(h^i|h^{i+1}) = \prod_{j=1}^{n_i} P(h^i_j|h^{i+1}) \tag{3.24}$$

Also, with h_j^i as a stochastic units and 1 as binary activation, Equation 3.25 represents the probability of each stochastic hidden unit.

$$P(h_j^i = 1|h^{i+1}) = sigm\left(b_j^i + \sum_{k=1}^{n_{i+1}} W_{jk}^i h_k^{i+1}\right) \tag{3.25}$$

$$sigm(x) = 1/(1 + exp(-x)) \tag{3.26}$$

The normal sigmoid (Equation 3.26) represents the activation function, b_j^i are the biases, W^i is the weights matrix. Each RBM follows equation 3.24 and equation 3.25 respectively in order; upward from bottom to top [94].

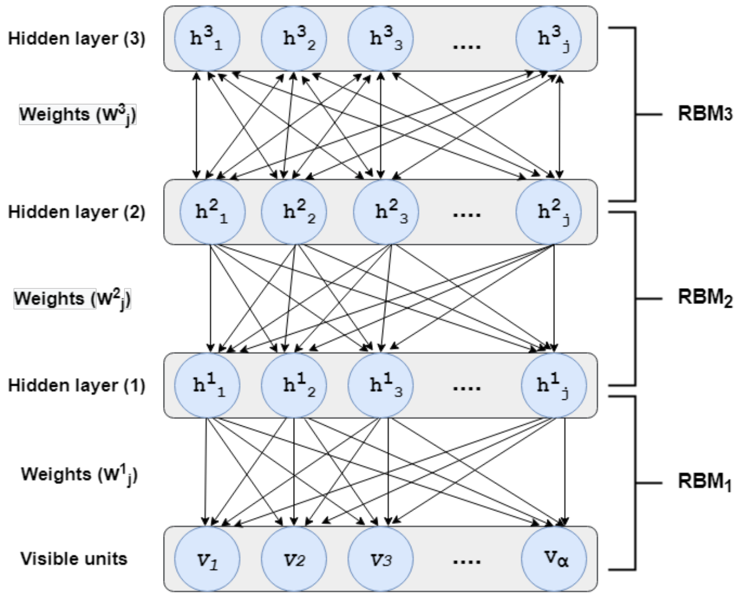

Figure 3.11: The illustration of DBN model with 3 hidden layers structure.

3.1.3.2 *Long-Short Term Memory (LSTM)*

LSTM is a specific type of Recurrent Neural Networks (RNN) that was proposed by Hochreiter and Schmidhuber in 1997 [95]. The essential unit of LSTM is a cell that replaces the hidden layer neurons of the RNN, and each cell is configured mainly by three gates: input gate, output gate and forget gate as shown in Figure 3.12.

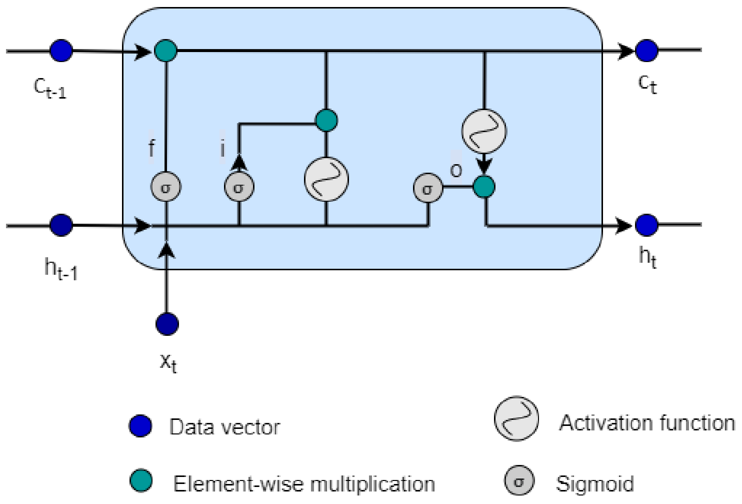

Figure 3.12: The Illustration of general LSTM memory cell [96].

LSTM architecture gives it the possibility to make a decision whether to forget or update the last hidden status with new information.

The following six equations describe the information processing steps of LSTM [96]:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.27}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3.28}$$

$$\tilde{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{3.29}$$

$$c_t = i_t \bigodot \tilde{c}_t + f_t \bigodot c_{t-1} \tag{3.30}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t + b_o) \tag{3.31}$$

$$h_t = o_t \bigodot tanh(c_t) \tag{3.32}$$

where t represents the time unit, f_t is the forget gate, i_t is the input gate, o_t is the output gate, W_* and U_* are the weight matrices, b_* are bias vectors, x_t is an input vector. Also, c_t represents the memory status vector, and h_t is the hidden status vector output obtained from c_t. in addition, σ is the sigmoid function, $\tilde{c}$ is the input modulation, h_t is the output and $\bigodot$ is a point-wise multiplication.

Therefore, in the first step of the LSTM process (Equation 3.27), the sigmoid function identifies the information that will be discarded according to its value. This step represents the forget gate procedure. The second step (Equations 3.28, 3.29 and 3.30) consists on placing the decision to update the information from the input according the sigmoid and *tanh* functions values. In other words, sigmoid makes the decision to update the information or discard the update, *tanh* obtain the value of weights, then the new cell state set by multiplying the values of the sigmoid and the *tanh*. In the final step (Equations 3.31 and 3.32) the output will be obtained by relying on the filtered version of the cell state.

3.1.3.3 *Multilayer Perception model of 6 layers (MLP-6L)*

MLP is a feed-forward neural network, usually applied on supervised learning tasks, based on back-propagation learning [97]. It consists of a neural network with input, output and one or more parallel hidden layers. The architecture of the MLP is described as a fully interconnected network, as shown in Figure 3.13. However, increasing the number of the hidden layers transforms the MLP from classical learning method into a DL method [98]. In this study, a MLP model with four hidden layers has been used, so, there are six layers in total and thus, we refer to it as MLP-6L.

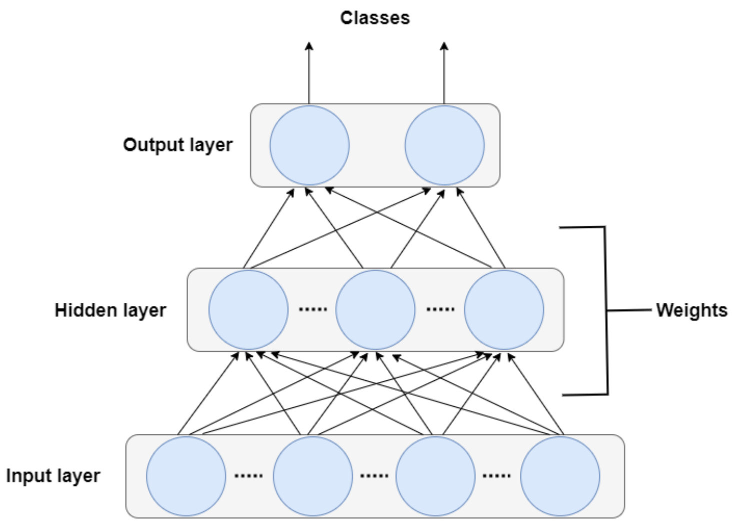

Figure 3.13: The illustration of basic three layers MLP.

Each processing unit on each layer is connected with the whole units in the following layer by weighted connections[97, 99]. Also, the input values represent the information fed forward into the network. The processing of the information in the hidden units depends on the input information and the weight value of each input-hidden unit connection. Accordingly, the information obtained by the output units depends on the values of the hidden units and the weight value of each hidden-output units connection [99]. The MLP training is developed gradually in several stages: in each stage, the output units obtained results are compared with the real data allocated in the training data, then an error signal is used to enhance the MLP expectation in the further stages in order to achieve almost identical values gradually

[99]. The activation function that obtains the MLP output(h_j) is described in Equation 3.33[97],

$$h_j = f\left(\sum_i x_i w_{ij} + b\right), \tag{3.33}$$

where $f()$ is the activation function, x_i is the activation of ith hidden layer unit, w_{ij} is the weight of the connection joining the jth neuron in a layer with the ith neuron in the previous layer, and b is the bias for the neuron.

Also, Equation 3.34 describes the error function that can be reduced by enhancing layers interconnection

$$E = \frac{1}{2}\sum_n \sum_k (t_k^n - h_k^n)^2, \tag{3.34}$$

Where t_k^n is the calculated output, h_k^n is the actual output value, n is the number of sample and k is the number of output units.

4 STUDY ON CLASSICAL CLASSIFICATION METHODS WITH NEW SIMPLE RESAMPLING APPROACHES TO PREDICT COMPANIES' FINANCIAL FAILURE

CONTENTS

This chapter presents a study addressed in a manuscript published in an international conference; "Predicting the Financial Status of Companies using Data Balancing and Classification Methods [100]". Accordingly, this study provides a significant solution for common drawbacks that occurs during dealing with data that has extremely inconsistent distribution. The main problems that might happen in this case are wasting important information by eliminating a big amount of the majority instances, and the overfitting problem that happens by replicating the minority instances to achieve a balanced state to the dataset.

4.1 BACKGROUND

The relevance of bankruptcy prediction problem is evident in today's world due to its effects on banks, businesses, and companies. There might be huge financial losses encountered due to bad judgment and analysis. On the other hand, many firms and organizations care about the financial state prediction to do some studies. Also, many companies need a financial coverage from other companies or firms, and this cases the creditor company cares about whether the debtor company is a solvent or not. Thus, in order to help improving the quality of such tasks, much efforts has been invested on building prediction models for aiding the decision makers to anticipate the events before they take place. However, developing an accurate bankruptcy prediction model is a challenging task due to the usual high imbalanced distribution in data, where the number of insolvent companies are much less than those successful ones. This makes it very difficult to create an accurate model to classify or forecast healthy companies from bankrupt ones. In that sense, some researchers select the easiest way to solve the data inconsistent distribution problem, in which they select randomly or under certain selection conditions a balanced potion from the original dataset. For example, in order to improve the predictions methods reliability and performance, Iturriaga and Sanz [101] selected a random balanced portion from US commercial banks dataset. The selected portion comprised of 386 records for bankrupt banks and 368 records for solvent banks, used to train a model constructed by combining MLP and Self-Organized Maps (MLP-SOM). And then, the proposed model used to predict the financial status of US commercial banks. In addition, Fedorova et al.[102], in order solve the data balancing problem, they proposed a certain financial conditions for the records in the dataset, but this condition did achieve the required balancing rate, it obtained a dataset contains 3173 records of solvent companies and 473 records of bankrupt ones, while the original dataset comprised of 3505 records of bankrupt companies and 504 records of solvent ones. Thus, there is not a big difference between the original dataset and the obtained one after applying the financial condition (the obtained data is extremely imbalanced), thus, the authors solved the problem by selecting a balanced portion randomly from the obtained dataset dataset. In other words, they randomly selected 444 records of solvent companies and combine it with the 473 records of the solvent companies in order to solve the data balancing problem, and improve the performance and reliability of the classifiers. Moreover, in [38], The problem of the inconsistent data distribution was solved by selecting a balanced subset using stratified sampling method. The selected subset contained 239 records for bankrupt companies and another 239 records for solvent companies.

Thus, the solutions of the data inconsistent distribution problem that addressed in [38, 101, 102] lead to a major problem, which is which is wasting pretty definitely important information by simply eliminating them. On the other hand, replicating the minority instances (bankrupt companies) might lead to overfitting the classifiers; making the prediction of the bankrupt companies very easy to the classifiers.

Accordingly, this chapter presents several approaches could be used to solve the data inconsistent distribution problem, avoiding wasting important information and the overfitting situation. The proposed approaches based mainly on splitting the original dataset into several balanced subsets, each subset contains the whole minority instances (bankrupt companies), and a partition of the majority instances (solvent companies). Then, each subset used to train several classifications algorithms individually. Finally, the final results obtained by calculating the average of the results obtained by each subset.

4.2 CONSIDERED DATASET

In this study,the problem of predicting the financial status of the companies has been addressed, transforming it into a classification one. A combination of the financial and non-financial data has been used. Many previous studies based on some classifiers or predicting method were not effected with more than one type of data; numerical [103].

A dataset brought from the Infotel database has been used; it is a company in charge to gather information in several domains about companies in Spain. Data from 471 companies in Spain during six years sequentially (1998 to 2003) has been used. In this study, several algorithms have been used in order to obtain good and accurate predictions about the financial state of the companies.

The dataset proposed in this work include particular domain attributes used in order to determine whether a firm succeed or fail. It includes 2859 instances, each one of them consists of 39 independent variables of different types (categorical and numerical). After removing meaningless variables (such as internal codes), 33 variables have been adopted, 27 of them are numeric and the remaining are categorical. Some of these variables refer to financial information. Each record represents a company in one year, and have an attribute *Bankruptcy* to mention the financial status for that firm. Table 4.1 shows the adopted independent variables in the dataset.

4.3 EXPERIMENTS AND RESULTS

This section discusses the differences between J48, Random forest and Naïve Bayes classifiers, considering the obtained results in four different (and incremental) experiments, using Weka[1] (a machine learning software suite).

The dataset used in these experiments is extremely unbalanced, it contains 2797 records labeled with healthy companies class and 62 records labeled with bankrupt companies, which is around 98% / 2% out of the whole dataset. This situation creates a challenge for classifiers to work properly.

1 http://www.cs.waikato.ac.nz/ml/weka/index.html

Table 4.1: Independent Variables

Financial Variables	Description	Type
Debt Structure	Long-Term Liabilities / Current Liabilities	Real
Debt Cost	Interest Cost / Total Liabilities	Real
Debt Paying Ability	Operating Cash Flow / Total Liabilities	Real
Debt Ratio	Total Assets / Total Liabilities	Real
Working Capital	Working Capital / Total Assets	Real
Warranty	Financial Warrant	Real
Operating Income Margin	Operating Income / Net Sales	Real
Return on Operating Assets	Operating Income / Average Operating Assets	Real
Return on Equity	Net Income / Average Total Equity	Real
Return on Assets	Net Income / Average Total Assets	Real
Stock Turnover	Cost of Sales / Average Inventory	Real
Asset Turnover	Net Sales / Average Total Assets	Real
Receivable Turnover	Net Sales / Average Receivables	Real
Asset Rotation	Asset allocation decisions	Real
Financial Solvency	Current Assets / Current Liabilities	Real
Acid Test	(Cash Equivalent + Marketable Securities + Net receivables) / Current Liabilities	Real

Non-financial Variables	Description	Type
Year	Corresponding to the sample	Integer
Size	Small | Medium | Large	Categorical
Number of employees		Integer
Age of the company		Integer
Type of company	Public Company | Limited Liability Company | Others	Categorical
Linked to a group	If the company is part of a group holding	Binary
Number of partners		Integer
Province code	Code of the location where the company is set	Categorical
Number of changes of location		Integer
Delay	If the company has submitted its annual accounts on time	Binary
Historic number of Judicial incidences	Since the company was created	Integer
Number of judicial incidences	Last year	Integer
Historic amount of money spent on judicial incidences	Since the company was created	Real
Amount of money spent on judicial incidences	Last year	Real
Historic number of serious incidences	Such as strikes, accidents...	Integer
Audited	If the company has been audited	Binary
Auditor's opinion	Favourable | Exceptions | Unfavourable	Categorical

The solution to deal with such an extremely unbalanced dataset is to use data balancing (or data resampling) methods. That stands on changing the size of the original dataset to get the most proper and optimum dataset to evaluate and training the classifiers [104]. Three methods have been applied here to balance the data: the first one is oversampling method, it stands on creating a superset from the original dataset by replicating some random instances to achieve the desired distribution. The second one is undersampling method, it stands on creating a subset of the original dataset by removing some records randomly. The third is a hybrid approach, which combines the two previous methods. Thus, it stands on removing some valueless records and replicating another part of the training dataset to achieve a fairer distribution of classes in the samples [105].

Actually, four metrics adopted in order to make a judgment about the classifiers; the first one is the *accuracy*, which represents the ability of the classifier to assign the correct class to each instance. The second is the *sensitivity*, It represents the capacity of the classifier regarding to assign the company to the bankruptcy class (prediction) while it is actually bankrupt (real status). The third metrics is the *specificity*. It represents the capacity of the classifier to assign the companies to the succeed class (prediction) while it is actually that (real status). The last metric is the *false positive rate (FPR)*, which represents the failure of the classifier in assigning bankrupt companies to bankruptcy class (wrong prediction), while their actual class is bankruptcy (real status), this metric is a complement value of specificity, both of them have the same Standard deviation value. In other words, the superior classifier gives the maximum accuracy, sensitivity and specificity, and the minimum false positive rate.

The experiments addressed in this section considered in a certain sequence in order to figure out the most convenient to reliant regarding the dataset circumstances; unbalanced dataset, and achieve the required criteria.

The first experiment solves the problem of the balancing by partitioning the dataset to several equally subsets under the coverage of the balancing techniques, i.e. (undersampling and oversampling), to makes the subsets balanced (not exactly). in other hand, the cross validation creates a problem regarding to the reliability; in the case of existing mutual records in test and training folds. To avoid the problem of the reliability, the need of the next experiment arose, it's based on splitting the original dataset to training and test sets, but unfortunately the problem of the metrics values inconsistency appeared due to the balancing problem in the training set; the classifier selects unrequired procedure. To improve this technique the experiment 3 based on using the balancing techniques in the training set of the previous experiment in order to overcome the inconsistency problem. The performance of the classifiers improved as expected but the problem of the inconsistency still exist. The last experiment created as a combination of the 2 previous ways to solve to problem, it stands on merging the dataset partitioning technique and splitting a test set from each partition, under the coverage of the balancing technique to make the training sets balanced after splitting the test sets. The new technique came as improvement of all the previous experiments' techniques, while it solves the problem of the reliability and eliminates the metrics inconsistency problem.

4.3.1 *Experiment 1: Using a balanced dataset*

In this experiment, the dataset has been split in 9 equal subsets, ignoring 7 records as outliers in order to have an exact number of samples in order to avoid the problem of balancing. Thus, each subset contains 310 patterns labeled as healthy companies class (i.e. bankruptcy = 'NO'). On the other hand, and in order to obtain a more balanced amount of patterns labeled with class bankruptcy (YES), the 62 originally available records have been duplicated using an oversampling technique to obtain 124 (30% of samples in every partial dataset). This technique aims to improve the performance of the classifiers. After the subsets created each classifier applied on each subset with 10-fold cross validation. For each classifier the average of the accuracy, sensitivity, specificity

Table 4.2: The performance measures values obtained in Experiment 1

Classifier	Accuracy	Accu. SD	Sensitivity	Se. SD	Specificity	FPR	Sp. & FPR SD
MLP	91.6538%	±0.0113	0.9058	±0.0211	0.9207	0.0791	±0.01411
Random Forest	96.7741%	±0.00669	0.9766	±0.0110	0.9641	0.0358	±0.0068
Naïve Bayes	58.8069%	±0.0825	0.9193	±0.0237	0.4555	0.5443	±0.1230

and false positive rate calculated. Table 4.2 shows the average values and the standard deviation for each metric.

Random forest classifier gave the best results in this experiment, with the maximum values of sensitivity and specificity, and the minimum value of FPR. Thus, it obtains a high performance in predicting the healthy states of the companies while their in fact healthy. in other words, the rate of missing the healthy companies prediction is very low. In addition, it obtain also the maximum sensitivity comparing with the others classifiers.This means, the it obtains a high performance in case of predicting the bankrupt companies while their in fact bankrupt. Thus, the rate of missing the bankrupt companies is very low. Also, the minimum amount of accuracy, sensitivity and specificity standard deviation makes it the most stable classifier for all the subsets in this experiment; this values represents the oscillation of the metrics values obtained by the classifier for each subset test. J48 ranked as a second one with very good results, it have yielded not much less accuracy, sensitivity and specificity than Random forest, and not much more metrics standard deviation values, makes it also stable for all of the subsets. Also, the metrics values are consistent, that means J48 select the expected behavior to solve the problem; it distribute the effort on all of the classes, not just predict one class most the time. The lowest rank, as expected, assigned to Naïve Bayes classifier with the minimum accuracy and minimum specificity, and maximum FPR.Thus, it obtains a low performance in predicting the healthy statuses of the companies while their in fact healthy. in other words, the rate of missing the healthy companies prediction is very high. The value of sensitivity metric in Naïve Bayes still high, which means it still predict failed companies in a high rate on the expense of predicting the healthy companies(not optimum behavior). The

Table 4.3: The performance measures values obtained in Experiment 2

Classifier	**Accuracy**	**Sensitivity**	**Specificity**	**FPR**
J48	94.2207%	0.1666	0.9588	0.0411
Random Forest	97.7233 %	0.1666	0.9946	0.0053
Naïve Bayes	19.6147 %	1	0.1788	0.8211

values of metrics standard deviation is the biggest with Naïve Bayes, proving that it is obviously unstable.

In this experiment, all the records of bankruptcy class have been oversampled in each subset, in order to make them more balanced. In addition cross validation have been used to test the classifiers. This makes the results of all the classifier not very reliable, as some patterns could be potentially located in training folds and test folds at the same time, thus, 'artificially' improving the accuracy and other metrics.

4.3.2 *Experiment 2: Training and test sets*

In this experiment, a subset of the original dataset has been used for testing, while the remaining data has been used to train the classifiers. In this case, data balancing techniques have not been used. The test set contains a specific percentage of the original dataset: 20% of random companies' records labeled as healthy (bankruptcy = 'NO'), i.e. 559 samples. and 20% of random companies' records labeled with bankruptcy class, i.e. 12 records samples.

Proposed classifiers have been used to classify the test set after being trained using a dataset containing 2238 records labeled with healthy class and 50 records labeled with bankruptcy class (training data).

As shown in table 4.3 the accuracy of Random forest and J48 are reasonable, values of specificity and FPR metrics are considerable.

As mentioned before, the rate of missing the healthy companies is low. The value of sensitivity contradicts the results of accuracy, the reason is because the training dataset and the test set are again extremely unbalanced. Therefore, in this experiment also J48 and Random forest unfortunately selects the easiest behavior, which is almost all of the time predicting healthy companies. Even if the accuracy is quite high the behavior of both classifiers make both of them inappropriate to solve the problem in this experiment circumstances. Naïve Bayes has a different situation; it predicts the bankruptcy case more than the reasonable, the rate of missing the bankrupt companies is 0 in the expense of predicting the healthy companies. This gave a poor accuracy for the same reason made the other classifiers confused: the unbalanced dataset.

4.3.3 *Experiment 3: Training and test sets (applying oversampling)*

As in previous experiment, a test set containing the 20% of the samples have been created and the remaining 80% have been used for training. However, in this experiment, a simple data balancing technique has been used on the training set. Thus, the records labeled with bankruptcy class have been replicated several times up to reach a 30% of records from bankrupt companies. Therefore, the training dataset contains totally 3197 records, 2238 records for healthy companies and 959 records for the failed ones.

J48, Random forest and Naïve Bayes classifiers have been applied on the test set, after being trained using the training dataset. Table 4.4 shows the obtained results for each classifier in this experiment.

The results of this experiment became worse than the previous one a little in the case of accuracy, specificity and FPR. On the other hand, sensitivity in this experiment is better than in the previous one, J48 classifier yield a value higher than it with Random forest, but the greatest one is given by Naïve Bayes classifier due to the same problem appears in all previous experiments;

Table 4.4: The performance measures values obtained in Experiment 3

Classifier	Accuracy	Sensitivity	Specificity	FPR
J48	92.2942 %	0.3333	0.9355	0.0644
Random Forest	96.8476 %	0.2500	0.9838	0.0161
Naïve Bayes	18.0385 %	1	0.1627	0.8372

predicting the bankruptcy state more than reasonable. The improvement point in this experiment over the previous one is the distribution of the effort in each classifier, while each one improved the prediction of the bankrupt companies in the expense of the healthy companies. In general, the classifiers made more effort to predict the 2 cases of the financial status as expected, this makes the overall results relatively better than the previous experiment.

In addition, the main issue of previous experiments has been repeated in this one; the unbalanced dataset with huge amount of replications, which made the classifier confused.

As an outline, also in this experiment, all of the classifiers aren't completely appropriate to solve the problem.

4.3.4 *Experiment 4: Several training and test subsets (using oversampling)*

In the last experiment, the dataset has been splitted into nine equal subsets. Each of them contains 310 patterns labeled as healthy companies, and 62 labeled as bankrupt. After this step, every subset has been divided to create a test set containing 20% of each class; 12 records labeled as bankrupt and 62 records labeled as healthy. Thus, every training subset contains 248 records for succeed companies and 50 records for failed ones. Then, the oversampling method was applied to all the training subsets to make them more balanced; 70% of its records for the healthy companies (248 records) and 30% for the failed ones (106 records).

Table 4.5: The performance measures values obtained in Experiment 4

Classifier	Accuracy	Accu. SD	Sensitivity	Se. SD	Specificity	FPR	Sp. & FPR SD
J48	84.9849 %	±0.04479	0.5462	±0.2119	0.9085	0.0913	±0.0252
Random Forest	91.2912 %	±0.0247	0.6018	±0.2143	0.9659	0.0340	±0.0221
Naïve Bayes	57.8078 %	±0.1409	0.8703	±0.1761	0.5214	0.4784	±0.1994

The three classifiers have been then used, and the obtained results are shown in Table 4.5.

In this experiment there are no mutual records in each training subset and its test set, also all the training subsets are balanced (70-30%). This gave strength to the obtained results and made the procedure more robust and reliable. In this case, all the classifiers gave considerable results with relatively low amount of standard deviation, being them equiponderant with regard to the consistency of all the metrics values. Comparing the classifiers results in this experiment, Random forest classifier results were the best with the maximum values of accuracy and specificity, and the lowest FPR value, and the lowest amounts of metrics standard deviation, which makes it the most appropriate classifier to solve the problem with the lowest missing the healthy and bankrupt companies rate, and the lowest oscillatory metrics value which makes it the most stable classifier. J48 ranked second with comparable results regarding to all the metrics, also it is convenient to solve the problem but the preference is for Random forest.

The lowest ranked classifier is Naïve Bayes with the minimum accuracy and specificity, and maximum FPR and standard deviation amount for each metric. As expected the values of sensitivity for Naïve Bayes was the biggest comparing with the others, but unfortunately, its values are lopsided; due to the same reason mentioned in all of the previous experiments, which means it is also inappropriate and unstable in this experiment also.

As shown in all of the previous experiments, Random forest achieved the most considerable results, makes it the ideal and the most stable classifier to do this job. This does not mean that J48 is not convenient or stable also. On the other hand, Naïve

Bayes obtains a poor performance, which makes inappropriate classifier to solve the problem addressed in this study. Using the data balancing methods not always obtain good results regarding the reliability. In fact, it is perfect to improve the performance of the classifiers, but in the case of using cross validation the replication of some records in the dataset represents a major problem and makes the results imprecise.

4.4 FINAL REMARKS

In this chapter, several experiments have been conducted using different datasets from the original dataset (extremely unbalanced) and considered in a certain sequence in order to figure out the convenient procedure to deal with the dataset to obtain the ideal results in predicting Spanish companies financial status. It started from partitioning the dataset into several subsets and using the balancing techniques in order to solve the problem of the balancing, this technique presents unreliable outcomes while the cross validation is used also. Then, the procedure of solving the problem changed to splitting the original dataset in training and test set in order to avoid the reliability problem on the expense of the metrics values consistency; the classifiers select an inappropriate behavior to solve the problem while the training set is extremely unbalanced. Thus, the last experiment came with a perfect solution in order to avoid the disadvantages of the previous procedures, it stands on integrating both of procedure; partitioning the dataset to several equally subsets and split test set for each partition with using the balancing techniques to make the training dataset balanced. This integration yielded the best results regarding the reliability and the consistency of the metrics values and prove itself as the most proper style to solve the problem. In addition, three well-knows classifiers devoted to predict Spanish companies' financial status in the aforementioned work stages, namely: J48, Random forest and Naïve Bayes classifiers. Taking into account the obtained results, Random Forest and J48 obtains considerable outcomes regarding the accuracy of the prediction, sensitivity (recall), specificity, and false positive rate metrics. The outcomes provided by Naïve Bayes are not as a required regarding to all of the metrics values. Random forest outcomes regarding to the financial status prediction are the best.

5 ANALYSING CLASSICAL CLASSIFICATION ALGORITHM WITH SEVERAL DIFFERENT ADVANCED BALANCING TECHNIQUES TO PREDICT COMPANIES' FINANCIAL FAILURE

CONTENTS

This chapter presents a study addressed in a manuscript published in an international conference; "Empirical evaluation of advanced oversampling methods for improving bankruptcy prediction [106]". Accordingly, it is a further step from the previous chapter in predicting companies financial failure, providing a comprehensive analysis about several existing balancing techniques can be used to improve the performance of classifiers prediction in an extremely imbalanced data, and concluding the most appropriate one to solve the inconsistent data distribution problem.

5.1 BACKGROUND

Bankruptcy is a formal insolvency proceeding involving either an individual or a company who have announced their inability to pay the outstanding debts. As a result for this legal status, the debtor's assets are liquidated in order to repay part of the distressed debts, and the remaining portion is discarded [107]. For this reason, the prediction of bankruptcy is a major concern for different financial personnel such as managers, stakeholders, creditors, investors and others who may be affected by the consequences of the financial failure [108].

A successful forecasting for this problem will give a broader perspective about the healthy situation of the business and help the decision makers to predict the events before they take place. For these reasons, there has been an intensive effort in the literature for developing statistical and artificial intelligence-based models to accurately forecast the financial status of the companies. Generally, from machine learning perspective, the prior judging for the company's status either it is bankrupt or non-bankrupt is considered as a binary classification problem.

Developing robust and accurate model for bankruptcy prediction is a complex task. There are so many challenges and difficulties that may negatively impact the model building process and badly affect the generalization performance. Among these challenges are the large number of variables that need to be studied, the missing or unavailable information, or the non-stationary nature of the bankruptcy conditions, which explains the difficulty of adapting a single time period and the necessity to study the company's conditions during multiple periods to reduce the time sensitivity of the model. Moreover, from a machine learning point of view, bankruptcy datasets are imbalanced by nature. This means that the classes are unevenly distributed where the bankrupt class has rare occurrences compared with the normal (i.e non-bankrupt) class. The essence of the problem is that standard classification algorithms deal with both classes as if they would have the same importance, where in fact, the bankrupt class should have more attention for getting more successful classification results. According to this consideration the generated models are biased towards the majority class, whereas the minority class will be neglected in most cases [109, 110]. Hence the model predictive power will be declined and the obtained results may not be reliable anymore.

In literature, different approaches for handling the imbalanced class distribution in datasets have been proposed. Among them, the external approach is the most frequently used. In it, the distribution of the class labels is altered by increasing the minority class patterns (i.e Oversampling) or by decreasing the majority class ones (i.e Undersampling). For bankruptcy prediction, most of the previous works that followed oversampling approaches focused on few methods which are the simple Random Oversampling (ROS) method [111] or the common Synthetic Minority Over-sampling Technique (SMOTE) [112]. For instance, Kim et al.[113] proposed Geometric Mean based Boosting (GMBoost) algorithm to predict the bankruptcy. Also, SMOTE had used to solve the data balancing problem. They proved that the proposed algorithm outperform Adaboost and Cost Sensitive Boosting regarding balanced and imbalanced financial data processing; SMOTE-GMBoost guaranteed equilibrium prediction of bankruptcy and solvency status, it obtained sensitivity from 74.5% to 78.5% and specificity from 77.5% to 82.0%. Also, Zhou[114] made a comparison between Linear discriminant analysis, Logistic regression, Decision trees, Neural networks and SVM classification models utilized to predict the financial status of US firms. Furthermore, in his study, imbalanced US and Japanese bankruptcy datasets were used. The author compared the performance of several balancing techniques, namely: Random oversampling with replication, SMOTE, Random Undersampling, and undersampling based on clustering from Gaussian mixture distribution. Also, the author conclude the ideal circumstances to use each balancing technique, and found that SVM was the superior model compared with the other models used in his study. Later, Le et al.[41] discussed the impact of using the balancing techniques on the Korean companies' bankruptcy status prediction performance. Thus, four classification models were used to predict the financial status, namely; RF, Decision tree, MLP and SVM. The dataset used was extremely imbalanced, so 5 balancing techniques were utilized to solve this problem; (SMOTE, BL-SMOTE, SMOTE-ENN, SMOTE-Tomek and ADASYN). Furthermore, the classification models applied on the data before and after balancing, RF outperform the other models in both cases, but using RF with SMOTE-ENN obtained the best results. More recently, Islam et al.[54] compared 13 classification models regarding predicting bankruptcy status using extremely imbalanced dataset. SMOTE was utilized to resample the data as a preprocessing stage, showing an improvement on the performance of the classification algorithms according the evaluation metrics.

Accordingly, this chapter presents a comprehensive analysis on the classification performance improvements that gained from wide amount of balancing techniques, and concludes the most appropriate one to solve the data distribution problem in financial datasets. In other words, the impact of 11 different advanced balancing techniques on the classifiers' performance and reliability are discussed in detail in order to figure out an ideal technique in balancing companies datasets. Thus, the 11 balancing techniques addressed in this study are: Random oversampling (ROS), Synthetic Minority Oversampling TEchnique (SMOTE), SMOTE with Tomek Links (SMOTE Tomek), SMOTE with Edited Nearest Neighbor (SMOTE-ENN), Borderline SMOTE, Safe-Level SMOTE, Adaptive Synthetic Sampling (ADASYN), ADOMS, SPIDER, SPIDER2 and Agglomerative Hierarchical Clustering (AHC).

5.2 ADVANCED OVERSAMPLING METHODS

In this section, several advanced oversampling techniques that will be used for handling the problem of the inconsistent distribution in the dataset have been described.

- ROS: Random oversampling is a non-heuristic technique, that used to balance an imbalanced data set by increasing the number of minority class members, by simply randomly replicating existing data points. While simple and powerful this method is very sensitive to overfitting [111].

- SMOTE: Synthetic Minority Oversampling Technique, introduces new examples in the training data to enrich the data space and counter the scattered data points in the distribution. It creates new synthetic examples along the line segments joining any or all of the k minority class nearest neighbors, in order to achieve the amount of the oversampling required it selects randomly the k nearest neighbors and then takes the difference between the feature vector (sample) for each minority class sample and its nearest neighbor, and multiply that by a random number that is between 0 and 1, then added it to the feature vector [112].

- SMOTE Tomek: Synthetic Minority Oversampling Technique and Tomek's modification of Condensed Nearest Neighbor, is an advanced oversampling algorithm introduced to overcome the overfitting problem and to remove noisy samples lying on the wrong side of the decision border. It applies a data cleaning method - Tomek links -. First, minor class instances are replicated using SMOTE, then Tomek links are identified and removed for both minor and major classes instances. Tomek links can be defined as follows: a pair of examples is considered a Tomek link if both examples are from different classes and they are the closest to each other, i.e given two examples Ei and Ej that belong to different classes we define the distance between them as $d(Ei, Ej)$ while there is no other example Ek such as $d(Ei, Ek) < d(Ei, Ej)$ or $d(Ej, Ek) < d(Ei, Ej)$ then the pair (Ei, EJ) is considered a Tomek links. If two examples form a Tomek link, then either one of these examples is noise or both examples are borderline [111].

- SMOTE-ENN: Synthetic Minority Oversampling Technique and Edited Nearest Neighbor(ENN), after applying SMOTE oversampling method, this method cleans the dataset by removing the noisy instances in order to increase the classifiers' generalization ability. the cleanup process is defined as follows: For each instance E, ENN will find its 3 nearest neighbors ,if E belongs to the majority class and 2 or more of the nearest neighbors are from minority class then E is removed, and visa versa for instances from minority class [111].

- Borderline SMOTE: During the training process most of the classification algorithms tries to learn the borderline of each class. The borderline examples are the ones that usually misclassified, hence this method focuses on these samples. Works as follows: for each instance E in the minority class it finds its k nearest neighbors, if all of them are from the majority class then this instance is considered noise and ignored. If the number of majority class in the k nearest neighbor is less than the number of instances of the minority class then this instance is considered safe and also ignored. The rest which have more majority class neighbors are considered in danger and these are the instances synthetically replicated. [115].

- Safe-Level SMOTE: Aims to create synthetic instances in safe regions only. Which works as follows: This method assigns a safe level to each minor instance p, the safe level is defined as the number of instances from the minority class in the k nearest neighbor, and then calculates the safe level ratio which is the safe level of p divided by the safe level of the nearest neighbors n. which will result in 5 different cases:

  1. safe level ratio is ∞ and safe level of p is 0, then both p and n are considered noise and ignored.

  2. safe level ratio is ∞ and safe level of p is not zero, which means n is noise so a synthetic instance is created far from n.

  3. safe level ratio is 1, then a synthetic instance is created along the line between p and n.

4. safe level ratio is greater than 1, then a synthetic instance is created closer to p because obviously p is safer than n.

5. safe level ratio is less than 1, then a synthetic instance is created closer to n because n is safer than p [116].

- ADASYN: Adaptive Synthetic Sampling: uses weights to evaluate minority class instances which are hard to predict. For each instance in the minority class, k nearest neighbors are found. Then the density distribution is calculated by dividing the number of instances in the k nearest neighbors that belong to the majority class by k. This value is a measurement of the distribution of weights for different minority class and it is used to determine the number of needed synthetic samples. This helps to reduce the bias and shifting the classification decision boundary to difficult instances [117].

- ADOMS: Adjusting the Direction of the synthetic Minority class Samples, works by generating synthetic examples to fit well in the actual data distribution of the data set, depending solely on the Principal component analysis technique, which is focuses on the variations and patterns in the data set. Synthetic examples will be created along the first principal component axis (the linear combination of the features that have the maximum variance among all linear combinations) of local data distribution which occupies the maximal amount of total variance in the feature space. Proved to be effective to reduce the drop of the classification performance of the experimental classifier in the class imbalance situations and helps to reduce drawbacks caused by newly generated synthetic samples for the minority class [118].

- SPIDER: Selective preprocessing of imbalanced data, this approach uses the internal characteristic of examples to drive their pre-processing, it classifies examples into two types, safe and noisy. Safe examples will be correctly classified by a constructed classifier, however any example that is classified as noisy has a very high chance to get misclassified, hence require pre-processing. The example get classified to be safe or noise by applying the NNR with the heterogeneous value distance metric (HVDM). Then, there is three techniques for pre-processing, they all involve modification of the minority class, however, the degree and scope of changes varies:

1. Weak amplification: amplifies the noisy examples from the minority class by adding as many of their copies as there are safe examples from the majority class in their 3 nearest neighbors.

2. weak amplification and relabeling: Extends on the first technique adding a labeling step: noisy examples from the majority class is located in the 3 nearest neighbor of noisy examples in the minority class are relabeled by changing their assignment from majority class to minority class.

3. strong amplification: Applies weak amplification on safe examples from the minority class. But for noisy examples each example is reclassified using an 5 nearest neighbor.

After that this technique pre-process examples according to their type [119].

- SPIDER2: Similar to SPIDER this method distinguish between safe, borderline and noisy examples, and it claims that the distribution of borderline and noisy examples causes difficulties for learning algorithms. The difference between SPIDER and SPIDER2, is that SPIDER2 method applies a two phase pre-processing for the examples from the majority and the minority classes, while SPIDER identifies the nature of the examples and then simultaneously process the minority and the majority examples, which could result in too extensive modifications in some regions of the Majority class examples [120].

- AHC: Agglomerative Hierarchical Clustering is a clustering algorithm that is used in data mining, this algorithm starts with each example as a cluster of one, and then finds min distance between two clusters and merge them. Removes the original clusters merged from the data set and add the resulted cluster, then iteratively repeat the process until reaches the required number of clusters. To overcome the imbalance issue in the data set, those clusters are used as a prototype to create synthetic examples. This techniques helps in increasing sensitivity for different classifiers [121].

5.3 EVALUATION MEASUREMENTS

In order to evaluate the classifier performance together with the oversampling methods, the confusion matrix which is shown in Table 5.1 has been used.

Table 5.1: confusion matrix

	Predict positive	Predict negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

TP and TN are correctly classified positive and negative instances, misclassified instances are represented by FP and FN [122], and are used to calculate four performance measures Type I error, Type II error, average Geometric Mean and Geometric Mean Standard Deviation (Stdev):

- Type I error: Error rate represented by the false negative rate

$$TypeIError = FN/(TP + FN) \tag{1}$$

- Type II error : Error rate represented by false positive rate [123]
$$TypeIIError = FP/(FP + TN) \tag{2}$$

- Geometric Mean : Popular Evaluation Measure [122]
$$GMean = TPrateXTNrat \tag{3}$$
$$TPrate = TP/(TP + FN) \tag{4}$$
$$TNrate = TN/(TN + FP) \tag{5}$$

5.4 EXPERIMENTS AND RESULTS

In all the experiments the C4.5 Decision Tree has been used as a classification algorithm. C4.5 is very common classifier based on ID3 algorithm with enhancements on handling missing values and continuous attribute value ranges with the ability to choose an appropriate attribute selection measure [124]. In general, decision trees are preferable for this kind of applications because they produce models that are interpretable and easy to explain by the decision makers.

For training and testing different independent datasets have been created, 5-folds cross validation is applied with stratified sampling which stands on splitting the training dataset into several equal (or almost equal) partitions, then use one of these partitions as a test set and the remaining partitions uses to train the classier, this procedure will be repeated for each partition. In other words, each partition will be a test set at each single step of cross validation. The final step of this technique is calculating the average accuracy of test each partition. Also, Stratified sampling is used to preserve the ratios of the two classes in the training and testing partitions, and make them as close as possible to the ratios in the all dataset.

We followed this approach: First, the bankruptcy data are normalized, then, the Oversampling methods are applied to handle

Table 5.2: The parameters used in the experiments

Algorithm	Parameter	Value
C4.5-C	Pruned	true
	Confidence	0.25
	Instances Per Leaf	2
SMOTE-I	Type of Interpolation	standard
SMOTE TL-I	Distance Function	HVDM
SMOTE ENN-I	Distance Function	Euclidean
ADASYN	Type of Interpolation	standard
ADOMS-I	Type of Interpolation	standard
SPIDER-I	Preprocessing Option	WEAK
Safe Level SMOTE	Type of Interpolation	standard
SPIDER2-I	Relabel	true
Borderline SMOTE I	Type of Borderline SMOTE	1
Common	Type of Interpolation	standard
	Alpha	0.5
	Mu	0.5
	Distance Function	HVDM
	Type of SMOTE	both
	Quantity of generated examples	1

imbalanced class distribution. Finally, the resulting data are processed using the C4.5 Classifier.

For the oversampling methods that uses k nearest neighbors algorithm to generate new instances for the minor class, different values for k have been examined, starting from $k = 3$ up to 19 with a step of 2.

Table 5.2 shows the parameter settings for C4.5 classifier and all oversampling methods. In the first step of the experiment, the C4.5 Classifier is evaluated without applying any resampling method. The results of this evaluation are shown in Table 5.3. Due to the high imbalanced data distribution, the classifier shows a poor performance in terms of Type I error.

Table 5.3: Evaluation measures values obtained in the original dataset

Classifier	Type I Error	Type II Error	G-Mean
C4.5-C	77.4%	0.46 %	0.4583

Table 5.4: Evaluation measures values obtained by each combination of C4.5 with balancing techniques. The best *k* parameter value for each approach is shown only.

Oversampling method	Type I Error	Type II Error	G-Mean	Accuracy
SMOTE (k=5)	24.19%	6.79%	0.838	92.82%
SMOTE TL (k=15)	17.74%	11.55%	0.853	88.32%
SMOTE ENN (k=13)	12.90%	12.37%	0.874	87.61%
Borderline SMOTE (k=19)	48.39%	2.86%	0.702	96.15%
Safe Level SMOTE (k=13)	27.41%	20.02%	0.757	79.81%
ROS	54.84%	2.00%	0.651	96.85%
ADASYN (k=5)	30.65%	6.76%	0.803	94.78%
ADOMS (k=5)	38.70%	4.76%	0.754	94.51%
SPIDER (k=3)	53.23%	2.03%	0.672	96.85%
SPIDER2 (k=11)	48.39%	1.82%	0.632	97.17%
AHC	48.38%	2.40%	0.699	96.53 %

The results in terms of Accuracy, Type I, Type II and G-mean are shown in Table 5.4 for all the oversampling approaches. While Type 2 error increased after applying all oversampling methods, the main focus was on Type I error because it is much more relevant in a bankruptcy prediction problem [125].

Figure 5.1 shows the changes in the evaluation measures for each of the top 5 methods when changing the number of neighbors. For example, $k = 13$ used with SMOTE ENN gave the lowest value for Type I error and an adequate value for Type II error; while $k = 7$ used with ADASYN gave the lowest value for type I error and an acceptable value for Type II error.

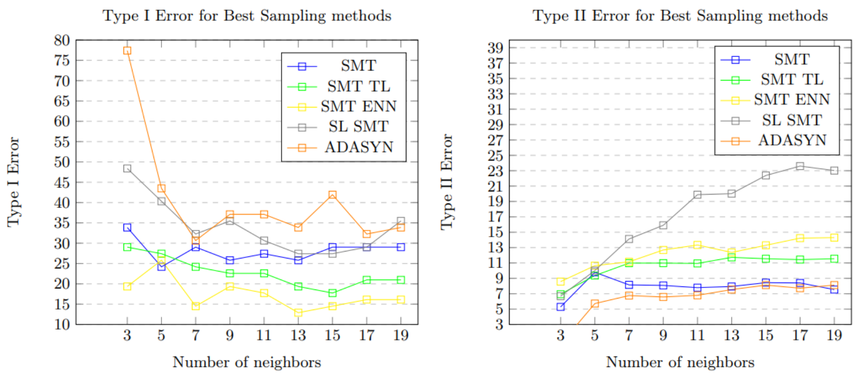

Figure 5.1: Impact of K parameter values (nearest neighbors) on the superior five resampling methods.

Table 5.5: The complexity of C4.5 decision tree

Method	Leafs	Nodes	Type I Error
C4.5 no Sampling	8.4	9.8	77.4%
SMOTE (k=5)	76	151.4	24.19%
SMOTE TL (k=15)	58.8	97.6	17.74%
SMOTE ENN (k=13)	54.2	79.8	12.90%

Finally, for each one of the previously mentioned methods, the effect of oversampling on the decision tree complexity has been studied.

Table 5.5 shows the tree characteristics after applying each sampling method. Thus, the tree complexity for SMOTE ENN is the best considering having the least number of leafs and nodes. From the results of this experiment, SMOTE ENN proved that it is the best oversampling technique to be used with decision tree classifiers.

5.5 FINAL REMARKS

In this chapter, the performance and the impact of eleven different oversampling options has been analysed with respect to using them to solve the inconsistent distribution of a real financial data. In addition, the impact of these different balancing techniques on the classifiers during predicting companies financial failure has been discussed as well. However, all oversampling methods used lead to enhance results regarding type I error compared with classifying the original dataset without using any oversampling method. The noticeable enhancements achieved by SMOTE methods might be due to the nature of this resampling technique, which replicates instances of minor classes depending on their neighbors. Nevertheless, SMOTE-ENN method obtained superior outcomes with with respect to identifying the minority instances (bankrupt companies).

6 IMPROVING THE PERFORMANCE OF CLASSICAL MACHINE LEARNING ALGORITHMS IN PREDICTING COMPANIES' FINANCIAL FAILURE

CONTENTS

This chapter presents a study addressed in a manuscript published in an international conference; "Forecasting Business Failure in Highly Imbalanced Distribution based on Delay Line Reservoir" [126]. This study aims to move one step further from the previous chapter by improving the overall performance of the classical classifiers in predicting companies financial failures.

6.1 BACKGROUND

As mentioned along the previous chapters of this doctoral thesis, predicting companies financial failure is a problem traditionally approached heuristically, which requires a wide knowledge about that companies, and usually is carried out by means of accounting experts. Nowadays it has become a critical problem, approached in the recent years by many researchers. The interest in forecasting the financial status of companies is based on both the management, who can thus count on information to correct foreseeable financial distress, and the credit bureaus and other potential investors, who need this information to evaluate their investment [127]. Recently, several researchers developed and applied wide amount of Artificial Neural Networks algorithms to financial and bankruptcy prediction, and shows considerable success. For instance, in [128], the authors adopted an ANN-based method for financial prediction which reached results as good or even better than a logit model. The authors reported better results when using ANN models, compared to classical statistical methods, as in [129, 130]. Moreover, Sung et al. [131] proposed a data mining approach to predict bankruptcy and financial distress. In [132], the authors devoted Back-Propagation and Optimal Estimation Theory to train the Neural Networks in predicting companies financial failure. They reported that using a neural network trained utilizing Back-Propagation outperforms several traditional statistical methods such as discriminant analysis and logit. Lacher et al. [133] proposed using standard neural network model to forecast the financial status of companies. In addition, in [134] compared several neural networks model to statistical methods in predicting firms financial status. More recently, in [50], the authors proposed using SVM with flexible kernel parameters values to predict bankruptcy in Korean industry firms, then, they also compared the obtained results to some standard statistical methods (i.e.,multiple discriminant analysis and Logit).

On the other hand, many researchers found a way to improve the performance of the machine learning classifiers. Thus, they approached more complex models by combining two or more classifiers together, or combining classifier with some data mining tool in order to improve the overall performance of the models in predicting the financial status. However, three main techniques could be used to combine classifiers and obtain hybrid models. The first technique is cascading the classifiers [135–137], which means using the classifiers sequentially, and use the output of the first classifier as input for the second classifier. The second technique based on clustering and classification [56, 138, 139], which means using the clustering to identify the majority and the minority instances and provide this facility to the classifier in order to predict the financial status of the companies. The Third technique based on integrating two or more classifier into one high performance and complex classifier [140–142].

Accordingly, in this chapter, as a further step from the previous chapters, the cascading technique has been used in order improve the performance of three different classifiers (i.e., J48, KNN and MLP) by combining each one of them with delay line reservoir (DLR) [143]. The obtained hybrid classifiers have been used to predict the financial status of Spanish companies. Also, it is worthy noting that SMOTE balancing technique have been devoted to solve the data inconsistency distribution problem.

Nevertheless, combining the classifier with DLR improved the overall performance of them in the case of processing imbalanced dataset, and even after balancing, the combinations shows better performance than using the standalone classifiers.

6.2 implementation details

For the proposed models in this study, DLR with fully connected input layer has been used, where all input connections have equal weight values $v > 0$; the sign of each input weight is generated from aperiodic pattern. In the reservoir layer, the units are organized in a line, where only elements on the lower sub-diagonal of the reservoir matrix W contains non-zero values

$W_{i+1,i} = r$, for $i = 1, 2, ..., N-1$, where r is the weight of all the feed-forward connections and N is the reservoir size. DLR has been driven with the input data and collect the reservoir states $\dot{X}$ using the following equation.

$$\dot{X} = \frac{1}{\alpha}\Big(-ax + f(Vs + Wx + z)\Big), \tag{6.1}$$

where f is the reservoir activation function (*tanh* in this study), V is the input to reservoir weight matrix, s is the input data, z is zero-mean noise, $a \in [0,1]$ is the leaking rate parameter, and $\alpha > 0$ is the time constant. The reservoir state $\dot{X}$ of DLR will be used as an input to a classification model to produce a classification readout. This can be expressed as follows:

$$Y = Model(\dot{X}) \tag{6.2}$$

where Model is a simple classification model. In this study, four different models will be applied. The first three models are simple well-known classification models, i.e., C4.5 decision tree algorithm, k-NN and MLP. The fourth model is an ensemble majority classification model. Therefore, combining DLR with these classification models will produce four different hybrid models, which will be mentioned as DLR-J48, DLR-k-NN, DLR-MLP, and DLR-Vote, respectively. The general idea of DLR reservoir state combined with classification models is shown in Figure 6.1.

6.3 EXPERIMENTAL DATA

Actually, the dataset considered in this chapter is the same considered in the previous two chapters. It is an extremely imbalanced data of Spanish companies, consists of 2797 instances (98%) belong to the majority class (Solvent companies) and the remaining 62 instance (2%) belong to the minority class (bankrupt companies). Each instance is comprising 27 numeric attributes , and 6 categorical ones. Full details about the Spanish companies dataset financial and nonfinacial independent features addressed in 4.2.

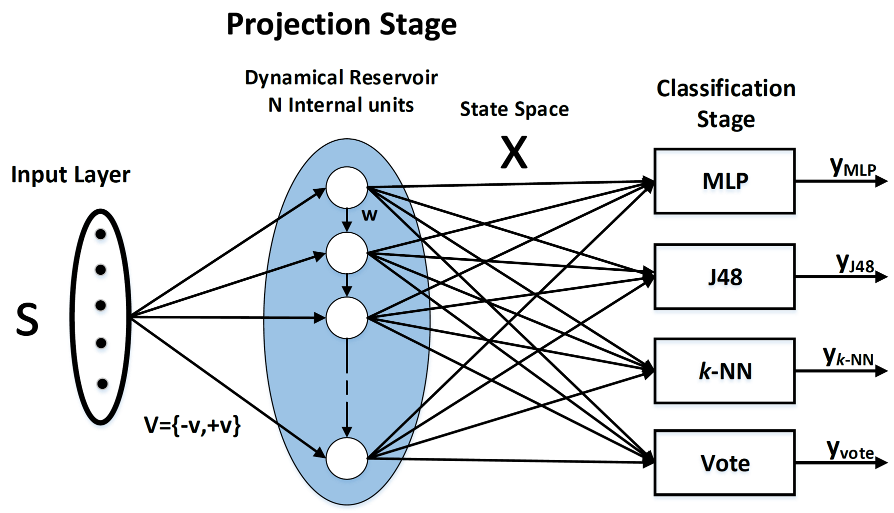

Figure 6.1: The graphical representation for the proposed hybrid classification models.

6.4 EXPERIMENTS AND RESULTS

The DLR part of the proposed framework is trained in a way that the optimal parameters including input weight value v, reservoir weight r, leaking rate a, zero-mean noise z, time constant α, and reservoir size N to be chosen by minimising the mean square error (MSE) values using linear regression based on the training dataset. Regarding k-NN, the number of neighbours is set to 1. For MLP, the number of hidden nodes is set to half the number of input features, the learning rate is 0.3, the momentum is 0.2 and the number epochs is 500. For training and testing, stratified sampling has been used to split the dataset, where 50% of the data is used for training and the remaining 50% is used for testing. Stratified sampling is important for imbalanced datasets to preserve the original ratio of the class labels. Thus, several following evaluation metrics calculated in order to evaluate the performance of the proposed models, namely: Accuracy, Precision, Recall and the geometric mean of the recalls of both classes (G-mean). In the experiments, two different scenarios have been

followed:
Scenario I: Applying the hybrid model DLR model in combination with a majority voting scheme of heterogeneous base classification algorithms (i.e k-NN, MLP, J48) independently. This approach will be compared to DLR in combination with each classifier independently, and it will be compared also with the performance of the base classifiers as well. All these classification methods will be trained using the original dataset; no oversampling.
Scenario II: Same classification methods will be applied this time on training datasets that are oversampled using Synthetic Minority Over-sampling Technique (SMOTE) at different ratios (i.e., 100%, 200%, 300%, 400%, and 500%). The goal of this scenario is to study the performance of the proposed approach after handling the imbalanced class distribution.
Results obtained using the original dataset: The evaluation results of all classification algorithms based on the training dataset without oversampling are shown in Table 6.1. As it can be seen, all classification methods shows very competitive classification accuracy rate. However, since the dataset is highly imbalanced, these ratios do not give a realistic indication of the performance of the classifiers. For these reasons, the other measures are examined. Having a look at the Precision, Recall and G-mean ratios, it can be seen that all DLR based models noticeably outperform the basic classifiers. The best performing model is DLR-MLP with a precision of 48.8%, recall of 64.5%, and a G-mean of 79.7%.
Results obtained using the balanced dataset: In an attempt to improve the classification results obtained in scenario I, all classifiers are trained based on the training dataset oversampled at different ratios (i.e., 100%, 200%, 300%, 400%, and 500%). The results of this experiment are shown in Table 6.2. In general, the results of all classification models have been significantly improved after applying the oversampling step. Moreover, all DLR based models still outperform the basic classifiers. Overall, the best classifier is DLR-J48 at oversampling ratio of 300%. It achieved a precision of 55.8%, recall of 93.5%, and G-mean of 95.9%.

Table 6.1: Evaluation metrics values obtained from each single and hybrid classifier applied to the original dataset.

Classifier	Accuracy	Precision	Recall	G-mean
J48	0.973	0.333	0.226	0.473
MLP	0.974	0.350	0.226	0.473
k-NN	0.968	0.143	0.097	0.309
Vote	0.977	0.417	0.161	0.401
DLR-J48	0.970	0.300	0.290	0.535
DLR-MLP	0.978	0.488	0.645	0.797
DLR-k-NN	0.971	0.273	0.194	0.437
DLR-Vote	0.974	0.385	0.323	0.565

6.5 final remarks

Along this chapter, the performance of three well-known classification algorithms, i.e. J48, k-NN, and MLP solving the bankruptcy prediction problem has been improved by using the reservoir state of DLR as a preprocessing stage for each one of them. In other words, the input point for the proposed three approaches is the reservoir state of DLR, and then the data pass to train the addressed classifiers. Accordingly, The proposed approach has been evaluated on a real world dataset, including information about both successful and failed Spanish companies during six years sequentially. The efficiency of using DLR to improve the classifiers performance demonstrated empirically. The use of DLR reservoir state can lead to performance improvements in Forecasting Business Failure over the normal ensemble voting or other single classifier models including J48, k-NN, and MLP.

Table 6.2: Evaluation metrics values obtained from applying each single and hybrid classifier applied to the oversampled datasets in different ratios.

Classifier	Accuracy	Precision	Recall	G-mean
SMOTE=100%				
J48	0.977	0.444	0.258	0.506
MLP	0.969	0.276	0.258	0.504
k-NN	0.966	0.200	0.194	0.436
Vote	0.977	0.450	0.290	0.537
DLR-J48	0.974	0.432	0.613	0.776
DLR-MLP	0.985	0.656	0.677	0.820
DLR-k-NN	0.971	0.321	0.290	0.535
DLR-Vote	0.977	0.474	0.581	0.757
SMOTE=200%				
J48	0.967	0.250	0.258	0.504
MLP	0.969	0.290	0.290	0.535
k-NN	0.967	0.214	0.194	0.436
Vote	0.973	0.316	0.194	0.438
DLR-J48	0.974	0.429	0.581	0.755
DLR-MLP	0.979	0.516	0.516	0.715
DLR-k-NN	0.971	0.310	0.290	0.535
DLR-Vote	0.978	0.500	0.484	0.692
SMOTE=300%				
J48	0.971	0.351	0.419	0.642
MLP	0.969	0.259	0.226	0.472
k-NN	0.965	0.194	0.194	0.436
Vote	0.972	0.304	0.226	0.472
DLR-J48	0.983	0.558	0.935	0.959
DLR-MLP	0.971	0.378	0.548	0.733
DLR-k-NN	0.972	0.333	0.290	0.535
DLR-Vote	0.980	0.543	0.613	0.778
SMOTE=400%				
J48	0.970	0.342	0.419	0.642
MLP	0.969	0.290	0.290	0.535
k-NN	0.964	0.161	0.161	0.398
Vote	0.978	0.474	0.290	0.537
DLR-J48	0.971	0.383	0.581	0.754
DLR-MLP	0.974	0.429	0.581	0.755
DLR-k-NN	0.972	0.364	0.387	0.617
DLR-Vote	0.979	0.514	0.581	0.757
SMOTE=500%				
J48	0.971	0.292	0.226	0.472
MLP	0.969	0.276	0.258	0.504
k-NN	0.962	0.171	0.194	0.435
Vote	0.974	0.350	0.226	0.473
DLR-J48	0.979	0.511	0.774	0.873
DLR-MLP	0.980	0.529	0.581	0.758
DLR-k-NN	0.971	0.353	0.387	0.617
DLR-Vote	0.980	0.531	0.548	0.737

7 APPLYING DEEP-LEARNING ALGORITHMS WITH ADVANCED DATA BALANCING TECHNIQUES TO PREDICT COMPANIES' FINANCIAL FAILURE.

CONTENTS

This chapter presents a comprehensive study addressed in the published manuscript *"Comparing the Performance of Deep Learning methods to predict companies' Failures"* [144]. And shows a further step from the previous chapters in predicting companies' financial failure.

7.1 BACKGROUND

Deep-Learning (DL) is a subset of Machine Leaning (ML) [145], heavily used in many fields such as Image Processing [146, 147], Computer Vision [148, 149], Fraud Detection [150, 151], Self-Driving Cars [152, 153], Speech Recognition [154, 155], Financial Forecasting [156, 157], Cyber Security [158, 159], etc. Accordingly, talking about the financial forecasting specifically , the problem of financial forecasting or bankruptcy prediction has attracted the attention of researchers since the Crash of 1929 [160]. The effects of bankruptcy on a company are of great significance, as they affect a large number of stakeholders, including workers, creditors and suppliers, and eventually, even entire countries.

Therefore, Machine Learning (ML), and more recently Deep Learning (DL) [161], have gained the interest of researchers in the financial area. More organizations are interested in collecting this important analytical information. For instance, in [43], the authors predicted companies financial status using DBN, the return of stock markets for solvent and bankrupt companies presented as binary images utilized in order to train the models. Furthermore, In [162], the authors proposed a hybrid model comprising DBN devoted to pre-train the data and SVM utilized to classify the data, then the proposed model adopted to predict French companies' financial distress. Also, Hosaka [44] proposed a method based on Convolutional Neural Networks (CNN) to predict bankruptcy using Japanese stock market data represented as a grayscale image. Moreover, the proposed method obtained the optimum results compared to classical and other DL classification methods. However, the data related to the companies' financial status are inherently imbalanced, since the bankruptcy is relatively uncommon in real life[163]. Several studies have been focused on addressing the lack of patterns of minority classes, such as bankrupt companies in the selected problem, because it is dramatically affecting the classifiers, causing a decrease in their reliability and performance. The reason is that those methods tend to build a model to predict the majority class. Thus, many balancing techniques have been proposed in order to solve this problem, using their own criteria to balance the data. For instance, Jang et al.[46] adopted LSTM to predict the Business Failure relying on listed US construction contractors, and then compared the results with others obtained utilizing Feed-forward neural network and SVM. Nevertheless, SMOTE-Tomek was used to solve the data balancing problem. In addition, the same authors [47] also proposed a model based on LSTM to predict the business failure probability from 1 to 3 years using accounting, construction market and macroeconomic variables. Moreover, SMOTE-Tomek balancing technique was used as a data preprocessing stage. Zięba et al. [39], compared several classifiers performance with a novel approach that applies EXtreme Gradiant Boosting (EXGB) for learning an ensemble of decision trees in order to predict companies' financial status in extremely imbalanced polish companies dataset. In addition, In [41] discussed the impact of using the balancing techniques on the Korean companies' bankruptcy status prediction performance. Thus, four classification models were used to predict the financial status, namely; RF, Decision tree, MLP and SVM. The dataset used was extremely imbalanced, so 6 balancing techniques were utilized to solve this problem; (SMOTE, BL-SMOTE, SMOTE-ENN, SMOTE-Tomek and ADASYN). Furthermore, the classification models applied on the data before and after balancing, RF outperform the other models in both cases, but using RF with SMOTE-ENN obtained the best results.

Consequently, this chapter conducts a complete analysis on optimize the configuration of DL methods in order to improve their performance in predicting companies financial failure, and considers them as solid alternatives outperforming several ML methods utilized to solve the same problem in the state of the art. Thus, several different DL methods, i.e., Deep Belief Network (DBN) [93], Multilayer Perceptron model of 6 Layers (MLP-6L) [97] and Long-Short Term Memory (LSTM)[95], have been considered in this chapter to predict companies' financial failure and compared with other standard machine learning algorithms. Each of the selected DL classifiers belongs to a different type of neural network, specifically, MLP-6L is a feed-forward neural network, LSTM is a recurrent neural network, and DBN is a greedily learning stochastic neural network comprised of directed and undirected layers. Moreover, three *bagging*[82] based ensemble methods, i.e, Random Forest (RF)[75], Support Vector Machine (SVM)[77] and K-Nearest Neighbor (KNN)[83], as well as two *boosting* based ensemble methods, i.e, Adaptive Boosting (AdaBoost)[87] and eXtreme Gradient Boosting (XGBoost)[91] have been applied also to solve the same problem and used as benchmark to evaluate the performance f the DL methods. Nevertheless, SVM and KNN, considered as ensemble models using the bagging technique in order to improve their performance, whereas RF is an ensemble of decision trees based on bagging. Also, AdaBoost and XGBoost are boosting-based ensemble methods.

Accordingly, the performance of all the aforementioned classification algorithm have been evaluated with respect to predicting the financial failure into 3 different and complicated financial datasets , consisting of real data with a highly imbalanced ratio will be tested. Thus, real Spanish, Taiwanese and Polish companies' datasets have been considered in order to evaluate the efficiency of the methods to predict the companies' financial failure. As previously mentioned, bankruptcy is rare in the real data, so the datasets that have been used are extremely imbalanced. The major differences between these datasets are the complexity level and the data type's diversity. The Spanish companies' dataset is a combination of nominal and numerical attributes values, and contains financial and non-financial data. On the other hand, the Taiwanese and Polish companies' datasets are more complicated according to the number of attributes and records. The Taiwanese dataset contains the largest number of attributes, while the Polish dataset contains the largest number of samples. Both of them contain only numerical financial attributes. These major differences could affect the behavior of the classifiers and the results. In addition, it might lead to making more accurate decisions about the most appropriate classifier to predict companies' financial failure.

To handle the data inconsistent distribution problem, eight advanced balancing techniques from the literature have been applied in the preprocessing stage, namely, SMOTE (Synthetic Minority Oversampling TEchnique) [112], BL-SMOTE (Borderline SMOTE) [164], SMOTE-ENN [111], K-means SMOTE[165], SMOTE-NC (SMOTE Nominal-Continuous)[112], SMOTE-Tomek (SMOTE with Tomek links)[111], SVM-SMOTE (Support Vector Machine with SMOTE)[166] and ADASYN (ADaptive SYNthetic sampling approach)[117]. These resampling techniques significantly enhance the behavior of the classifiers, i.e., they dramatically decrease the classifiers' minority class misclassification.

7.2 EXPERIMENTAL DATASETS CONSIDERED

As stated, in this chapter, the performance of DL methods to solve real-world complex problem have been discussed. In other word, the impact and the efficiency of using DL algorithm to predict companies financial failure as a well-know example of real-world problem have been studied. Thus, the same dataset used in the previous chapters (4,5 and 6) to train several standard ML and DL algorithms (i.e., Spanish companies' dataset) has been considered in this chapter as well.

However, limiting this study to a single dataset may not be sufficient to evaluate the impact of the standard ML and DL in solving real-world problems (i.e., predicting companies financial failures). That's why the Taiwanese and Polish companies datasets have been considered to predict bankruptcy, given their differences with the Spanish companies' dataset, particularly, the complexity level and data types. The Taiwanese dataset is more complex than the Spanish one, and the Polish dataset is the most complex of them all. The Taiwanese companies' dataset was collected from the Taiwan Economic Journal over 10 years (1999 to 2009) and it contains 6819 records in total: 6599 records of solvent companies (97%), and the rest corresponding to bankrupt companies (220 records). Besides, it is comprised of 95 financial attributes. However, the companies in this dataset were selected according to two conditions, i.e., the information of each company should be available three years before the decision about its financial status, and the size of each company should match with quite enough companies to compare. On the other hand, the decision about each company's financial status is based mainly on Taiwan's stock exchange business regulations. Further information about this dataset is available in [38].

With respect to the Polish companies' dataset, it contains real data collected from Emerging Markets Information Service (EMIS), but focused on enterprises of that country. EMIS is a database comprised of information about emerging markets around the world. It is important to note that the bankrupt Polish companies' data was collected from 2007 to 2013, and the solvent companies' data refer to 2007 to 2012.

This dataset is also extremely imbalanced; the total amount of samples is 10,000, of which 203 are bankrupt (2.03%) and 9797 samples belong to solvent companies (97.97%). In addition, it has 64 numerical financial attributes, and there are not categorical values. More information about this dataset is available in [39]. It can be downloaded from the *Kaggle* ML community web [1].

7.3 EXPERIMENTAL SETUP DETAILS

As stated before, this study aims to predict companies' financial failure considering it as a classification problem. Accordingly, due to the high attainment of DL algorithms regarding classification, the performance of three different types of DL methods, i.e., DBN, MLP-6L and LSTM, and five well-known (and very effective) ensemble classification methods, i.e., RF, SVM, KNN, AdaBoost and XGBoost have been compared; in order to find the best classifier with respect to the problem addressed.

The methods are compared by considering a set of evaluation metrics rather than just computing the usual 'accuracy' measure,

1 `https://www.kaggle.com/c/companies-bankruptcy-forecast`

due to the high imbalance existing in the datasets. These are described in Subsection 7.3.1.

In this regard, and previously to the application of the algorithms, it is mandatory to address the existing data inconsistency problem, in order to improve the classification performance. To this end, several advanced data resampling techniques have been applied to balance the datasets considered. They aim to enhance the classification performance by increasing the minority class instances and decreasing the majority class instances (depending on each balancing technique procedure). Table 7.1 presents the majority and the minority classes' distribution in the three datasets before and after the data balancing step.

Table 7.1: The amount of the records belong to the bankrupt and solvent companies in the three considered datasets (i.e., Spanish, Taiwanese and Polish data). Noteworthy, SMOTE-NC is not used to balance the Taiwanese and Polish datasets because it contain numerical data only.

	Spanish companies' data		*Taiwanese companies' data*		*Polish companies' data*	
Balancing techniques	Bankrupt	Solvent	Bankrupt	Solvent	Bankrupt	Solvent
Original dataset	62	2797	220	6599	203	9797
SMOTE	2797	2797	6599	6599	9797	9797
BL-SMOTE	2797	2797	6599	6599	9797	9797
SVM-SMOTE	1346	2797	2308	6599	4708	9797
ADASYN	2806	2797	6523	6599	9860	9797
SMOTE-NC	2797	2797	-	-	-	-
SMOTE-Tomek	2765	2765	6565	6565	9794	9794
SMOTE-ENN	2651	2494	6260	5433	9757	8546
K-means SMOTE	2797	2797	6599	6599	9797	9797

As shown in the table, the oversampling and the clustering-based techniques, i.e., SMOTE, BL-SMOTE, ADASYN, SMOTE-NC and K-Means SMOTE, generated balanced datasets that contain almost the same percentage for both classes (50% of samples of the majority class and 50% of minority class instances). ADASYN generated more minority class instances compared to the other oversampling methods, depending on the minority data distribution density. Therefore, it generates more synthetic samples in the case of low data distribution density. On the other hand, SVM-SMOTE generated less balanced datasets (32% of the minority class and 68% of the majority class instances), so, this will model a more realistic situation for the problem (with less 'artificial' information added), this could have a negative effect on the results obtained by the classifiers.

The hybrid oversampling-undersampling techniques, i.e., SMOTE-ENN and SMOTE-Tomek, first oversample the minority class to achieve a very close amount to the majority class instances, then the majority class instances are undersampled depending on their cleaning procedure: ENN and Tomek Links. Thus, they generated datasets with variant balancing proportions, even turning the ratio to have more samples of the minority class than those of the majority class.

Specifically, the number of Tomek links in the Spanish companies' dataset (after SMOTE oversampling) was 32, so, after removing those links, the generated balanced dataset contains 2765 instances for each class. In the Taiwanese dataset the number of links was 34, thus, the generated dataset after balancing contains 6565 records for each class, whereas in the Polish dataset the number of links was just three, so, the data distribution among classes remains almost the same. On the other hand, SMOTE-ENN conducts a deeper data cleaning than SMOTE-Tomek. So, after oversampling, it removes the instances belonging to different classes when compared with at least two of their neighbors, that is why there is a considerable difference in the final data distribution.

Moreover, SMOTE-NC is devoted to balancing datasets including nominal and continuous attributes, thus, given the data types in the Spanish, Taiwanese and Polish companies' datasets, SMOTE-NC has only been used to balance the Spanish one, since it contains a mix of categorical and numerical variables, whereas all the variables in the other datasets are numerical.

7.3.1 *The Evaluation Metrics Considered*

Several metrics can be used to measure the performance of any classifier, computed by combining the results obtained in the confusion matrix (see Table 7.2) [167]. Four categories are composing this matrix:

1. *True Positives (TP)*: amount of samples correctly classified as bankrupt.
2. *False Positives (FP)*: amount of samples incorrectly classified as bankrupt.
3. *True Negatives (TN)*: amount of samples correctly classified as solvent.
4. *False Negatives (FN)*: amount of samples incorrectly classified as solvent.

Table 7.2: Confusion matrix.

		Actual	
		Bankrupt	Solvent
Classified	Bankrupt	TP	FN
	Solvent	FP	TN

Thus, since the binary classification accuracy results are not reliable while the data considered is extremely imbalanced (the classifiers always tend to predict the majority class and ignore the minority class), several metrics have been computed to make a better judgment about each classifier's performance and reliability. These metrics are:

- Accuracy [168]: Performance of the classifier in terms of assigning the correct class to each instance.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.1}$$

- Recall (Sensitivity) [169]: Performance of the classifier regarding assigning each sample/company to the 'bankrupt' class (prediction) while it is actually bankrupt (real status).

$$Recall = \frac{TP}{TP + FN} \tag{7.2}$$

- Specificity [169]: Represents the performance of the classifier assigning every company to the 'solvent' class (prediction) while it is actually solvent (real status).

$$Specificity = \frac{TN}{TN + FP} \tag{7.3}$$

- Precision [169]: Performance of the classifier regarding the ratio of assigning the correct class to each sample.

$$Precision = \frac{TP}{TP + FP} \tag{7.4}$$

- Type I error [168]: Also known as False Positive Rate (FPR). It represents the failure of the classifier to assign bankrupt companies to the 'bankrupt' class (wrong prediction), while its actual class is 'bankrupt' (real status).

$$Type\ I\ error = \frac{FP}{TN + FP} = 1 - Specificity \tag{7.5}$$

- Type II error [168]: Also known as False Negative Rate (FNR), represents the failure of the classifier in assigning solvent companies to 'solvent' class (wrong prediction), while its actual class is 'solvent' (real status).

$$Type\ II\ error = \frac{FN}{TP + FN} = 1 - Recall \tag{7.6}$$

Moreover, while the aim of this study is to predict the companies' failure, *recall* and *Type II error* are the most important metrics; they evaluate the performance of the classifiers regarding classifying and misclassifying the bankrupt companies. On the other hand, wholly focusing on the prediction of bankruptcy does not describe the real effectiveness of the classifiers, thus, the remaining metrics are important also to evaluate their overall performance.

7.3.2 *Implementation details*

In order to implement the DL methods in this study, *tensorflow* [98] has been used, which is a software library introduced by Google mainly for DL processing. Also, *tensorflow* has been used as a back end for *Keras* [98], which is an open-source library framework written in Python adopted to implement some DL methods. On the other hand, RF, ensemble SVM, ensemble KNN and AdaBoost have been applied using *scikit-learn* module[170]; it is a package in *Python* programming language comprising implementations of several supervised and unsupervised ML algorithms. Furthermore, XGBoost algorithm has been implemented using *xgboost* [2] package.

To achieve the highest performance and obtain more reliable results from each classifier used in this study, 10-fold cross-validation has been considered in the datasets. That is, the data are split randomly into 10 partitions, nine partitions are used to train the classifier in each iteration, and the remaining partition is used to test the obtained model. Thus, 10 iterations are performed, using in each of them a different test partition [171].

2 https://xgboost.readthedocs.io/en/latest/

7.3.3 *Main Hyperparameters considered to the DL and Ensemble classifier*

Generally, the performance of any DL or ensemble method is tightly linked to the appropriate selection of its hyperparameters values, since some of them have a big effect on the method learning behavior. Selecting the ideal values of the hyperparameters highly depend on the problem to solve, so, they are normally set after an exhaustive experimentation process. Thus, some of the hyperparameters considered in the methods are:

1. *Learning rate*: it is the most important DL method hyperparameter used to adapt the model to the problem. It is in the range [0.0, 1.0]. Selecting an appropriate value to obtain the optimum results is critical: a large learning rate leads the model to suboptimal solutions, whereas a too small learning rate extremely increases the probability of model stagnation.

2. *Epoch*: is a complete learning cycle of the training data in the model.

3. *Batch size*: normally used in artificial neural networks (NNs), is the number of instances that pass through the NN in each epoch, and used to train the model before updating its internal configuration (the weights).

4. *Sigmoid activation function*: is a popular function associated with each neuron in the neural networks, adopted to transform its input data between 0.0 and 1.0 (model output).

5. *Dropout*: is a regulation technique able to improve the accuracy (reducing the overfitting problem) by 'removing' some visible or hidden neurons [172].

6. *Softmax activation function*: it is a function that transforms the outputs in a vector of probabilities, the sum of these probabilities must be up to one. In addition, it improves

the accuracy of the classifiers by increasing the probability of the selected class at the expense of the others. Equation 7.7 presents the *softmax* formula [173]

$$f(x_i) = \frac{exp(x_i)}{\sum_j exp(x_j)} \tag{7.7}$$

where x_i is an input vector, x_j is an output vector and exp is a standard exponential function.

7. *Categorical cross-entropy loss function*: it is a function that reduces the wrong predictions by calculating the difference between the classes probabilities generated by the softmax activation function and the desired output probability (0,1), this allows the model to minimize its deficiencies by adjusting the weights.

8. *Adam optimizer*: is a fast gradient descent optimization algorithm [174] applied in deep NNs. It also guarantees more accurate results by updating the model weights and learning rate during the training.

9. *Rectified Linear Unit (ReLU) activation function*: it is a fast nearly linear function associated normally with each hidden layer or with the output layer for certain types of applications, it is used to solve the vanishing gradient problem during the backpropagation learning by returning 0 if the input value is less than 0 and the same value if it is 0 or more, equation 7.8 shows the formula of ReLU [173].

$$f(x) = max(0, x) = \begin{cases} x_i, & if\ x_i \geq 0 \\ 0, & if\ x_i < 0 \end{cases} \tag{7.8}$$

10. *Kernel Function:* It is a mathematical operator used to transform the data into a required form in order to avoid some complex calculations in the classifiers, and to improve their performance. Several kernel functions could be used to enhance the performance of SVM such as Linear, Polynomial,

Sigmoid and Radial Based Function (RBF) depending on the data distribution[81]. In this study, the RBF has been used is described by the following formula:

$$K(x, x_i) = exp(-\gamma||x - x_i||^2), \gamma > 0 \tag{7.9}$$

Where x and x_i are input samples, and γ is a kernel function parameter.

11. *Number of Estimators (N_estimators):* defines the number of base estimators constructing the ensemble model.

12. *Number of Samples (Max_Samples):* is the number of samples to select from the dataset to train in each estimator, it could be set as a float number in the range of 0.0 to 1.0 (representing a percentage amount of samples), or an integer value in the range of 1 to the total number of samples in the dataset (representing the exact number of samples to fit in each ensemble's estimator).

13. *Number of Features (Max_Features)*: is the number of features to select from the dataset to train in each estimator, also it could be set as a float number in the range of 0.0 to 1.0 (representing a percentage amount of features), or an integer value in the range of 1 to the total number of features in the dataset.

14. *Nearest neighbors parameter (K)*: it is a parameter associated with the KNN method, representing the number of nearest neighbors to select for each sample in the dataset.

15. *BallTree algorithm:* is an algorithm used to search for the nearest neighbors; it organizes the data according to distance metric values defined using two points[175].

16. *Minkowski distance:* is a metric that measures the similarity (distance) between two points in the data according to the following formula:

$$D(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^{\frac{1}{p}} \right)^p, \tag{7.10}$$

where x_i and y_i are data points, and p is an integer. The value of p could make Minkowski metric equal to other metrics, if $p = 1$ (City block distance), $p = 2$ (Euclidean distance) and $p = \infty$ (Chebyshev distance) [86].

17. *Entropy criterion:* is an impurity measure used to evaluate the quality of the data splitting during building the decision trees, less entropy (impurity) more information gain (pure data).

It is important to note that not all these parameters are considered in every DL and ensemble method applied in this study. Tables 7.3 and 7.4 enumerate the usage and considered value for each hyperparameter in the DL and in the Ensemble methods respectively.

Table 7.3: Hyperparameters types and values have been used for each DL method, where DLM is DL method, LRA is Learning RAte, AO is Adam optimizer, BS is Batch Size and CCE is Categorical Cross-Entropy. In addition, '✓' or a value refers to the use of the hyperparameter, and '-' refers to the unused.

DLM	LRA	AO	BS	Epochs	Sigmoid	Softmax	ReLU	CCE	Dropout
DBN	0.005, 0.1	-	40	150	✓	-	-	-	0.2
LSTM	-	✓	15	200	-	✓	-	✓	-
MLP-6L	-	✓	15	200	-	✓	✓	✓	0.2

Table 7.4: Hyperparameters types and values have been used for each ensemble method, where EM is Ensemble Method, KF is Kernel Function, BTA is BallTree algorithm and MDI is Minkowski distance. In addition, '✓' or a value refers to the use of the hyperparameter, and '-' refers to the unused.

EM	KF	N_estimators	Max_Samples	Max_Features	K	BTA	MDI	Entropy
RF	-	100	1.0	1.0	-	-	-	✓
SVM	RBF	40	1.0	1.0	-	-	-	-
KNN	-	40	1.0	1.0	5	✓	✓	-
AdaBoost	-	50	1.0	1.0	-	-	-	✓
XGBoost	-	50	1.0	1.0	-	-	-	✓

7.4 ANALYTICAL RESULTS

As aforementioned, in binary classification problems, the performance and the reliability of any classifier are dramatically affected by the data balancing ratio. Furthermore, in case the data is extremely imbalanced, the classifier tends always to predict the majority class and somehow 'ignore' the minority class. This behavior gains significant results with respect to the accuracy, but it yields poor performance regarding the prediction of the minority class. In other words, the model obtains high accuracy at the expense of the reliability [41, 100].

Therefore, due to the sensitivity of the DL methods to the data distribution inconsistency, eight advanced balancing techniques have been previously applied in this study.

Thus, different datasets have been generated from the Spanish, Taiwanese and Polish datasets - one per each balancing method -. Therefore, every classification algorithm has been tested considering each of the produced datasets.

The following sections present and discuss the results obtained, analyzing specifically the behavior of each DL and ensemble method regarding bankruptcy and solvency misclassification, as well as the effects of each balancing technique on their outcomes.

7.4.1 *Applying DBN with each balancing technique*

DBN with 10-folds cross-validation was applied to each dataset generated using the balancing techniques. Thus, after several trials of DBN application on the three datasets, the most appropriate *RBM* and fine-tuning learning rates must be set respectively to 0.005 and 0.1 (See Table 7.3), also the optimum value of batch size is 40. The activation function for DBN obtaining the best results in this study is *sigmoid*, while *dropout* was set to 0.2.

Table 7.5: The performance indicators (metrics) values obtained by applying DBN to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9174	0.9640	0.8707	0.8819	0.1293	0.0360
BL-SMOTE	0.9175	0.9462	0.8888	0.8953	0.1112	0.0538
SVM-SMOTE	0.9095	0.9094	**0.9096**	0.8516	**0.0904**	0.0906
ADASYN	**0.9409**	**0.9783**	**0.9011**	**0.9136**	**0.0989**	**0.0217**
SMOTE-NC	0.9232	0.9615	0.8849	0.8934	0.1151	0.0385
SMOTE-Tomek	0.9180	0.9677	0.8683	0.8806	0.1317	0.0323
SMOTE-ENN	**0.9414**	**0.9854**	0.8946	**0.9092**	0.1054	**0.0146**
K-means SMOTE	0.9080	0.9687	0.8472	0.8638	0.1528	0.0313
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.8538	0.8861	0.8216	0.8339	0.1784	0.1139
BL-SMOTE	0.8728	**0.9204**	0.8251	0.8414	0.1749	**0.0796**
SVM-SMOTE	0.8714	0.8673	**0.8729**	0.7062	**0.1271**	0.1327
ADASYN	0.8543	0.8532	**0.8555**	**0.8578**	**0.1445**	0.1468
SMOTE-Tomek	**0.8821**	0.9173	0.8468	**0.8589**	0.1532	0.0827
SMOTE-ENN	**0.8809**	**0.9465**	0.8153	0.8367	0.1847	**0.0535**
K-means SMOTE	0.8485	0.8742	0.8227	0.8314	0.1773	0.1258
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.7557	**0.8398**	0.6716	0.7191	0.3284	**0.1602**
BL-SMOTE	0.7638	0.7909	0.7368	**0.7536**	0.2632	0.2091
SVM-SMOTE	**0.8075**	0.6654	**0.8758**	0.7290	**0.1242**	0.3346
ADASYN	0.7353	0.7732	0.6973	0.7217	0.3027	0.2268
SMOTE-Tomek	0.7434	0.8230	0.6637	0.7103	0.3363	0.1770
SMOTE-ENN	**0.8151**	**0.8531**	**0.7718**	**0.8115**	**0.2282**	**0.1469**
K-means SMOTE	0.7463	0.7868	0.7057	0.7287	0.2943	0.2132

The results obtained by DBN applied to the balanced datasets are shown in Table 7.5. These are approximately divergent according to the metrics used in this study to evaluate the performance of all DL and ensemble methods. As it can be seen, DBN shows higher performance with the simplest data (i.e., Spanish dataset) than the other dataset used in this study; more complex data leads to less *accuracy*, *recall* and *precision* metrics values. The results on the Spanish dataset are better overall than the results on the Taiwanese and Polish data, given that the first dataset is 'the simplest' one. For the Polish companies' dataset (the most complex), DBN moderately misses predicting solvent and bankrupt companies as stated in *recall* and *specificity* metrics. Looking at the Spanish companies' dataset, the use of SMOTE-ENN yields the best results in combination with DBN regarding *accuracy*, *recall* and *type II error*, also getting this approach the second-best *precision* value. On the other hand, SVM-SMOTE obtained the complementary best results, i.e., the best *specificity*, and *type I error*. Moreover, ADASYN obtains the best *precision*, also gets the second-best in the remaining metrics. In other words, SMOTE-ENN shows the optimum performance in the classification of bankrupt companies, whereas SVM-SMOTE is the best in classifying solvent ones in this dataset.

With respect to the Taiwanese companies' dataset, again the combination of SMOTE-ENN and DBN yields the lowest bankruptcy misprediction. It obtains the best *recall* and *type II error*, while BL-SMOTE is the second-best technique according to the same two metrics. On the other hand, SVM-SMOTE yields the lowest solvency misprediction as stated in *specificity* and *type I error*, being ADASYN the second-best technique. SMOTE-Tomek can be also remarked, as it obtains the best *accuracy* and *precision*, but very close to SMOTE-ENN and ADASYN methods, respectively.

Regarding the Polish companies' dataset, SMOTE-ENN leads DBN to obtain the optimum *accuracy*, *recall*, *precision* and *type II error*, and the second-best *specificity* and *type I error*. Thus, this is the best approach overall. However, again SVM-SMOTE obtains the best results according to the *specificity* and *type I error*.

Generally, the performance of any ML algorithm depends mainly on the data fitting. Over-fitting yields excellent training results but poor validation ones, whereas under-fitting obtains poor training and validation results [176]. Good-fitting yields relatively close training and validation values. So, discovering if the DBN model learns adequately is quite tricky [177]; since the model's input is just training data. Figure 7.1 illustrates the training loss in the DBN fine-tuning stage. It specifically shows the loss of the superior combination of DBN with balancing techniques with respect to bankruptcy prediction on the three datasets (i.e., DBN + SMOTE-ENN).

The loss of the DBN on the Spanish and Polish data starts from 0.6, and decreases until epoch 100, then they are stabilized at around 0.13 and 0.38, respectively. The loss on the Polish data starts at around 0.5 and stabilizes after epoch 40 at around 0.22 of loss value. From these curves the main conclusion is that the model is not over-fitted, given that the model they show is not training perfectly (loss values are not very low). On the other hand, while the input of DBN model is just training data (no validation curves) it is hard to determine if the model was under-fitted. To solve this issue, extracted 20 subsets of different sizes have been extracted from each balanced dataset generated in this study. For each subset, DBN was trained and tested using 10-fold cross-validation to trace the model's performance with respect to the training and validation data sizes. Thus, Figure 7.2 illustrates the DBN model's training and validation curves obtained using the best data balancing technique for predicting bankrupt companies (i.e., SMOTE-ENN). As it is shown in the figure, each training curve is relatively close to the validation curve (both belong to the same dataset), which shows good-fitting of the Spanish, Taiwanese and Polish data to the DBN model.

Thus, the firm conclusion extracted from the DBN results is that utilizing SMOTE-ENN to balance the datasets helps the DL method to reach the lowest bankruptcy misclassification, which is the most relevant prediction in the problem addressed in this study.

7.4.2 *Applying LSTM with each balancing technique*

In this experiment, four stacked layers of LSTM has been applied on the datasets generated by the balancing techniques in order to improve the accuracy of the model outcomes. Moreover, $softmax$ activation function has been considered in the output layer, so the

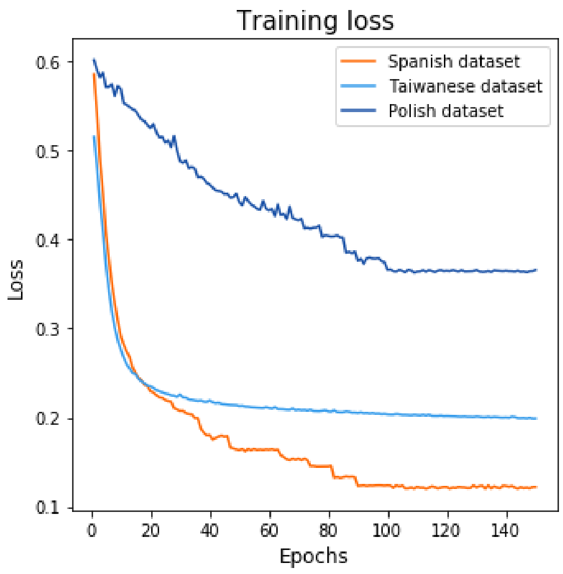

Figure 7.1: The illustration of the loss curves returned by training DBN model using balanced datasets generated by SMOTE-ENN from the three datasets; the best approach in predicting companies financial failure.

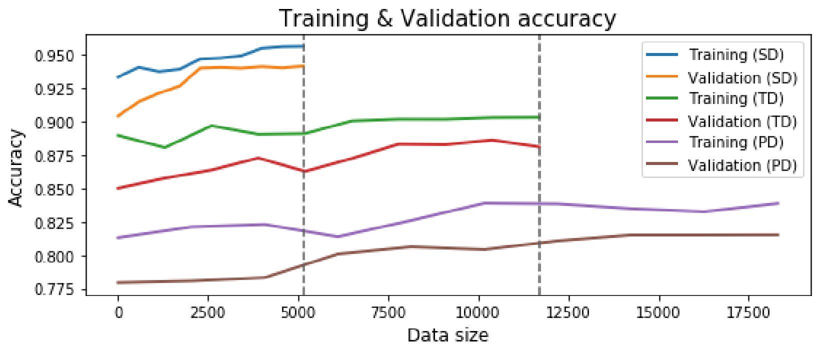

Figure 7.2: The illustration of the training and validation curves that returned from DBN and SMOTE-ENN; the best approach in predicting companies financial failure. The size of each dataset are determined by the vertical lines.

class attribute previously transformed into two attributes, since using this function gets extremely better results than considering *sigmoid* one. In addition, considering that the class attribute transformed into two attributes, *categorical cross – entropy* applied as a loss function. Moreover, after intensive experimentation on LSTM model with the three datasets, setting the batch size to 15 yielded the best results.

Besides, *Adam optimizer* was used to reduce the loss and to obtain more accurate outcomes by updating the model weights and learning rate during the training phase.

Looking at the obtained results presented in Table 7.6, it can be seen the substantial improvement compared to the previous experiment with regard to the Polish companies' dataset. Thus, applying SMOTE-ENN previously to LSTM, leads to outperforming the rest of the balancing techniques with the hardest dataset. This combination obtains the best values for all the metrics, so, it is the most adequate for classifying both bankrupt and solvent companies in this dataset. SMOTE-Tomek gets very close results for *accuracy*, *recall*, and *type II error* metrics (related to bankruptcy prediction), whereas BL-SMOTE is the second-best regarding *specificity*, *precision*, and *type I error* (focused on healthy companies prediction).

Table 7.6: The performance indicators (metrics) values obtained by applying LSTM to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9929	0.9996	0.9861	0.9863	0.0139	0.0004
BL-SMOTE	0.9925	0.9982	0.9868	0.9869	0.0132	0.0018
SVM-SMOTE	0.9923	0.9964	0.9882	0.9884	0.0118	0.0036
ADASYN	0.9939	**1.0**	0.9878	0.9881	0.0122	**0.0**
SMOTE-NC	0.9887	0.9925	0.9850	0.9852	0.0150	0.0075
SMOTE-Tomek	**0.9955**	**1.0**	**0.9908**	**0.9914**	**0.0092**	**0.0**
SMOTE-ENN	0.9942	**1.0**	0.9880	0.9889	0.0120	**0.0**
K-means SMOTE	**0.9948**	**1.0**	**0.9896**	**0.9898**	**0.0104**	**0.0**
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9891	**0.9998**	0.9783	0.9788	0.0217	**0.0002**
BL-SMOTE	0.9884	0.9955	**0.9814**	0.9816	**0.0186**	0.0045
SVM-SMOTE	0.9788	0.9809	0.9780	0.9399	0.0220	0.0191
ADASYN	**0.9897**	**1.0**	0.9795	0.9797	0.0205	**0.0**
SMOTE-Tomek	0.9879	0.9995	0.9762	0.9768	0.0238	0.0005
SMOTE-ENN	**0.9905**	0.9997	0.9799	**0.9829**	0.0201	0.0003
K-means SMOTE	0.9789	0.9758	**0.9821**	**0.9820**	**0.0179**	0.0242
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9896	**0.9985**	0.9808	0.9812	0.0192	**0.0015**
BL-SMOTE	0.9891	0.9921	**0.9860**	**0.9861**	**0.0140**	0.0079
SVM-SMOTE	0.9851	0.9847	0.9700	0.9700	0.0300	0.0153
ADASYN	0.9898	0.9969	0.9827	0.9830	0.0173	0.0031
SMOTE-Tomek	**0.9904**	**0.9985**	0.9823	0.9826	0.0177	**0.0015**
SMOTE-ENN	**0.9931**	**0.9988**	**0.9866**	**0.9884**	**0.0134**	**0.0012**
K-means SMOTE	0.9902	0.9955	0.9849	0.9851	0.0151	0.0045

In the case of the Spanish companies' dataset, the roles are inverted, and SMOTE-Tomek together with LSTM yielded the best outcomes according to all the metrics. This time K-Means SMOTE is the second-best also in all the metrics, however, SMOTE-ENN gets very close results to these approaches, even reaching a perfect value for *recall* and *type II error*, as it is also obtained by the two aforementioned methods.

Regarding the Taiwanese companies' dataset, the combination of ADASYN with LSTM shows perfect performance regarding predicting the bankrupt companies according to *recall* and *type II error*, whereas SMOTE obtains the second-best results in the same case. On the other hand, the combination of K-means SMOTE with LSTM yields the best results in the metrics related to the prediction of the solvent companies, while BL-SMOTE is the second-best. SMOTE-ENN obtains the best *accuracy* and *precision* values.

Moreover, differently than in the previous experiment, LSTM inputs are training and validation data. Figure 7.3 illustrates the curves of LSTM training and validation accuracy/loss in each epoch, obtained from the best combinations in bankruptcy prediction in the three datasets. The best fusions of LSTM are with SMOTE-Tomek, ADASYN and SMOTE-ENN. Thus, the training shows higher accuracy and lower loss than the validation in all curves, but very close values showing good-fitting of the LSTM model to the Spanish, Taiwanese and Polish data. It is important to note that LSTM results are much better than DBN ones, obtaining great indicators for all the metrics in almost all the cases (all the balancing techniques) and in the three datasets. Thus, for every balanced dataset, LSTM tends to predict the correct status of the companies accurately whether they are bankrupt or solvent, with values mostly above 0.98 and errors close to 0 for almost all the approaches.

7.4.3 *Applying MLP-6L with each balancing technique*

Following the process of previous experiments, a deep learning MLP approach with 6 layers, called MLP-6L, has been applied to the eight datasets built after applying each one of the balancing methods. In this case, after exhaustive experimentation on the MLP-6L model, setting the batch size to 15, and using *Adam optimizer* yielded the best results. Again, in this experiment, the class attribute has been transformed into two attributes in order to use $softmax$ in the output layer, as it was compared with *sigmoid* in preliminary tests yielding much better outputs. However, in the hidden layers, *Rectified Linear Unit*

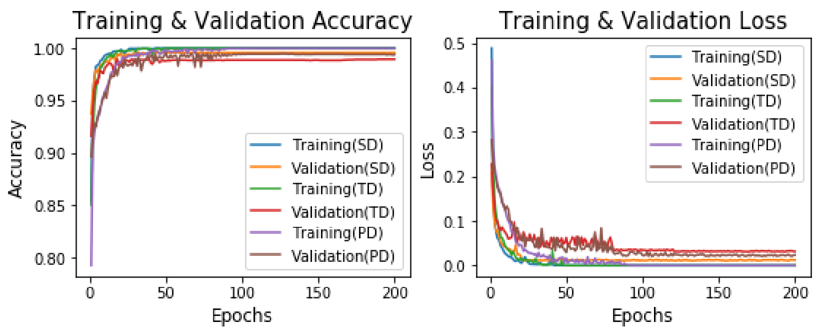

Figure 7.3: The illustration of the training and validation curves that returned from the LSTM and SMOTE-Tomek, ADASYN and SMOTE-ENN that applied invidiously to Spanish, Taiwanese and Polish data ,respectively. These approaches are the best in predicting companies financial failure. The size of each dataset are determined by the vertical lines.

(ReLU) has been considered as the activation function. In addition, because the class attribute was transformed into two attributes, $categorical\ cross - entropy$ was used as loss function.

Table 7.7: The performance indicators (metrics) values obtained by applying MLP-6L to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	**0.9980**	**1.0**	**0.9961**	**0.9961**	**0.0039**	**0.0**
BL-SMOTE	0.9959	0.9975	0.9943	0.9943	0.0057	0.0025
SVM-SMOTE	0.9952	0.9963	0.9946	0.9890	0.0054	0.0037
ADASYN	0.9975	**1.0**	0.9950	0.9951	0.0050	**0.0**
SMOTE-NC	0.9936	0.9939	0.9932	0.9933	0.0068	0.0061
SMOTE-Tomek	0.9973	**1.0**	0.9944	0.9948	0.0056	**0.0**
SMOTE-ENN	**0.9986**	**1.0**	**0.9972**	**0.9974**	**0.0028**	**0.0**
K-means SMOTE	0.9964	0.9989	0.9939	0.9940	0.0061	0.0011
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9910	**1.0**	0.9820	0.9823	0.0180	**0.0**
BL-SMOTE	0.9900	0.9939	0.9861	0.9862	0.0139	0.0061
SVM-SMOTE	0.9839	0.9753	0.9870	0.9633	0.0130	0.0247
ADASYN	0.9905	**1.0**	0.9813	0.9813	0.0187	**0.0**
SMOTE-Tomek	**0.9912**	0.9998	0.9826	0.9830	0.0174	0.0002
SMOTE-ENN	**0.9963**	**1.0**	**0.9921**	**0.9931**	**0.0079**	**0.00**
K-means SMOTE	0.9832	0.9736	**0.9927**	**0.9926**	**0.0073**	0.0264
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9945	0.9991	0.9899	0.9900	0.0101	0.0009
BL-SMOTE	0.9936	0.9950	0.9922	**0.9923**	0.0078	0.0050
SVM-SMOTE	0.9921	0.9911	**0.9932**	0.9859	**0.0068**	0.0089
ADASYN	**0.9953**	**0.9996**	0.9909	0.9911	0.0091	**0.0004**
SMOTE-Tomek	0.9947	0.9995	0.9899	0.9900	0.0101	0.0005
SMOTE-ENN	**0.9967**	**0.9999**	**0.9930**	**0.9939**	**0.0070**	**0.0001**
K-means SMOTE	0.9915	0.9994	0.9826	0.9850	0.0174	0.0006

As it can be seen in Table 7.7, the results obtained by the MLP-6L model applied on all the generated datasets are excellent. Being precise, SMOTE-ENN obtains the best outcomes for almost all the metrics in all the datasets, just getting the second-best for solvent-related metrics (*specificity* and *type I error*) in the Taiwanese and Polish companies' datasets.

In summary, SMOTE-ENN leads the MLP-6L model to obtain the highest *accuracy* and lowest bankrupt companies misclassification rate according to *recall* and *type II error* metrics. Also, the combination of MLP-6L with SMOTE-ENN obtains the second-best solvent companies classification and misclassification rates as stated in *specificity* and *type I error*. Again, it can be seen the complexity of each dataset, since four approaches are able to obtain the highest scores (1.0) for some metrics and the lowest errors (0) with the Spanish dataset, and three approaches with Taiwanese dataset, whereas they are close to the maximum and minimum values, but did not reach them with the Polish data.

Furthermore, Figure 7.4 illustrates the curves of training and validation accuracy/loss obtained from the best MLP-6L combinations in predicting the companies failure (MLP-6L + SMOTE-ENN) in the three datasets. As it is shown in the figure, the validation accuracy and loss are better than the training ones. On the other hand, the validation loss shows relatively better performance than in the previous experiment. Also in this experiment, the training and validation accuracy/loss curves are very close, which shows good-fitting of the three datasets to the MLP-6L model.

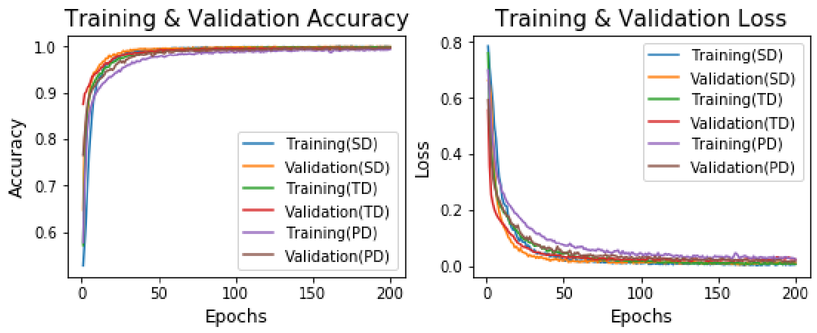

Figure 7.4: The illustration of the training and validation curves that returned from MLP-6L and SMOTE-ENN; the best approach in predicting companies financial failure. The size of each dataset are determined by the vertical lines.

7.4.4 *Applying RF with each balancing technique*

This section presents the results on the application of RF, which is a very effective classifier based on ensembles. As previously told, this is not a DL approach, but it has been extensively used in many difficult classification problems in the literature and obtained excellent results. Thus, after an exhaustive experimentation process with the three datasets, the most appropriate number of ensemble model estimators (*N_Estimators*) is 100; a larger number causes an increment in the computation time and obtains the same results, whereas a smaller number of estimators does not obtain results as good as 100. In the other hand, to fit an entire *bootstrap* in each estimator, *Max_features* and *Max_Samples* are set to 1.0. Besides, the trees splitting criterion is *entropy* while it yields better results than the other criteria in this work. Thus, this method has been combined with the same eight balancing techniques as the DL approaches and tested it on the same three datasets.

Table 7.8: The performance indicators (metrics) values obtained by applying RF to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset

Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9968	0.9979	0.9957	0.9957	0.0043	0.0021
BL-SMOTE	0.9964	0.9961	**0.9968**	0.9968	**0.0032**	0.0039
SVM-SMOTE	0.9947	0.9930	0.9957	0.9924	0.0043	0.0070
ADASYN	**0.9971**	**0.9989**	0.9952	0.9955	0.0048	**0.0011**
SMOTE-NC	0.9955	0.9982	0.9928	0.9929	0.0072	0.0018
SMOTE-Tomek	0.9967	0.9986	0.9949	0.9950	0.0051	0.0014
SMOTE-ENN	**0.9979**	**0.9989**	**0.9968**	**0.9970**	**0.0032**	**0.0011**
K-means SMOTE	0.9921	0.9850	**0.9993**	**0.9993**	**0.0007**	0.0150

Taiwanese companies' dataset

Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9796	0.9950	0.9642	0.9653	0.0358	0.0050
BL-SMOTE	0.9841	0.9918	0.9764	0.9768	0.0236	0.0082
SVM-SMOTE	0.9745	0.9584	0.9801	0.9443	0.0199	0.0416
ADASYN	0.9796	0.9966	0.9627	0.9636	0.0373	0.0034
SMOTE-Tomek	0.9807	**0.9968**	0.9647	0.9658	0.0353	**0.0032**
SMOTE-ENN	**0.9906**	**0.9970**	**0.9834**	**0.9857**	**0.0166**	**0.0030**
K-means SMOTE	**0.9844**	0.9724	**0.9964**	**0.9963**	**0.0036**	0.0276

Polish companies' dataset

Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9932	0.9984	0.9880	0.9881	0.0120	0.0016
BL-SMOTE	**0.9935**	0.9949	0.9920	**0.9921**	0.0080	0.0051
SVM-SMOTE	0.9919	0.9858	**0.9948**	0.9891	**0.0052**	0.0142
ADASYN	0.9930	0.9986	0.9874	0.9877	0.0126	0.0014
SMOTE-Tomek	**0.9933**	**0.9988**	0.9878	0.9879	0.0122	**0.0012**
SMOTE-ENN	0.9926	**0.9992**	0.9852	0.9871	0.0148	**0.0008**
K-means SMOTE	0.9892	0.9797	**0.9988**	**0.9988**	**0.0012**	0.0203

Table 7.8 shows the obtained results for RF in conjunction with the data balancing methods. Thus, the combination of the classifier with SMOTE-ENN outperforms the rest of the approaches concerning the prediction of the bankrupt companies in all the datasets, getting the best *recall* and *type II error*, and also it obtains the best *accuracy* and the second-best values for the remaining metrics in the Spanish and Taiwanese datasets. On the other hand, the combination of the classifier and K-means SMOTE yields the best results in predicting the solvent companies in

all the datasets, obtaining the best *specificity*, *precision* and *type I error*.

Furthermore, to trace the performance of the RF model with respect to the training and validation data sizes, the same procedure addressed in experiment (7.4.1) has been conducted in this experiment as well. In other words, also in this experiment, from each balanced dataset generated in this study, 20 subsets of different sizes were extracted, and then each subset was adopted to train and test RF using 10-fold cross-validation. Figure 7.5 shows the training and validation accuracy curves obtained by the best approaches regarding bankruptcy prediction in the three datasets (RF + SMOTE-ENN). The best three combinations' training accuracy is equal to 1.0, whereas each validation accuracy curve is increasing being close to the training curves and showing that the three datasets good-fitted to RF.

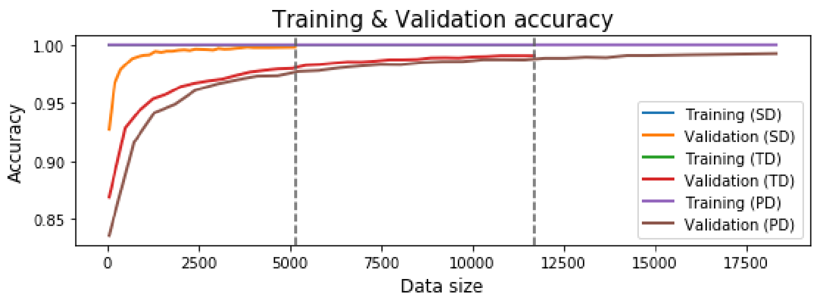

Figure 7.5: The illustration of the training and testing curves that returned from RF and SMOTE-ENN; the best approach in predicting companies' financial failure in all of the datasets (i.e., Spanish dataset (SD), Taiwanese dataset (TD) and Polish dataset (PD)). The size of each dataset is determined through the vertical lines.

7.4.5 *Applying ensemble SVM model with each balancing technique*

In this experiment, SVM has been to the generated datasets after balancing. Thus, in order to improve the performance of this algorithm, it has been proposed as an ensemble model using the bagging technique. The SVM kernel function that have been used is *RBF* because it yielded better performance compared to the other well-known kernel functions (*Linear* and *Sigmoid*) in the problem addressed in this study. Moreover, the most appropriate number of estimators (*N_Estimators*) is 40, whereas the *Max_features* and *Max_Samples* are set as in the previous experiment.

Table 7.9: The performance indicators (metrics) values obtained by applying SVM to each balanced dataset returned by the balancing techniques.The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9694	0.9796	0.9592	0.9602	0.0408	0.0204
BL-SMOTE	0.9726	0.9782	0.9671	0.9675	0.0329	0.0218
SVM-SMOTE	0.9697	0.9630	**0.9735**	0.9539	**0.0265**	0.0370
ADASYN	0.9806	0.9913	0.9691	0.9716	0.0309	0.0087
SMOTE-NC	0.9680	0.9864	0.9496	0.9515	0.0504	0.0136
SMOTE-Tomek	0.9697	0.9791	0.9603	0.9611	0.0397	0.0209
SMOTE-ENN	**0.9825**	**0.9940**	0.9703	**0.9729**	0.0297	**0.0060**
K-means SMOTE	**0.9884**	**0.9800**	**0.9968**	**0.9967**	**0.0032**	**0.0200**
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9560	0.9848	0.9271	0.9311	0.0729	0.0152
BL-SMOTE	0.9639	**0.9851**	0.9426	0.9450	0.0574	**0.0149**
SVM-SMOTE	0.9498	0.9125	0.9629	0.8958	0.0371	0.0875
ADASYN	**0.9885**	0.9784	**0.9987**	**0.9986**	**0.0013**	0.0216
SMOTE-Tomek	0.9543	0.9837	0.9249	0.9291	0.0751	0.0163
SMOTE-ENN	0.9741	**0.9921**	0.9535	0.9605	0.0465	**0.0079**
K-means SMOTE	**0.9836**	0.9677	**0.9994**	**0.9994**	**0.0006**	0.0323
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.8660	0.9694	0.7626	0.8034	0.2374	0.0306
BL-SMOTE	**0.9136**	**0.9824**	0.8447	**0.8636**	0.1553	**0.0176**
SVM-SMOTE	0.9056	0.9023	**0.9072**	0.8240	**0.0928**	0.0977
ADASYN	0.8629	0.9686	0.7567	0.8004	0.2433	0.0314
SMOTE-Tomek	0.8655	0.9692	0.7618	0.8029	0.2382	0.0308
SMOTE-ENN	0.9018	**0.9846**	0.8074	0.8536	0.1926	**0.0154**
K-means SMOTE	**0.9885**	0.9784	**0.9987**	**0.9986**	**0.0013**	0.0216

As it can be seen in Table 7.9, the combination of the ensemble SVM and SMOTE-ENN yields the best performance regarding predicting and mispredicting bankrupt companies in the three datasets according to *recall* and *type II error* values, whereas applying K-means SMOTE obtains the second-best values for the same metrics in the Spanish data, and the application of BL-SMOTE to the Taiwanese and Polish companies' datasets yields the second-best values as well. On the other hand, K-means SMOTE shows the best performance with respect to predicting and mispredicting the solvent companies in all datasets. It obtains the best values of *specificity*, *precision* and *type I error*. In addition, K-means SMOTE obtains the highest *accuracy* values in the Spanish and Polish companies' datasets and the second-best for the Taiwanese one, whereas ADASYN obtains the best *accuracy* value on the Taiwanese data. In addition, also in this experiment, the same procedure addressed in the previous experiment used to trace the performance of the ensemble SVM model.

Figure 7.6 illustrates the training and validation curves obtained by the best approaches in predicting bankrupt companies in the three datasets (i.e., SVM + SMOTE-ENN). Thus, different from the previous experiments, the training and validation values increase along with each other simultaneously; the validation curves are very close to the training ones for each approach, which makes us conclude that the three datasets are good-fitted by the ensemble SVM model.

7.4.6 *Applying ensemble KNN model with each balancing technique*

This subsection presents the results obtained by the application of KNN to the balanced datasets generated by the eight balancing techniques. In this experiment, again KNN has been proposed as an ensemble model using the *bagging* technique as was the case in the previous experiment. Thus, the most proper value of K in the KNN was 5; a lower value increases the impact of the noise on the classifier's results, whereas a higher value increases the computation time and obtained worse results. In addition, *BallTree* algorithm has been used to find the nearest neighbors; it obtains better results than the other searching algorithms in

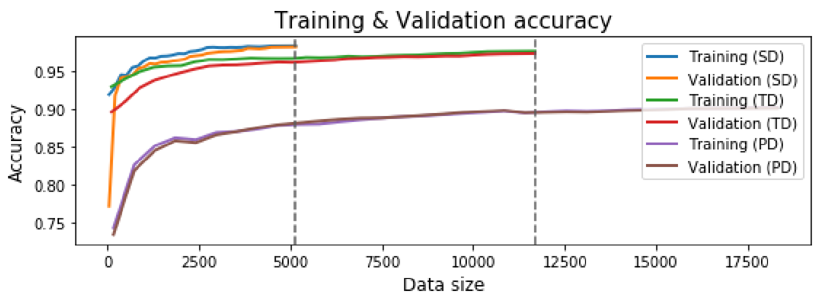

Figure 7.6: The illustration of the training and testing curves that returned from ensemble SVM and SMOTE-ENN; the best approach in predicting companies financial failure in all of the datasets (i.e., Spanish dataset (SD), Taiwanese dataset (TD) and Polish dataset (PD)). The size of each dataset is determined through the vertical lines.

the considered datasets. The distance metric used is *Minkowski*. Moreover, the *N_Estimators* parameter was set to 40; a larger number causes an increment in the computation time and obtains the same results. Also in this experiment, in order to fit an entire *bootstrap* in each estimator, *Max_features* and *Max_Samples* are set to 1.0.

Table 7.10: The performance indicators (metrics) values obtained by applying KNN to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9753	**1.0**	0.9507	0.9531	0.0493	**0.0**
BL-SMOTE	0.9793	0.9986	0.9600	0.9617	0.0400	0.0014
SVM-SMOTE	0.9704	0.9955	0.9564	0.9277	0.0436	0.0045
ADASYN	**0.9833**	**1.0**	**0.9655**	**0.9687**	**0.0345**	**0.0**
SMOTE-NC	0.9703	0.9950	0.9456	0.9484	0.0544	0.0050
SMOTE-Tomek	0.9762	**1.0**	0.9523	0.9547	0.0477	**0.0**
SMOTE-ENN	**0.9835**	**1.0**	**0.9659**	**0.9691**	**0.0341**	**0.0**
K-means SMOTE	0.9739	0.9996	0.9482	0.9508	0.0518	0.0004
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9478	**1.0**	0.8956	0.9056	0.1044	**0.0**
BL-SMOTE	0.9565	0.9965	0.9165	0.9227	0.0835	0.0035
SVM-SMOTE	0.9512	0.9783	0.9417	0.8547	0.0583	0.0217
ADASYN	0.9467	0.9998	0.8941	0.9032	0.1059	0.0002
SMOTE-Tomek	0.9481	**1.0**	0.8963	0.9060	0.1037	**0.0**
SMOTE-ENN	**0.9843**	**1.0**	**0.9665**	**0.9714**	**0.0335**	**0.0**
K-means SMOTE	**0.9833**	0.9709	**0.9958**	**0.9957**	**0.0042**	0.0291
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9455	**0.9997**	0.8914	0.9020	0.1086	**0.0003**
BL-SMOTE	0.9649	0.9959	0.9340	0.9379	0.0660	0.0041
SVM-SMOTE	0.9566	0.9904	0.9403	0.8886	0.0597	0.0096
ADASYN	0.9448	**0.9997**	0.8895	0.9011	0.1105	**0.0003**
SMOTE-Tomek	0.9453	0.9992	0.8915	0.9022	0.1085	0.0008
SMOTE-ENN	**0.9788**	**0.9999**	**0.9546**	**0.9618**	**0.0454**	**0.0001**
K-means SMOTE	**0.9895**	0.9798	**0.9993**	**0.9993**	**0.0007**	0.0202

As it is shown in Table 7.10, and as in the previous set of experiments, the combination of the ensemble KNN and SMOTE-ENN obtained the best *recall* and *type II error* values for the three datasets, i.e., a good bankrupt classification performance.

Paying attention to each dataset, SMOTE-ENN leads the ensemble KNN to obtain the best values of all evaluation metrics in the Spanish companies' dataset showing that it is the ideal combination in order to predict the bankrupt and solvent companies, while the combination with ADASYN yields very close results proving that it could be an adequate alternative.

Regarding the Taiwanese companies' dataset, SMOTE-ENN obtains the best values of *accuracy*, *recall* and *type II error*, and also the second-best values of the remaining metrics. K-means SMOTE shows a great performance regarding predicting the solvent companies compared to the other balancing techniques, as it obtains the best *specificity*, *precision* and *type I error*, and the second-best *accuracy*.

For the Polish companies' dataset, again the combination of ensemble KNN with SMOTE-ENN yields the best results regarding predicting the bankrupt companies, and shows that it is the second-best combination to predict the solvent companies. On the other hand, K-means SMOTE this time also shows a big improvement in the prediction of solvent companies.

Furthermore, as explained in the previous experiment, Figure 7.7 illustrates the training and validation curves obtained from the best ensemble KNN and balancing techniques in predicting bankrupt companies in the three datasets (KNN + SMOTE-ENN). Thus, in this experiment, the training and validation values also increase along with each other simultaneously for each approach, thus, the Spanish, Taiwanese and Polish data are good-fitted to the ensemble KNN.

7.4.7 *Applying AdaBoost with each balancing technique*

AdaBoost is an ensemble method based on the *boosting* technique that showed better performance than bagging and standard classifiers in the classification using imbalanced datasets. In a previous study considered the Spanish companies' dataset [63], applying AdaBoost on the dataset without resampling or Feature Selection obtained the highest *recall* value (0.6) compared to the other classifiers. That value is still relatively low, but it significantly improved later after applying the balancing techniques. In this

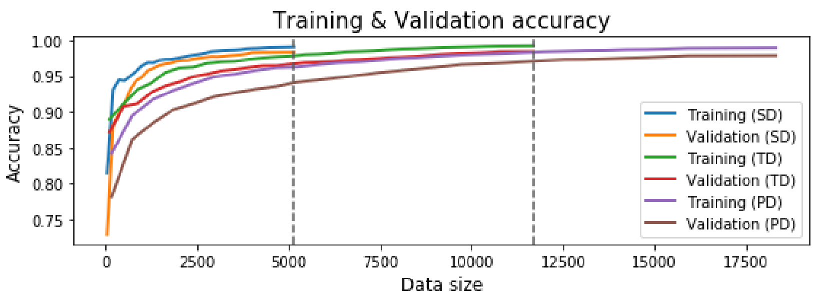

Figure 7.7: The illustration of the training and testing curves that returned from ensemble ensemble KNN and SMOTE-ENN; the best approach in predicting companies financial failure in all of the datasets (i.e., Spanish dataset (SD), Taiwanese dataset (TD) and Polish dataset (PD)). The size of each dataset is determined through the vertical lines.

subsection, the results obtained by AdaBoost method applied to the generated datasets after balancing are presented. Thus, in this experiment, the most appropriate number of estimators (*N_Estimators*) is 50, whereas *Max_features* and *Max_Samples* are set to 1.0 in order to fit the entire data in each estimator. In addition, *Entropy* has been used to measure the quality of data splitting during building the decision trees.

Table 7.11: The performance indicators (metrics) values obtained by applying AdaBoost to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9873	0.9946	0.9800	0.9803	0.0200	0.0054
BL-SMOTE	0.9864	0.9954	0.9775	0.9779	0.0225	0.0046
SVM-SMOTE	0.9828	0.9879	0.9800	0.9651	0.0200	0.0121
ADASYN	**0.9911**	**0.9970**	0.9847	0.9859	0.0153	**0.0030**
SMOTE-NC	0.9828	0.9946	0.9710	0.9718	0.0290	0.0054
SMOTE-Tomek	0.9886	0.9957	0.9816	0.9819	0.0184	0.0043
SMOTE-ENN	**0.9916**	**0.9962**	**0.9867**	**0.9877**	**0.0133**	**0.0038**
K-means SMOTE	0.9900	0.9875	**0.9925**	**0.9925**	**0.0075**	0.0125
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9343	0.9454	0.9232	0.9250	0.0768	0.0546
BL-SMOTE	0.9545	0.9645	0.9445	0.9457	0.0555	0.0355
SVM-SMOTE	0.9449	0.9016	**0.9600**	0.8878	**0.0400**	0.0984
ADASYN	0.9417	0.9528	0.9307	0.9316	0.0693	0.0472
SMOTE-Tomek	0.9440	0.9538	0.9342	0.9355	0.0658	0.0462
SMOTE-ENN	**0.9600**	**0.9697**	0.9491	**0.9559**	0.0509	**0.0303**
K-means SMOTE	**0.9817**	**0.9750**	**0.9883**	**0.9882**	**0.0117**	**0.0250**
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.8976	0.9288	0.8664	0.8743	0.1336	0.0712
BL-SMOTE	**0.9359**	**0.9650**	0.9068	**0.9120**	0.0932	**0.0350**
SVM-SMOTE	0.9327	0.9157	**0.9409**	0.8818	**0.0591**	0.0843
ADASYN	0.8972	0.9275	0.8667	0.8751	0.1333	0.0725
SMOTE-Tomek	0.8933	0.9232	0.8633	0.8711	0.1367	0.0768
SMOTE-ENN	0.9203	0.9513	0.8849	0.9040	0.1151	0.0487
K-means SMOTE	**0.9895**	**0.9822**	**0.9968**	**0.9968**	**0.0032**	**0.0178**

In Table 7.11, it can be seen the significant impact of K-means SMOTE on the AdaBoost regarding predicting the bankrupt and solvent companies in the Taiwanese and Polish datasets. It shows a considerable improvement in the metrics values compared to the rest of the balancing techniques. In the Taiwanese dataset, the second-best results in predicting bankrupt companies have been obtained by using SMOTE-ENN, and BL-SMOTE in the Polish companies' dataset.

Regarding the Spanish companies' dataset, ADASYN leads AdaBoost to the minimum bankruptcy misclassification; it obtains the best *recall* and *type II error*, and the second-best *accuracy*. On the other hand, in the Spanish dataset also, K-means SMOTE shows the optimum performance with respect to predicting the solvent companies; it obtains the best *specificity*, *precision* and *type I error* values. Furthermore, SMOTE-ENN produces close values to the best results, it obtains the best *accuracy* and the second-best values for the rest of the metrics. Figure 7.8 shows the training and validation curves obtained by the best approaches concerning bankruptcy prediction in all datasets, which are AdaBoost in conjunction with ADASYN for the Spanish data, and AdaBoost in conjunction with K-means SMOTE for the Taiwanese and Polish data. The figure shows that AdaBoost fits small training data better than larger one; there is an inverse correlation between the training accuracy and the data size. On the other hand, the validation accuracy is relatively the same for all data sizes, specifically, with the Taiwanese and Polish data. Also in this experiment, the training curves are close to the validation ones, showing that the datasets are well-fitted by AdaBoost.

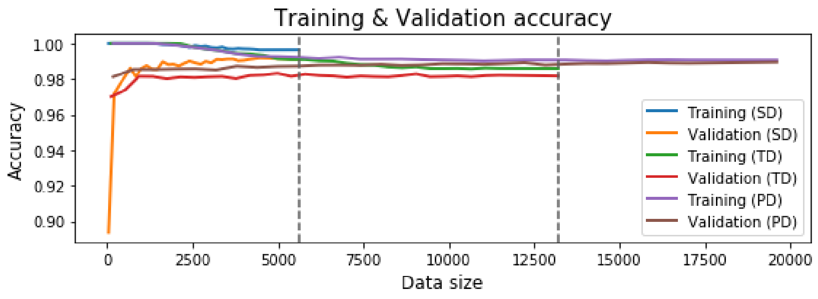

Figure 7.8: The illustration of the training and testing curves that returned from AdaBoost in conjunction with ADASYN and K-means SMOTE; ADASYN is the best to balance the Spanish data (SD), whereas K-means SMOTE is the best to balance the Taiwanese dataset (TD) and Polish dataset (PD). The size of each dataset is determined through the vertical lines.

In the other hand, using K-means SMOTE as a preprocessing stage for the AdaBoost method incredibly improves its performance regarding predicting both classes, i.e., bankruptcy and solvency, in the numeric datasets (Taiwanese and Polish datasets).

7.4.8 *Applying XGBoost with each balancing technique*

This subsection presents the last experiment, carried out to find the most appropriate combination of classification algorithm XGBoost which is another *boosting*-based ensemble method, with the balancing approaches. XGBoost has also shown a higher performance in the imbalanced data classification compared to the other standard and ensemble methods[178], but as in the case of AdaBoost, using the balancing techniques significantly improves the classification performance. Thus, as in the previous experiment, the optimum number of estimators (*N_Estimators*) is 50, and also the entire data was fitted in each estimator by setting *Max_features* and *Max_Samples* to 1.0.

Table 7.12: The performance indicators (metrics) values obtained by applying XGBoost to each balanced dataset returned by the balancing techniques. The first best indicators values are highlighted in gray and bold, whereas the second best are highlighted in bold only.

Spanish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	**0.9950**	0.9989	0.9911	0.9912	0.0089	0.0011
BL-SMOTE	0.9948	0.9968	**0.9929**	**0.9929**	**0.0071**	0.0032
SVM-SMOTE	0.9931	0.9955	0.9918	0.9855	0.0082	0.0045
ADASYN	0.9941	0.9982	0.9900	0.9901	0.0100	0.0018
SMOTE-NC	0.9941	**0.9993**	0.9889	0.9891	0.0111	**0.0007**
SMOTE-Tomek	0.9944	**0.9993**	0.9895	0.9897	0.0105	**0.0007**
SMOTE-ENN	**0.9955**	**0.9996**	0.9912	0.9918	0.0088	**0.0004**
K-means SMOTE	0.9916	0.9864	**0.9968**	**0.9968**	**0.0032**	0.0136
Taiwanese companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	0.9870	0.9979	0.9762	0.9768	0.0238	0.0021
BL-SMOTE	0.9858	0.9927	0.9788	0.9791	0.0212	0.0073
SVM-SMOTE	0.9800	0.9692	0.9838	0.9546	0.0162	0.0308
ADASYN	**0.9872**	**0.9974**	0.9771	0.9774	0.0229	**0.0026**
SMOTE-Tomek	0.9867	**0.9974**	0.9761	0.9766	0.0239	**0.0026**
SMOTE-ENN	**0.9923**	**0.9991**	**0.9846**	**0.9867**	**0.0154**	**0.0009**
K-means SMOTE	0.9845	0.9741	**0.9948**	**0.9947**	**0.0052**	0.0259
Polish companies' dataset						
Balancing technique	**Accuracy**	**Recall**	**Specificity**	**Precision**	**Type I error**	**Type II error**
SMOTE	**0.9942**	**0.9996**	0.9889	0.9890	0.0111	**0.0004**
BL-SMOTE	0.9941	0.9956	0.9925	**0.9926**	0.0075	0.0044
SVM-SMOTE	0.9932	0.9887	**0.9954**	0.9905	**0.0046**	0.0113
ADASYN	0.9941	0.9994	0.9889	0.9891	0.0111	0.0006
SMOTE-Tomek	**0.9943**	0.9993	0.9894	0.9895	0.0106	0.0007
SMOTE-ENN	0.9936	**0.9998**	0.9864	0.9883	0.0136	**0.0002**
K-means SMOTE	0.9918	0.9846	**0.9990**	**0.9990**	**0.0010**	0.0154

Table 7.12 shows the results achieved by the XGBoost method with each balancing technique. SMOTE-ENN surpassed the rest of the techniques in predicting the bankrupt companies (on the three datasets), while K-means SMOTE is the optimum to predict the solvent companies in all datasets also. Paying attention to the Spanish companies' dataset, as before, the conjunction of SMOTE-ENN with XGBoost gets the best *accuracy*, *recall* and *type II error*, whereas SMOTE-NC and SMOTE-Tomek obtain the second-best *recall* and *type II error*. In addition, again K-means SMOTE yields the best *specificity*, *precision* and *type I*

error, while BL-SMOTE achieves the second-best values for the same metrics. Also, in the Taiwanese companies' dataset, the conjunction with SMOTE-ENN leads the XGBoost model to make the best bankrupt companies prediction in comparison with the other balancing techniques. Thus, it gets the best *accuracy*, *recall* and *type II error*, and the second-best values for the remaining metrics.

In addition, the outperforming balancing techniques applied to the Polish companies' dataset are the same used with Spanish and Taiwanese ones (SMOTE-ENN and K-means SMOTE), the second-best *recall* and *type II error* are obtained by SMOTE, while SVM-SMOTE yields the second-best *specificity* and *type I error*. In other words, SMOTE-ENN could be considered as the more adequate preprocessing stage before applying XGBoost aiming to predict the bankruptcy situation, whereas K-means SMOTE is the best to predict the solvent companies.

Furthermore, Figure 7.9 illustrates the training and validation accuracy curves obtained by the superiors XGBoost and balancing techniques combinations concerning bankruptcy prediction in all datasets (XGBoost + SMOTE-ENN). Thus, the same as experiment (7.4.4) curves, all approaches training curves are equal to 1.0, and the validation curves are less, but increasing being close to the training ones. Also in this experiment, the three datasets good-fitted to the XGBoost model.

7.4.9 *Results Summary*

This section aims to clarify the findings of the experiments conducted.

Firstly, a visual representation of all metrics values obtained by each combination of classifier and balancing technique addressed in this study, and applied to the Spanish, Taiwanese and Polish companies' datasets is presented in the Figures 7.10, 7.11 and 7.12, respectively, where it can be seen the high values reached

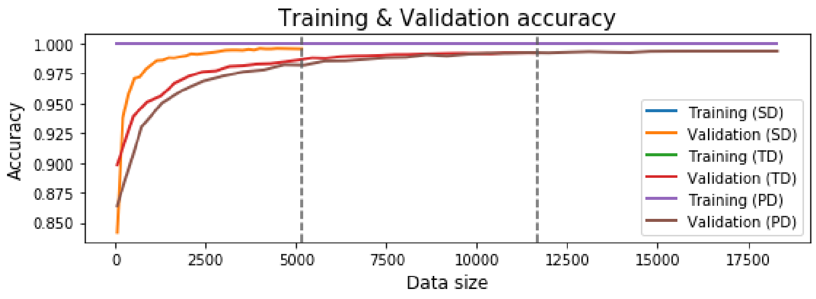

Figure 7.9: The illustration of the training and testing curves that returned from ensemble XGBoost model and SMOTE-ENN; the best approach in predicting companies financial failure in all of the datasets (i.e., Spanish dataset (SD), Taiwanese dataset (TD) and Polish dataset (PD)). The size of each dataset is determined through the vertical lines.

by almost all the methods in most of the metrics (very low errors also). Just DBN shows a lower performance overall, with a noticeable variation of the metrics values obtained.

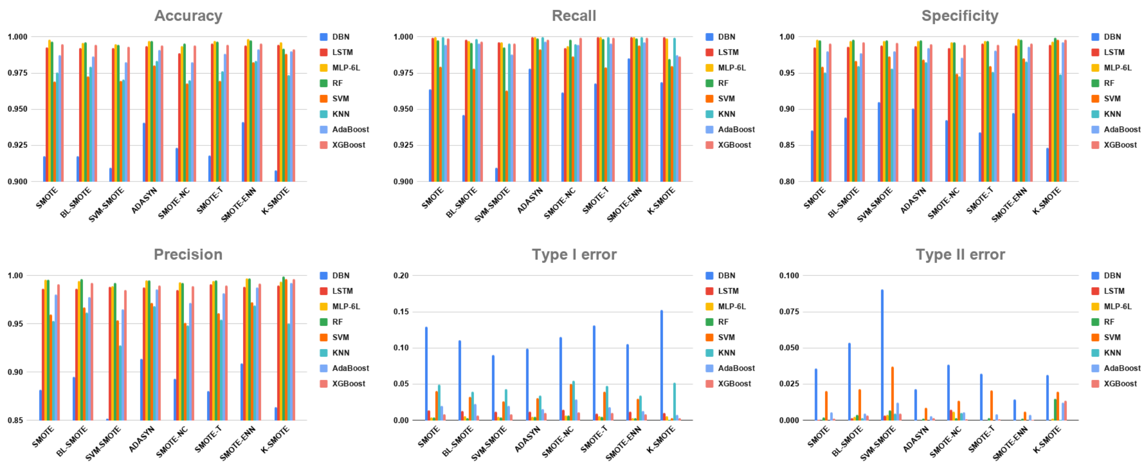

Figure 7.10: Graphical representation for every evaluation metric values returned by each classifier applied to the Spanish companies' dataset.

The reason is DBN was designed mainly to process a different type of data than those considered in this study, i.e., images;

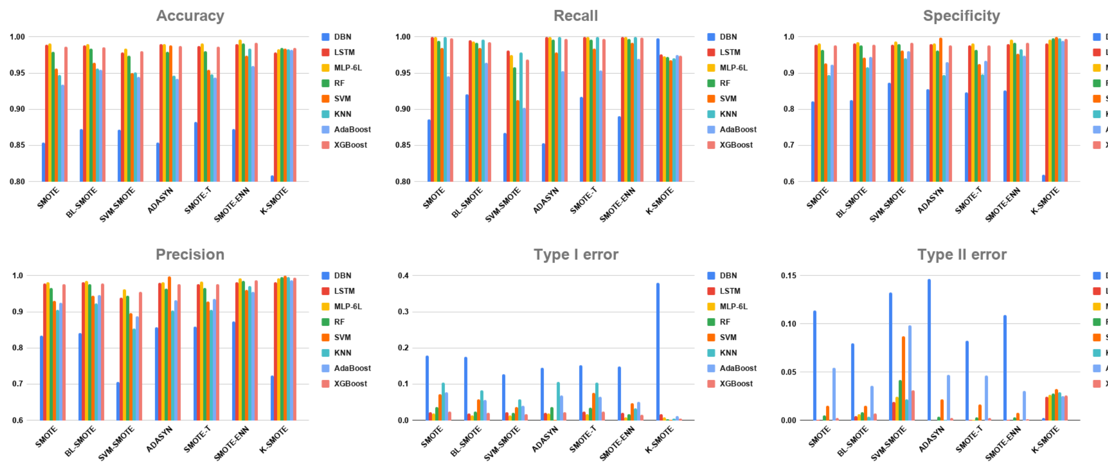

Figure 7.11: Graphical representation for every evaluation metric values returned by each classifier applied to the Taiwanese companies' dataset.

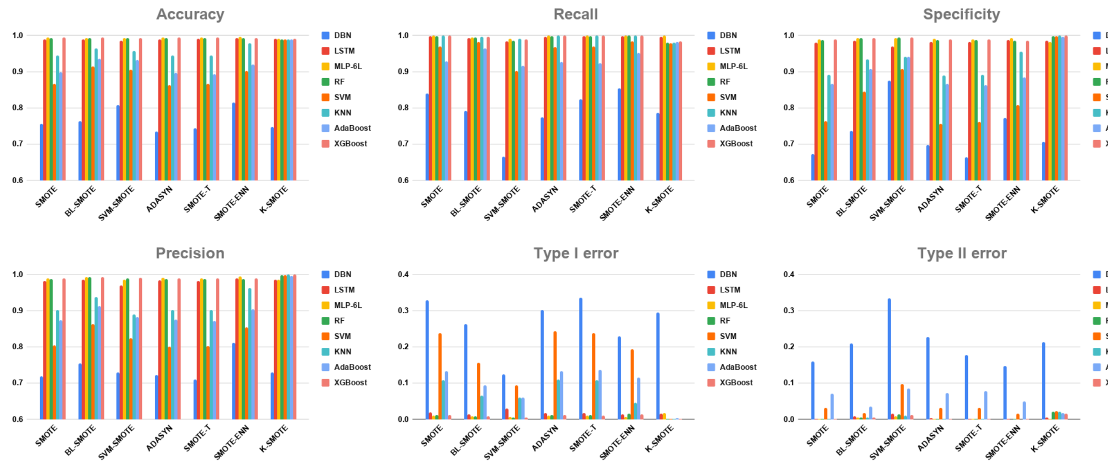

Figure 7.12: Graphical representation for every evaluation metric values returned by each classifier applied to the Taiwanese companies' dataset.

since it was focused on image and pattern classification [43, 179, 180].

MLP-6L, LSTM, RF and XGBoost in conjunction with every balancing technique stand out with high efficiency regarding predicting companies' financial status. In other words, the evaluation metrics values obtained by these algorithms applied on the Spanish companies' dataset are excellent: *accuracy* and *recall* metric values exceeded 0.99, while *type I error* is around 0.01, and *type II error* is very close to 0.0 most of times. Thus, these methods yield very low bankruptcy and solvency misclassification.

Moreover, ensemble SVM, ensemble KNN and AdaBoost show good performance regarding predicting the bankrupt companies in the Spanish dataset, ensemble SVM achieved *accuracy* and *recall* metric values exceeded 0.96. Ensemble KNN shows significant performance in predicting the bankrupt companies, it obtains *recall* around 1.0 for all the balancing techniques, but not high *specificity* as much as the other classifiers. Also, AdaBoost proved that it could be a competitor in predicting bankruptcy with *accuracy* and *recall* exceeded 0.98.

In order to select the best approach to address the bankruptcy prediction problem, now a comparison of all the best combinations between classifiers with data balancing techniques is presented, according to the obtained results for each of them previously described in Sections 7.4.1 to 7.4.8. These are summarized in Table 7.13, which shows the evaluation metrics values obtained by the best approaches.

Table 7.13: Results obtained from the superior combinations of classifiers and balancing techniques in predicting companies financial′ failure (minority instances) in the three datasets. The best values highlighted in gray and bold, whereas the second best are highlighted in bold only.

Dataset	Combinations	Accuracy	Recall	Specificity	Precision	Type I error	Type II error
Spanish companies′ dataset	**DBN + SMOTE-ENN**	0.9414	0.9854	0.8946	0.9092	0.1054	0.0146
	LSTM + SMOTE-Tomek	0.9955	**1.0**	0.9908	0.9914	0.0092	**0.0**
	MLP-6L + SMOTE-ENN	**0.9986**	**1.0**	**0.9972**	**0.9974**	**0.0028**	**0.0**
	RF + SMOTE-ENN	**0.9979**	0.9989	**0.9968**	**0.9970**	**0.0032**	0.0011
	SVM + SMOTE-ENN	0.9825	0.9940	0.9703	0.9729	0.0297	0.0060
	KNN +SMOTE-ENN	0.9835	**1.0**	0.9659	0.9691	0.0341	**0.0**
	AdaBoost + ADASYN	0.9911	0.9970	0.9847	0.9859	0.0153	0.0030
	XGBoost + SMOTE-ENN	0.9955	0.9996	0.9912	0.9918	0.0088	0.0004
Taiwanese companies′ dataset	**DBN + SMOTE-ENN**	0.8809	0.9465	0.8153	0.8367	0.1847	0.0535
	LSTM + ADASYN	0.9897	**1.0**	0.9795	0.9797	0.0205	**0.0**
	MLP + SMOTE-ENN	**0.9963**	**1.0**	**0.9921**	**0.9931**	**0.0079**	**0.0**
	RF + SMOTE-ENN	0.9906	0.9970	0.9834	0.9857	0.0166	0.0030
	SVM + SMOTE-ENN	0.9741	0.9921	0.9535	0.9605	0.0465	0.0079
	KNN + SMOTE-ENN	0.9843	**1.0**	0.9665	0.9714	0.0335	**0.0**
	AdaBoost + K-means SMOTE	0.9817	0.9750	**0.9883**	**0.9882**	**0.0117**	0.0250
	XGBoost + SMOTE-ENN	**0.9923**	0.9991	0.9846	0.9867	0.0154	0.0009
Polish companies′ dataset	**DBN +SMOTE-ENN**	0.8151	0.8531	0.7718	0.8115	0.2282	0.1469
	LSTM + SMOTE-ENN	0.9931	0.9988	0.9866	0.9884	0.0134	0.0012
	MLP-6L + SMOTE-ENN	**0.9967**	**0.9999**	0.9930	0.9939	0.007	**0.0001**
	RF +SMOTE-ENN	0.9926	0.9992	0.9852	0.9871	0.0148	0.0008
	SVM +K-means SMOTE	0.9885	0.9784	**0.9987**	**0.9986**	**0.0013**	0.0216
	KNN +SMOTE-ENN	0.9788	**0.9999**	0.9546	0.9618	0.0454	**0.0001**
	AdaBoost + K-means SMOTE	0.9895	0.9822	**0.9968**	**0.9968**	**0.0032**	0.0178
	XGBoost + SMOTE-ENN	**0.9936**	0.9998	0.9864	0.9883	0.0136	0.0002

As it can be seen in the table, SMOTE-ENN is the best data balancing method applied to the Spanish companies′ dataset, and just in two cases other algorithms reach better results: SMOTE-Tomek and ADASYN. The combination of MLP-6L with SMOTE-ENN shows the highest performance in predicting both the bankrupt and solvent companies, but the combination of RF and SMOTE-ENN obtains very close results.

With respect to the Taiwanese companies′ dataset, due to the increment of its complexity compared to the Spanish one, it is expected that the extra load on the classifier could affect its performance whether decrease bankruptcy, solvency prediction or both. Thus, the increase of the complexity nearly imperceptibly affected the performance of MLP-6L and XGBoost concerning the estimation of bankrupt companies, and shows a noticeable affection in all metrics of the remaining classifiers. Furthermore, also in this dataset, the combination of MLP-6L with SMOTE-ENN outperforms the other combinations in predicting the bankrupt and solvent companies with the best values of all metrics. The combinations of LSTM + ADASYN and ensemble KNN + SMOTE-ENN obtain the best results concerning predict the bankrupt companies, both combinations got the same values of *recall* and *type II error* obtained by the superior combination (MLP-6L + SMOTE-ENN).

Regarding the Polish companies' dataset, it is the most complex one in this study, with this data also, MLP-6L with a data preprocessing stage using SMOTE-ENN is the best approach to classify the bankrupt companies reaching the best *accuracy*, *recall* and *type II error*, whereas the combination of ensemble SVM with K-means SMOTE shows the optimum performance in predicting the solvent companies.

However, as a conclusion, SMOTE-ENN shows significant performance with most of the classifiers no matter the data complexity. The combination of MLP-6L with SMOTE-ENN obtains the best bankrupt companies classification in all the datasets according to the *accuracy*, *recall* and *type II error*. Also, yields the best solvent companies prediction in the Spanish and Polish companies, whereas ensemble SVM with K-means SMOTE were the best in the Polish one.

Furthermore, the aforementioned summary is based mainly on the outperforming combinations of classifiers and balancing techniques in predicting bankruptcy (the aim of this study). The real superior combinations in predicting the solvent companies are K-means SMOTE with RF, ensemble SVM and ensemble KNN in the Spanish, Taiwanese and Polish datasets, respectively.

7.5 COMPARISON WITH APPROACHES ADDRESSED IN THE STATE OF THE ART

Once the DL and ensemble methods have been tested together with the data balancing techniques, this section aims to compare the most effective ones with previous algorithms/results of the state of the art working with the same datasets considered in this study.

First, focusing on the Spanish companies' dataset, Table 7.14 presents a comparison between the best approach selected in this work (MLP-6L + SMOTE-ENN) and the best algorithms or combinations found in five previous works, namely: [62, 63, 100, 106, 181]. In [100], the superior classifier was RF applied to the Spanish dataset balanced using a technique based on dividing the dataset into subsets, which were balanced using a simple oversampling approach. In [106], several balancing techniques were utilized in order to solve the data inconstancy problem as a preprocessing stage before applying C4.5 classifier. SMOTE-ENN balancing technique outperformed all the rest of the balancing techniques. In addition, in [181], combining simple DLR status space with MLP obtained the best results compared with other classifiers. Just SMOTE was applied to solve the data inconsistency problem. Whereas, in [63], the combination of SMOTE and AdaBoost ensemble methods utilizing Reduced Error Pruning Tree (REPT), yielded the best results compared with other basic and ensemble classifiers. It also outperformed the results of using this combination with five different Feature Selection approaches. Finally, in [62], the combination of RF with a Cost-Sensitive Classification (CSC) method outperformed many other ensemble and cost-sensitive methods.

Common metrics computed in all the studies are considered in Table 7.14. As it can be seen in the table, the outputs obtained by MLP-6L + SMOTE-ENN clearly outperform the rest of the results obtained by previous approaches, regarding the bankruptcy prediction and misprediction, according to *accuracy*, *recall* and *type II error* metrics.

Table 7.14: Best evaluation metrics (i.e., accuracy, recall and type II error) values yielded by the best approach in this and previous studies on the Spanish data. The superior values are highlighted in gray and bold.

Metric	MLP-6L+ SMOTE-ENN	In [100]	In [106]	In [181]	In [63]	In [62]
Accuracy	**0.9986**	0.9129	0.8761	0.985	0.983	0.9117
Recall	**1.0**	0.6018	0.8763	0.677	0.55	0.912
Type II error	**0.0**	0.3982	0.1237	0.323	0.45	0.088

Table 7.15: Best evaluation metrics (i.e., accuracy, recall, specificity, type I and type II error) values yielded by the best approach in this and previous studies on the Taiwanese data. The superior values are highlighted in gray and bold.

Combination	Accuracy	Recall	Specificity	Type I error	Type II error
MLP-6L + SMOTE-ENN	**0.9963**	**1.0**	**0.9921**	**0.0079**	**0.0**
SVM + SDA-FC [38]	0.815	0.792	0.837	0.163	0.208

Table 7.16: Best evaluation metric (i.e., AUC) values yielded by all of the approaches in this and a previous study on the Polish data. The superior values are highlighted in gray and bold.

Balancing Technique	DBN	LSTM	MLP-6L	RF	SVM	KNN	AdaBoost	XGBoost	EXGB [39]
SMOTE	0.7557	0.9897	0.9945	0.9932	0.8660	0.9456	0.8976	0.9942	**0.959**
BL-SMOTE	0.7639	0.9890	0.9936	**0.9934**	0.9136	0.9649	0.9359	0.9941	0.944
SVM-SMOTE	0.7706	0.9773	0.9921	0.9903	0.9047	0.9653	0.9283	0.9920	0.940
ADASYN	0.7352	0.9898	0.9952	0.9930	0.8627	0.9446	0.8971	0.9941	0.941
SMOTE-Tomek	0.7433	0.9904	0.9947	0.9933	0.8655	0.9453	0.8932	**0.9943**	0.955
SMOTE-ENN	**0.8125**	**0.9927**	**0.9965**	0.9922	0.8960	0.9772	0.9181	0.9931	
K-means SMOTE	0.7463	0.9902	0.9910	0.9892	**0.9886**	**0.9895**	**0.9895**	0.9918	

The second comparison will be focused on the Taiwanese dataset, which is more complicated than the Spanish one. Here the performance of the superior combination of classifier and balancing techniques applied to the Taiwanese dataset in this study with the best approach addressed in [38] have been compared. In that work, the authors proved that combining the Financial Ratios (FRs) and Corporate Governance Indicators (CGIs) improves the classifiers' performance in predicting Taiwanese companies' financial status. Moreover, five Feature Selection approaches were compared for reducing data dimensionality after this combination. Thus, SVM with Stepwise Discriminant Analysis (SDA) Feature Selection method to the combination of FRs and CGIs (FC) obtained the best results. Table 7.15 shows the results of SVM + SDA-FC applied to the Taiwanese dataset, and the best approach used in this study. As it can be seen in the table, MLP-6L + SMOTE-ENN yields much better results than the best approach in the previous study.

The final comparison is focused on the Polish companies' dataset. The best results obtained by all the approaches tested here with the algorithm proposed in [39] have been compared, where the authors compared the performance of several classifiers and regression methods with a novel approach that utilizes Extreme Gradient Boosting (EXGB) for learning an ensemble of decision trees in order to predict companies' financial status. Moreover, EXGB is widely used in Kaggle competitions[3] on classification problems. Thus, the results found in [39] outperformed all the referenced methods there, regarding the prediction of companies' financial status. In their study, they divided the dataset into five subsets depending on the years.

Area Under ROC Curve (AUC) is the only metric used to evaluate the classification and regression models' performance. Receiver Operating Characteristics (ROC) curve [182] is a graph showing the performance of classifiers and regression models by means of a series of thresholds. In the case of binary classification, there is only one threshold value. Equation 7.11 represents the AUC formula:

$$AUC = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right) \tag{7.11}$$

Thus, in order to compare the methods, the AUC value have been computed for all the DL and RF methods considered in this study together with the data balancing techniques. Table 7.16

3 https://www.kaggle.com/competitions

shows the results of this metric for all the approaches and the one in [39].

As it can be seen, LSTM, MLP-6L, RF and XGBoost, combined with all the balancing techniques, and ensemble SVM, ensemble KNN and AdaBoost in conjunction with just K-means SMOTE, get better results than the EXGB algorithm. Thus, again MLP-6L + SMOTE-ENN reaches the best metric performance beating in almost four points the state of the art method.

Accordingly, the firm conclusion extracted from the comparison results is that the superior approach adopted to predict companies' financial failure in this study (i.e., MLP-6L + SMOTE-ENN) outperformed the other approaches addressed in the state of the art with the same Spanish, Taiwanese, and Polish companies' datasets.

7.6 FINAL REMARKS

This chapter has addressed companies' financial status classification problem using three DL algorithms, three bagging ensemble and two boosting ensemble classification methods. As DL algorithms, the Deep Belief Network (DBN), Multi-Layer Perceptron with 6 layers (MLP-6L), and Long-Short Term Memory (LSTM) have been chosen. The bagging ensemble classifiers are Random Forest (RF), Support Vector Machine (SVM) and K-Nearest Neighbor (KNN); and the boosting ensemble classifiers are ADaptive boost (AdaBoost) and Extreme Gradient Boosting (XGBoost).

A difficult and very imbalanced problem is faced in this work, using three different datasets: A Spanish companies' dataset, which was provided by Infotel company, a Taiwanese companies' dataset, collected from Taiwan economic journal, and a Polish companies' dataset, collected from the Emerging Markets Information Service (EMIS). In order to cope with this problem, three types of balancing techniques have been utilized, namely: oversampling (SMOTE, Borderline SMOTE, SMOTE-NC, SVM-

SMOTE and ADAYSN), oversampling+undersampling (SMOTE-ENN and SMOTE-Tomek); and clustering-based balancing (K-means SMOTE), so eight data resampling methods have been combined with the classifiers.

Several metrics have been considered (in addition to the classical accuracy) to properly measure the performance of each classification method applied to each balanced dataset. After extensive experiments were done, and according to the evaluation metrics, MLP-6L applied to the datasets generated using SMOTE-ENN balancing technique obtained the best results regarding predicting companies' financial failure.

Indeed SMOTE-ENN was the mutual superior balancing technique for all DL methods leading them to reach the lowest bankruptcy misclassification. The best DL approaches have been compared with state of the art methods applied by other authors to the same datasets, outperforming the results previously obtained.

8

A NOVEL DATA BALANCING TECHNIQUE: DBBI-SMOTE (DISTANCE-BASED BORDER INSTANCES SMOTE)

CONTENTS

In this chapter, as a novel part of this thesis, and never published before, a novel data balancing technique named Distance-Based Border Instances SMOTE (DBBI-SMOTE) is presented. Nevertheless, this chapter is a further step of the previous chapters of this doctoral thesis. It produces a novel data balancing technique having enhancements on some existing data balancing methods, and shows high performance in improving the reliability of the results obtained by the classifiers.

8.1 background

The inconsistent distribution of the instances in the data space mostly becomes an obstacle in supervised learning. Generally, the data with inconsistent instances distribution is called imbalanced datasets, and it can be recognized depending on the number of instances that belong to each class in the dataset. Thus, the dataset can be defined as imbalanced if it contains a number of instances belonging to a certain class that severely exceed the number of instances that belong to the other classes [183, 184]. Due to the scarcity of certain important observations in the real world, such as some rare medical cases, aircraft accidents and bankrupt companies, the data balancing problem usually occurs with the real data. Moreover, the data balancing problem presents a significant challenge for the classifiers. The machine learning classifiers tend always to determine the instances as majority instances. This means that the accuracy obtained from the classifiers will be extremely high, but the overall accuracy is definitely unreliable. In other words, during the classification procedure, the most important instances to predict are the minority instances, because they represent the rare cases that the concerned parties are normally interested in. But, in the case of the imbalanced data distributions, the classifiers ignore the minority instances (rare observations) and keep predicting the majority instances (common observations). Accordingly, many researchers have tried to solve this problem in order to improve the performance of the classification. The first and the easiest approaches addressed in the literature to solve the balancing problem are Random Oversampling and Random Undersampling. Random Oversampling is the approach of increasing the number of the minority instances in the dataset by replicating them several times to achieve a certainly required balancing ratio. Random Undersampling is the approach used to balance the dataset by decreasing the number of the majority instances by random eliminating until achieving a certainly required balancing ratio. Besides, Random Oversampling and Random Undersampling are not convenient to solve the balancing problem in most of the cases, because Random oversampling most probably will cause an overfitting situation in the classifiers, whereas Random Undersampling is a waste for definitely important data. Accordingly, several researchers pay more attention to further solutions that could avoid the major drawbacks of the Random Oversampling and Random Undersampling [185–188]. One of the other approaches used in the literature to avoid data balancing problems is Cost-sensitive learning [189] which is not modifying the original imbalanced dataset directly, instead, it adopted a cost-matrix for the instances that belong to different classes to improve the classification using imbalanced data. On the other hand, many other researchers proposed balancing approaches based on creating new synthetics having the same characteristics of the minority instances and labeled with the same class. Thus, Synthetic Minority Oversampling TEchnique (SMOTE) [112], is the most known method that follows the procedure of creating new minority synthetics to fill the balancing gap in the dataset and avoid overfitting. Later, many other researchers paid more effort regarding improving the performance of SMOTE and avoiding the drawbacks that could happen in some data distribution circumstances [190–193]. Thus, the balancing techniques based on SMOTE could be grouped into 3 categories: 1- Oversampling methods [117, 164, 186] 2- Hybrid methods (Oversampling-undersampling) [111, 194, 195] 3- Clustering-based methods [196, 197].

Thus, along with this chapter, a novel data balancing method named "Distance-Based Minority Instances SMOTE (DBBI-SMOTE)" proposed to fill the gap in the existing SMOTE variations and avoid some drawbacks in their procedure. Mainly, DBBI-SMOTE has been constructed based on the procedures of SMOTE and Borderline SMOTE balancing methods. In addition, DBBI-SMOTE has been evaluated comprehensively using 4 different companies' datasets concerning the data types and the complexity level. The considered datasets are of Spanish, Taiwanese, Polish and US companies, all of these datasets are extremely imbalanced and represent an ideal environment to evaluate the feasibility of any resampling method in the literature. Nevertheless, the performance of the new proposed method compared to other 34 SMOTE variations belong to the 3 aforementioned categories. The full distribution of the novel balancing method is addressed in Section 8.3. Also, full details about the compared 34 SMOTE variations are described in the next section 8.2, and finally, the comparison with the other methods proposed in the literature is explained in Section 8.4.

8.2 THE COMPARED DATA BALANCING TECHNIQUES

This section presents in detail the procedures of the compared SMOTE variants to the new proposed one (DBBI-SMOTE). Basically, SMOTE variants are grouped into 3 categories: Oversampling methods, Hybrid methods (oversampling-undersampling) and Clustering-Based methods. Thus, the following subsections identify the category and explain the procedure of each one of the compared SMOTE variants.

8.2.1 *Oversampling methods*

Actually, many balancing techniques are based mainly in their procedures on increasing the amount of the minority instances to balance the dataset and ignore the majority instances. The following balancing techniques are only oversampling-based techniques.

8.2.1.1 *Synthetic Minority Oversampling TEchnique (SMOTE)*

SMOTE is a more advanced technique than random oversampling with replacement. Instead, in order to improve the reality of the oversampled data and prevent replicating the minority class instances, SMOTE generates new synthetics belonging to the minority class depending on the nearest neighbors of each instance in the same class. Moreover, the new synthetics are similar to the minority instances but not the same. [112].

The main procedure of SMOTE data balancing technique comprised of several steps:

- Explore the dataset to determine the minority class instances.
- Select randomly one minority class instance and find its five nearest neighbors that belong to the same class.
- Depending on the oversampling ratio, a certain number of nearest neighbors will be selected in order to generate the new synthetics; if the oversampling ratio set to 200%, only two of the selected instance nearest neighbors will be selected to generate the new synthetics.
- The creation of each new synthetic is mainly done by finding the difference between the randomly selected minority instance and one of its nearest neighbors, multiplying the difference with a random number between 0 and 1, and then adding the new value to the minority instance.

Furthermore, Equation 8.1 presents the formula of generating new synthetics depending on two minority instances [198].

$$s = x + u\,.\,(x^r - x),\ 0 \leq u \leq 1 \tag{8.1}$$

where, s is a new generated synthetic, x^R is a randomly selected instance from the five nearest neighbors of the minority instance x. In addition, several measurements could be used to calculate the distance between each minority instance and the other instances that belong to the same class. Actually, all of the measurements are based mainly on subtracting each feature in the instance features' vector from its corresponding feature in the features' vectors of the other minority instance. The most common distance measurements used are Euclidean distance and Manhattan distance. Equation 8.2 represents the formula of the Euclidean distances [199].

$$d(p,q) = d(q,p) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} \tag{8.2}$$

Where d is the distance between the instances p and q, and n is the number of features in the instance features' vector.

8.2.1.2 *Borderline SMOTE*

It is another oversampling technique based on the standard SMOTE, but targeting only the borderline minority class instances, that is, the ones closer to the line separating the data of two classes [164]. In other words, Borderline SMOTE is an advanced version of SMOTE balancing technique, it targets the instances with a high probability to be misclassified, and then uses those instances to generate new synthetics. Thus, the output of the Borderline SMOTE concretely improves the performance of the classification.

Accordingly, the following steps comprising the procedure of Borderline SMOTE [115]:

- Identify the minority and the majority classes instances
- Find the 5 nearest neighbors of each instance that belongs to the minority class.
- Depending on the variety of the class labels of the nearest neighbors, the minority instance is intended into 3 categories:

  1. If all of the nearest neighbors of the selected minority instance are belong to the majority class, then the selected minority instance is a **noise** instance and it will be ignored due to the definite chance of being misclassified.
  2. If the amount of the minority neighbors is more than the majority neighbors, then the selected minority instance is **safe**; it will be classified correctly.

3. If the amount of the majority neighbors are more than the minority ones, then the selected minority instance will be on the borderline (**dangerous**).

- For each instance in the **dangerous** category, the five nearest instances that belong to the same class (minority) will be found.

- Randomly select one of the nearest minority instances to generate the new instances.

Moreover, the Equation 8.3 represent the formula of the Borderline SMOTE that used to generate a new synthetic.

$$s_j = P_i' + r_j * dif_j \tag{8.3}$$

where is s_j is a new synthetic, P_i' is a *dangerous* instance, r_j is a random number between 0 and 1, and dif_j is the difference between the **dangerous** instance and the selected positive nearest neighbors to generate a new synthetic [115].

8.2.1.3 *Support Vector Machine with SMOTE (SVM-SMOTE)*

In this case, SVM is applied in order to approximate the decision boundary and borderline. Then, for the instances far away from the borderline, an extrapolation technique is used to generate minority class instances. On the other hand, for the instances closer to the borderline an interpolation technique similar to SMOTE is used to generate the minority instances [166]. In other words, the main stages of the SVM-SMOTE procedure are listed as follows:

- Training Support Vector Machine (SVM) in order to find the borderline instances in the dataset.

- Depending on the density of the majority class instances around the borderline instances, the new synthetic will be generated using interpolations and extrapolation as explained in the follows:

  - If more than the half of nearest neighbors are majority instances, the new synthetics will be generated using extrapolation.

  - If If more than half of nearest neighbors are minority instances, the new synthetics will be generated using interpolation.

8.2.1.4 *ADaptive SYNthetic sampling approach (ADASYN)*

ADASYN uses a weighted distribution for different samples of the minority class according to their level of learning difficulty. That is, more synthetic data is generated for the samples of minority classes that are more difficult to learn [117]. Furthermore, the procedure's stages of ADASYN are listed as follows:

- Explore the data to determine the minority and majority instances, and calculate the balancing ratio.

- Depending on the balancing ratio, define the number of the new instances to be generated.

- Find the nearest neighbors for each instance that belongs to the minority class.

- Normalize the data.

- Generate new synthetics depending on the density of the data distribution; more density less new synthetics, whereas less density more new synthetics.

Furthermore, Equation 8.4 shows the formula to generate a new synthetic by ADASYN [117].

$$s_i = x_i + (x_{zi} - x_i) * \lambda \tag{8.4}$$

where s_i is a new synthetic, $(x_{zi} - x_i)$ is the difference vector in n dimensional spaces, and λ is a random number between 0 and 1.

8.2.1.5 *Modified SMOTE (MSMOTE)*

This is another powerful balancing technique developed as an enhancement of SMOTE by Hu et al. in 2009 [185]. Moreover, in order to enhance the quality of the obtained balanced data and improve the classification performance, MSMOTE groups the data instances into 3 categories, namely:

- Security samples: which improve the performance of the classifiers (easy to classify)
- Noise samples: which extremely decrease the performance of the classifier (very hard to classify)
- Border instances: which affect the performance of the classification depending on how the usage of it to generate new synthetics.

Moreover, the main difference between MSMOTE and SMOTE is the criterion of finding each instance's nearest neighbors depending on the three categories aforementioned. Accordingly, for each randomly selected instance from the minority class:

- If all of the instance's neighbors are belong to the same class (minority), then the selected instance is secured. Thus, MSMOTE finds the five nearest neighbors for the instance, and generates new synthetics using SMOTE formula.
- If all of the instance's neighbors are belong to the other class (majority), then the selected instance is noise and it will be ignored.
- If the neighbors of the selected instance are separated between two classes (majority and minority), then the selected instance is on the borderline. Thus, MSMOTE finds the nearest neighbor to the selected instance, and uses this neighbor with the selected minority instance to generate new synthetics.

8.2.1.6 *Safe Level-Synthetic Minority Oversampling TEchnique (Safe Level SMOTE*

It is another advanced balancing technique based on SMOTE in its procedure. Basically, it finds the safe level for each minority instance in the dataset in order to allocate a more convenient place to generate the new synthetics. The *safe level* of the minority instance is the number of its minority neighbors on k, whereas the *safe level ratio* is the result of dividing the *safe level* of the selected minority instance on the safe level of its minority neighbors [116]. Accordingly, *safe level* and *safe level ratio* are used to allocate the new synthetics very close to the largest *safe level*, which means that all of the new synthetics are located in safe regions. Moreover, let us consider that p is the randomly selected minority instance, n is a selected nearest neighbor of p. Safe Level SMOTE in its procedure follows one out of five cases, which are [116]:

1. If *safe level ratio* $= \infty$ and the *safe level* of $p = 0$, this means that both p and n are noise instances, then no new synthetics will be generated.

2. If *safe level ratio* $= \infty$ and the *safe level* of $p \neq 0$, this means that only n is noise, then the new synthetics will be generated by duplicating p to guarantee that it would be as far as possible from the noise instance n.

3. If *safe level ratio* $= 1$ and the *safe level* of p and n is same, then the new synthetics will be generated along the line between p and n (p and n are both safe as same as each other).

4. If the *safe level ratio* > 1, means that the *safe level* of p is greater than the *safe level* of n, thus, the new synthetics will be generated near p because it is safer than n.

5. If the *safe level ratio* < 1, means that the *safe level* of p is less than the *safe level* of n, thus, the new synthetics will be generated near n because it is safer than p.

8.2.1.7 *SMOTE-OUT*

It is an advanced balancing technique proposed by Koto in 2014 [186]. The main idea of SMOTE-Cousin is to avoid the major drawback of SMOTE. In other words, considering that there are two minority instances that are very close to each other, then the line that is supposed to be used to create the new synthetics will be very short, and some minority instances would be duplicated as it is. This case might cause overfitting of the data to the classifiers. Thus, SMOTE-OUT generate the new synthetics out of the line between a pair of very close minority instances, but as far as possible from the nearest majority instance. Let u is a minority instance and the center of the circle that SMOTE-OUT will generate the new synthetics in, a is the distance of between u and the nearest minority class neighbor x, and b is the difference between u and the nearest majority class instance v ($b = u - v$), then:

- In the case of a is greater than or equals b, the distance between u and x will be considered as the radius of the circle that will be used to create the new synthetics in.
- In the case of a is less than b, b will be considered as the radius of the circle that will be used to create the new synthetics.

.

8.2.1.8 *SMOTE-Cousin*

It is an extension of the previously mentioned balancing method (SMOTE-OUT), and proposed by the same author in [186]. The main difference between SMOTE-Cousin and SMOTE-OUT is the criteria to find the similarity between the minority instances and its nearest majority neighbor instance. In other words, the author proposed a new algorithm enhancing the performance of the euclidean distance in order to find the similarity between instances. Equation 8.5 shows the formula of the proposed similarity algorithm.

$$sim(u,v) = \frac{\sum u.v}{\sqrt{\sum u^2} * \sqrt{\sum v^2}} \qquad (8.5)$$

8.2.1.9 *Selected-SMOTE*

As in the previous subsection, Selected-SMOTE is also an extension of the previously mentioned balancing method (SMOTE-OUT), and proposed by the same author in [186]. The main idea of Selected-SMOTE is to select the most efficient attributes to be synthesized and avoid meaningless attributes. For instance, Selected-SMOTE synthesizes the attributes that could affect (positively) the performance of the classifiers and ignores the attributes that might lead to meaningless extra calculations.

8.2.1.10 *Gaussian-SMOTE*

It is a resampling approach proposed by Lee et al. [187] in order to avoid a major common drawback that could happen in several balancing techniques such as SMOTE, Borderline SMOTE and Safe Level SMOTE. In other words, the other balancing techniques in their procedures could have a drawback called the "over-generation" problem, in which the balancing technique might generate many synthetics between the minority instance and only one of its K neighbors. Thus, Gaussian-SMOTE solves this problem by using the Gaussian algorithm to increase the variate of the generated synthetics.

8.2.1.11 *Surrounding Neighborhood-based SMOTE (SN-SMOTE)*

This is another variant of SMOTE proposed by Garcìa et al. in 2012 [200]. The main enhancement carried out by SN-SMOTE over SMOTE method is the way to select the nearest neighbors of the minority instances in the dataset. In other words, instead of using the normal Euclidean distance to find the nearest neighbors as happens in SMOTE, SN-SMOTE proposed an alternative to find the nearest neighbors, and might extend the region to create the new synthetics. Thus, the alternative used to find the nearest neighbors that considered by SN-SMOTE is Nearest Centroid Neighborhood (NCN) [201] which is a concept to select the nearest neighbor of a certain minority instance. The output K nearest neighbors of NCN are as close to the selected minority instance as possible, and the centroid of it is also close as possible to the minority instance.

8.2.1.12 *Locally Linear Embedding SMOTE (LLE-SMOTE)*

It is an advanced sampling technique proposed by Wang et al. [202]. On the contrary to SMOTE, LLE-SMOTE guarantees that the generated new synthetics will be located as far as possible from the majority instances in the dataset. The main difference between SMOTE and LLE-SMOTE procedures is using the Locally Linear Embedding (LLE) in LLE-SMOTE to reduce the dimensionality of the data by mapping it into linearly separable one, and then using the normal SMOTE to oversample the minority class instances.

8.2.1.13 *Deterministic Version of SMOTE (SMOTE-D)*

As aforementioned in its name, it is a deterministic version of SMOTE proposed by Torres et al. in 2016 [203]. The main steps of SMOTE-D procedure are summarized as follows:

- Explore the dataset, and calculate the amount of the minority and majority class instances.

- Calculate the number of synthetics to be generated depending on the required balancing ratio (e.g., 50% of the balanced dataset belong to the minority instance and the other 50% belong to the other class).

- Find K nearest neighbors for each minority instance, and then find the distances between the selected minority instance and its neighbors.

- Calculate the standard deviation of the distances between each minority instance and its neighbors.

- Calculate the fraction of the standard deviation of each minority instance from the total standard deviation.

- Calculate the fraction of each distance between the selected minority instance and its K neighbors from the sum of the instances between the same minority instance and all of its K neighbors.

- Find the number of instances that suppose to be generated between the minority instance and each one of its neighbors.

- Divide the feature difference between each minority instance and each one of its neighbors by the number of synthetic instances to be generated from those two instances, and then add 1 to the yielded number.

- Finally, add the results calculated in the previous step to the selected minority instance as many as generating new synthetics.

8.2.1.14 *Automatic Neighborhood size Determination in SMOTE (AND-SMOTE)*

It is a modification of SMOTE balancing technique proposed by Yun et al. in 2016 [204]. However, in AND-SMOTE the Automatic Neighborhood size Determination (AND) is used to determine the ideal minority instances nearest neighbors that increase the effectiveness of the generated synthetics during the classification even if the dataset contains noises.

8.2.1.15 *SMOTE with Local Neighbourhood Extension (LN-SMOTE)*

It is a modification of SMOTE proposed by Maciejewski and Stefanowski in 2011 [205]. The main idea of the LN-SMOTE is to avoid some drawbacks in the other existing balancing methods in that time such as SMOTE and Safe Level SMOTE procedures. However, in the contrary of SMOTE and Safe Level SMOTE methods, LN-SMOTE pays more attention to the distribution of the majority instances in the dataset set by using the Local Neighbourhood (LN) extension. Thus, the LN-SMOTE method avoid the long distant neighbors exploring by looking for the local neighbors of the selected minority instance regardless to the class labels of them, and then create the new synthetics between the minority instance and its minority and majority neighbors.

8.2.1.16 *Combination of SMOTE and Particle Swarm Optimization (SMOTE-PSO)*

It is another enhancement of SMOTE method, proposed in 2017 by Cervantes et al. [206]. SMOTE-PSO is developed in order to improve the performance of Support vector machine in classifying extremely imbalanced dataset. Thus, depending on the density of the minority class instances, SMOTE-PSO generates the new synthetics in the regions that have less minority instances density.

8.2.1.17 *Neighborhood Triangular SMOTE (NT-SMOTE)*

It is an improved version of SMOTE proposed by Xu et al. [207] in 2014. Thus, NT-SMOTE improves the performance of SMOTE by considering using k nearest neighbors and triangular area sampling approaches in its procedure. The steps of the NT-SMOTE procedure are listed as follows:

- For each minority instance belongs to the minority class, find two nearest neighbors.
- Randomly select a data point in the triangle and consider it as a new instance.
- Randomly select a minority instance and calculate the distance between it and the whole instances in the dataset.
- Select the two nearest instances to the data point found in the second step, and generate new synthetics between these two instances by linear interpolation algorithm.

8.2.1.18 *Distance-SMOTE*

It is a modification of SMOTE balancing technique proposed by Calleja and Fuentes [208]. Distance-SMOTE is based on the feature space instead of the data space as occurs in SMOTE method. Thus, the steps of the Distance-SMOTE method in handling the data inconsistent distribution problem are listed as follows:

- Explore the dataset to recognize the minority and the majority instances.
- Find the k nearest minority neighbors to each minority instance in the dataset; the instance and the neighbors are belong only to the minority class.
- For each minority instance, calculate the mean of its minority instances in order to generate *mean instance*.
- Calculate the difference between the selected minority instance and the mean instance, and then multiply the results

by a random number between 0 and 1 to generate a new synthetic.

- Add the new generated synthetics to the dataset.

8.2.1.19 *Majority Weighted Minority Oversampling TEchnique (MW-MOTE)*

It is another modification of SMOTE proposed by Barua et al. [209] in 2012. However, the main motivation of MWMOTE is to improve the performance of the existing balancing techniques at that time by identifying the hard-to-classify minority instances, and assigning a certain weight to each of them depending on the euclidean distances with the nearest majority instances. MW-MOTE uses the weighted minority instances to generate new synthetics in the minority class region. Accordingly, the main procedure steps of MWMOTE are summarized as follows:

- Explore the dataset to identify the minority and majority classes instances.

- Set the number of the new synthetics to be generated with respect to the required balancing ratio.

- Find the *k* nearest neighbors for each minority instance in the dataset.

- Eliminate the minority instances that don't have any minority instance in their nearest neighbors (noise).

- For each minority instance (after eliminating the noises), find the nearest majority instances using the euclidean distance.

- Set importance weights to each minority instance depending on the distance with the nearest majority instances.

- Use the minority instances with the highest importance weights to generate new synthetics to guarantee that the new synthetics will be located in the minority class cluster.
- Insert the new generated synthetics in the dataset version obtained after eliminating the noises.

8.2.1.20 *Learning Vector Quantization based SMOTE (LVQ-SMOTE)*

This balancing technique was proposed by Nakamura et al. [210] in order to improve the performance of the existing oversampling methods. Basically, LVQ-SMOTE devoted using Learning Vector Quantization (LVQ) which is a supervised version of *k*-means algorithm to find several centroids (codebooks) for each feature constructing the instances in the dataset, then generating new synthetics according to the similarity of the codebooks with the minority instances in the dataset.

8.2.1.21 *Adjusting the Direction Of the synthetic Minority clasS isntances (ADOMS)*

It is a modification on the SMOTE method proposed by Tang and Chen [118] in 2008. However, ADOMS uses Principal Components Analysis (PCA) technique to explore the data structure variance in order to improve the quality of the generated synthetics. Thus, the main steps of ADOMS in solving the data balancing problem are listed as follows:

1. Explore the dataset in order to identify the majority and minority classes instances.
2. Randomly select one of the minority instances.
3. Find *k* neighbors to the selected minority instance.

4. Calculate the variance between the selected minority instance and its k neighbors using PCA.

5. Calculate the euclidean distance between the minority instance and one of its neighbors.

6. Generate new synthetics along the line between the minority instance and the randomly selected neighbor with respect to the variance value obtained in step (3).

8.2.2 *Hybrid balancing methods (oversampling-undersampling)*

Basically, Several hybrids (oversampling-Undersampling) have proven their performance in the literature in solving the data balancing problem. Those classifiers are based mainly in their procedure on oversampling the minority instances by creating new synthetics that belong to the same class, and undersampling the majority instances. The compared hybrid balancing methods are described as follows:

8.2.2.1 *SMOTE with Edited Nearest Neighbor (SMOTE-ENN)*

It is one of the most powerful balancing methods. SMOET-ENN combines the oversampling using SMOTE and the undersampling using ENN. The ENN step calculates the nearest k neighbors of each instance, and if most neighbors are of a different class, it eliminates the instance [111]. In other words, the procedure of the SMOTE-ENN stands mainly on 2 steps:

- Use SMOTE to balance the dataset.

- Use the Edited Nearest Neighbor (ENN) to undersample the dataset by finding the five nearest neighbors of each instance in the balanced dataset (minority and majority), and then eliminate the instances that have 3 or more neighbors that belong to the other class.

8.2.2.2 *SMOTE with Tomek links (SMOTE-Tomek)*

It is another hybrid (oversampling-undersampling) high performance balancing technique. The first stage of the SMOTE-Tomek procedure is oversampling the minority class instances to be equal to the majority ones, the next stage is the data cleaning (undersampling) by eliminating Tomek Links. Tomek Links represent pairs of data instances, each instance in those pairs belongs to a different class (i.e., one instance belongs to minority class, and the other instance belongs to the other class) [111]. Moreover, the main conditions to select the instances to construct the Tomek Link are:

- Each instance belongs to a different class; one minority instance and one majority instance.
- Both instances are very close to each other; the distance between the instances in each Tomek Link suppose to be the lowest compared to the distance between any instance in the Link and the remaining instances in the dataset.

8.2.2.3 *Improved SMOTE (ISMOTE)*

It is a hybrid balancing technique(oversampling-undersampling) proposed as an enhancement on SMOTE. In other words, ISMOTE in its procedure based on oversampling the minority instances and undersampling the majority instances using Distance-Based Undersampling method. Furthermore, during generating the new synthetics, SMOTE normally assigned the same weights for all of the neighbors of each minority instance in the dataset, whereas ISMOTE assigns a certain weight for each neighbor depending on the class labels; minority neighbors take higher weights, while the majority neighbors take lower weights [211].

Accordingly, the main stages of ISMOTE procedure are described as follows [211]:

- Oversampling:
  - Explore the dataset in order to determine the minority instances.
  - Find the distance between each minority instance and the whole instances in the dataset.
  - Select the nearest neighbors for each minority instance.
  - Generate new synthetics between each minority instance and its nearest neighbors using the following formula: $S = x_i + Random(0, 1) * (y_i - x_i) * w(y_i)$, where x_i is the randomly selected minority instance, y_i is the randomly selected neighbor, and w is the weight set for the neighbor y_i.
  - Insert the new generated instances into the dataset.
- Undersampling:
  - Explore the dataset and find the instances that belong to the majority class.
  - Calculate the distance between each majority instance and the whole minority instances in the dataset.
  - Compute the average of the distances between each majority instance and the minority instances.
  - If the found average is higher than a predefined threshold, then the selected majority instance will be eliminated.

8.2.2.4 *Improved SMOTE method based on Support Degree (SDSMOTE)*

It is a hybrid (oversampling-undersampling) balancing technique proposed by Li et al. in 2014 [194]. Moreover, SDSMOTE uses normal SMOTE for oversampling the minority instances, and Neighbor Cleaning Rule (NCR) [212] for undersampling. Thus, the procedure of SDSMOTE is explained as follows:

- Set the amount of minority synthetics that should be generated.
- Select a minority instance randomly, find the distances between the selected instance and every majority instance in the dataset, and then calculate the sum of those distances (S).
- Find the summation of all sums (S) that calculated in the previous step.
- Calculate the average distance between the minority instances with the minority instances.
- Find the support degree by drawing a circle covering the selected minority instance and a certain distance as the center and the radius, then count the number of minority instances that covered by the drawn circle and consider it as the support degree of the minority instance; larger support degree higher probability that the instance will be determined as a boundary instance.
- The instances with the highest probability to be determined as boundary instances will be used to generate more new synthetics compared to the instances with the lowest probability.
- Insert the new generated instances to the dataset.

8.2.2.5 *Partially guided hybrid sampling approach (G-SMOTE)*

It is another enhanced version of SMOTE balancing technique, proposed by Sandhan and Choi in 2014 [213]. In their approach, they devoted the bootstrapping method to concurrently oversampling the minority instances and undersampling the majority instances, and creating an ensemble of classifiers.

8.2.2.6 *SMOTE with Iterative-Partitioning Filter (SMOTE-IPF)*

It is an extension of SMOTE balancing technique, proposed by Sáez et al. in 2015 [214]. Basically, SMOTE-IPF uses SMOTE to generate new minority synthetics in order to balance the dataset, and then uses Iterative-Partitioning Filter (IPF) to eliminate the noise that already exists in the original dataset before oversampling, and also the noise synthetics generated by SMOTE. Moreover, the procedure's steps of the IPF cleaning technique are listed as follows:

- Split the balanced dataset into several equal-size partitions.
- Apply C4.5 classifier to each partition, and use the results to evaluate the whole balanced dataset.
- Find the noise instances identified in the previous step using a voting scheme.
- Eliminate the noise instances found in the previous step.

8.2.2.7 *SMOTE with Rough Sets theory (SMOTE-RSB$*$)*

It is an extension of SMOTE, proposed by Ramentol et. al [215] to solve the overgeneralization problem that happens during balancing the dataset using SMOTE. The overgeneralization problem can be defined as generating the new minority synthetics in the region of the minority instances. Accordingly, the stages of SMOTE-RS$B*$ procedure are summarised as follows:

- Balancing the dataset by generating new synthetics belonging to the minority class using SMOTE technique.

- Improving the quality of the generated minority synthetics in the previous stage using editing techniques based on the Rough Set Theory (RST)[216].

8.2.2.8 *Adaptive Neighbor Synthetic minority oversampling technique under 1NN outcast handling (ANS)*

It is an oversampling technique based on the consist of SMOTE method, proposed by Siriseriwan and Sinapiromsaran [195] to handle some drawbacks of several existing sampling methods. However, the main enhancement held by ANS was selecting optimally the value of the k parameter during finding the nearest neighbors; inappropriate selection of the nearest neighbor parameter may lead to locating the new synthetics in the majority class region, because the nearest neighbor might be located far away from the minority instance (In the majority class region). In other words, ANS selects only the nearest minority instance to generate the new synthetics. On the other hand, ANS eliminates the minority instances outliers from the original dataset, which are the minority instances surrounded by majority instances and might cause locating the new synthetics in the majority region.

8.2.2.9 *SPY*

It is an advanced balancing technique proposed by Dang et al. [217] in 2015. As with all of the aforementioned balancing techniques, SPY also provides improvements on some existing balancing methods in certain cases. The main idea of SPY is to revert the class labels of some majority instances depending on its place in the minority instances region, which is defined by a threshold of the number of majority neighbors to each minority ones. Thus, the steps of SPY during balancing the dataset is listed as follows:

- Identify the majority and minority classes instances in the dataset.
- Find the k nearest neighbors for each instance minority instance.
- If the number of the majority neighbors is less than the predefined threshold of the neighboring majority instances, revert the class of the majority neighbors to be minority ones.

8.2.3 *Clustering-based balancing methods*

Other balancing techniques based in their procedures on clustering to identify the majority and the minority instances regions before oversampling. The compared clustering-based balancing techniques are described as follows:

8.2.3.1 *Cluster Ensemble Synthetic Minority Oversampling TEchnique (CE-SMOTE)*

It is another variant of SMOTE, proposed by Chen et al. in 2010 [196]. Basically, CE-SMOTE in its procedure creates an ensemble of clusters to allocate the boundary to generate the new synthetics. The steps of the CE-SMOTE are listed as follows:

- In the first step, select the features that will be used to create clusters, randomly select the centers of the clusters, and use K-means method to create partition from the training dataset (unsupervised learning).

- Match the created clusters.
- For each instance in the training dataset (class attribute is excluded), compute the cluster consistency index.
- Find the minority instances that have cluster consistency index less than a predefined threshold, and put those instances into a certain set named *P-Boundary*.
- Oversampling the minority instances that located in *P-Boundary* with respect to the predefined sampling rate.
- Insert the new generated samples into the training dataset.

8.2.3.2 *Clustering-Based Implementation of SMOTE (Cluster-SMOTE)*

Cluster-SMOTE is an oversampling approach proposed by Cieslak et al. [197]. Normally, the imbalanced dataset contains a very low amount of minority instances, this situation increases the difficulty of finding the minority-majority classes' borders, then leads the oversampling methods, such as SMOTE, to locate the new synthetics in an inappropriate distribution. Thus, Cluster-SMOTE solves this problem by using k-means clustering to find the minority instances in the dataset, and then applying SMOTE to each cluster to generate the new synthetics.

8.2.3.3 *Cure-SMOTE*

It is a clustering-based SMOTE variant proposed by Ma and Fan in 2017 [188]. CURE-SMOTE clusters the dataset using CURE (Clustering Using Representatives), which is a hierarchical clustering algorithm that can be used with different data shapes, and efficient for large datasets. Thus, the steps of CURE-SMOTE are described as follows:

- Normalize the dataset and determine the minority class instances.
- Calculate the distance between each minority instance and the other instances that belong to the same class.
- Consider each minority instance as an individual cluster.
- Merge the clusters depending on the distance between each pair of clusters, and then update the center and the representative points.
- Generate new synthetics between the center point and the representative points.

8.2.3.4 *Agglomerative Hierarchical Clustering (AHC)*

It is another variant of SMOTE method proposed by Cohen et al. [121]. AHC is a clustering algorithm used to handle the data inconsistent distribution problem. In other words, AHC considers each minority instance as an individual cluster, and then calculates the distances between each cluster and the rest clusters, after that, merge the clusters that are located very close to each other (minimum distance). Removes the original clusters merged from the data set and add the resulted cluster, then iteratively repeat the process until reaches the required number of clusters. To overcome the imbalance issue in the data set, those clusters are used as a prototype to create synthetic examples.

8.3 DISTANCE-BASED BORDER INSTANCES SMOTE (DBBI-SMOTE)

Basically, DBBI-SMOTE method is proposed to handle the inconsistent data distribution problem. Similar to SMOTE method and all of its variants, DBBI-SMOTE method generates new synthetics from the minority data instances to obtain a balanced version of the original dataset in order to improve the performance of the classifiers. SMOTE method is a powerful advanced balancing technique, proved its feasibility in wide research areas regarding improving the performance of the ML classifiers, but that does not preclude the occurrence of some hiccups in its procedure depending of the original data distribution states. In other words, SMOTE in its procedure randomly selected a minority instance from the dataset, find the k neatest neighbors for this instance, and then generates the new synthetics in the hyperplane between the selected minority instance and its neighbors. Moreover, the procedure of SMOTE might lead to creating some of the new synthetics in the region of the majority class in case the selected minority instance is located in that region. Figure 8.1 illustrates the case of SMOTE in which creating the new synthetics in the majority region. Furthermore, Borderline

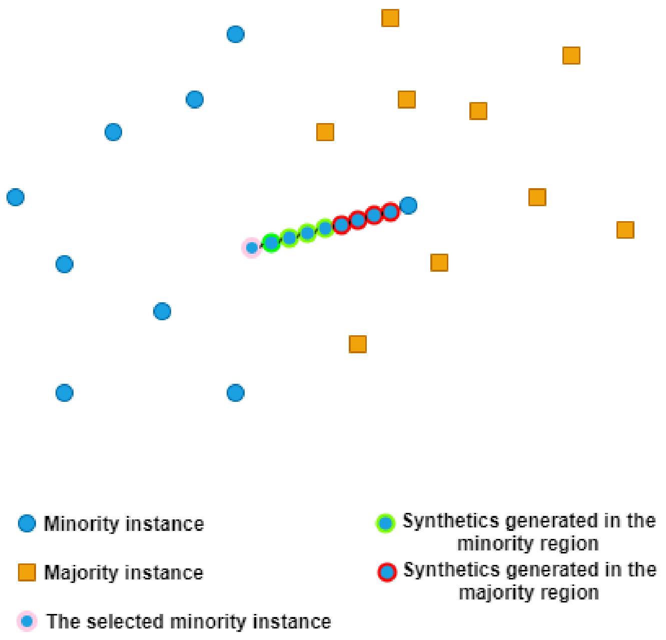
Minority instance
Majority instance
The selected minority instance
Synthetics generated in the minority region
Synthetics generated in the majority region

Figure 8.1: The data distribution case that leads SMOTE to create new synthetics in the region of the majority class.

SMOTE is another well-known balancing technique used widely in the literature to solve the data distribution problem [218–220].

Thus, Borderline SMOTE enhances the procedure of SMOTE by focusing on the instances that are hard to classify and use only them to generate the new synthetics which will be in charge to improve the performance of the classifier more than SMOTE. The hard-to-classify instances are called "borderline instances", which are located in the borderline between the minority and the minority region. Technically, borderline instances are defined as the instances that have one or more instances belonging to the other class joining their nearest neighbors. Furthermore, the procedure of Borderline SMOTE does not cope with all of the data distributions, which creates a major drawback and challenge to the balancing method. For example, if the minority instances region is far from the majority one (see figure 8.2), which means that all of the minority instances are safe (no borderline instances). This case pretty definitely cause a major failure in borderline SMOTE.

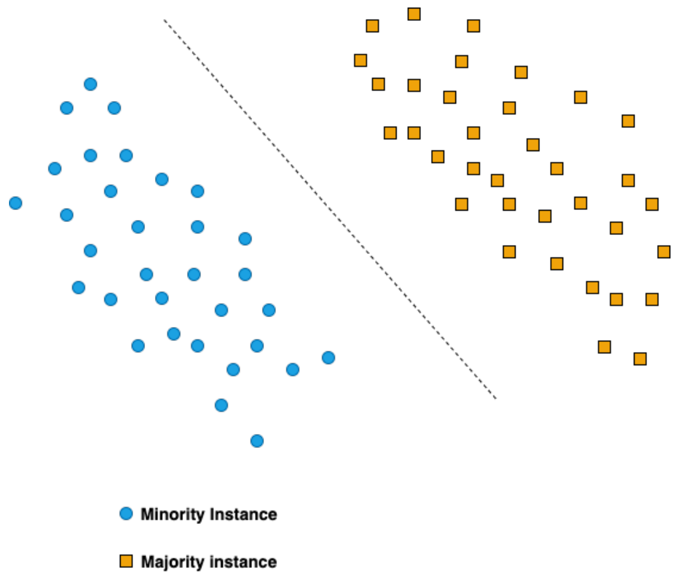

Figure 8.2: The data distribution case which causes a failure in Borderline SMOTE procedure.

Thus, we propose the Distance-Based Border Instances SMOTE (DBBI-SMOTE) to avoid the drawbacks of some existing balancing techniques (i.e., SMOTE, Borderline SMOTE and MSMOTE), and improve the overall quality of the generated synthetics. Actually, DBBI-SMOTE avoids the major drawback of SMOTE by taking into consideration the location of the majority instances in the data distribution. It analyses the data in order to find the nearest majority instance to each minority instance in the dataset, and try to locate the new synthetics as far as possible from the neighbor majority instance. Accordingly, this procedure guarantees versatility of the generated new synthetics, but with retaining the essential properties of the minority class instances. On the other hand, DBBI-SMOTE avoids the drawback of Borderline SMOTE as well. Unlike Borderline SMOTE, Due to the limitation of the minority instances in the imbalanced data, each one of the minority instances presents valuable information for the classifiers, thus, DBBI-SMOTE gives the same important factor to all of them, and use them all to generate new synthetics. In addition, drawing on the fact of the importance of each single minority instance in the dataset, and taking into consideration that a large portion of the minority instances might be noise, DBBI-SMOTE pays more attention to the noise samples that belong to the minority class, by increasing the number of its nearest neighbors to look for in order to reduce the probability to discard that instances and waste probably important information to the classifier. Table 8.1 shows the amount of the majority, minority and noise instances in the Spanish, Taiwanese, Polish and US companies' datasets considering that the k parameter set as the default of several balancing techniques (k = 5), and after extending the amount of the nearest neighbors (k = 10). Accordingly, as it can be seen in the table, a large portion of the minority instances in the four datasets are noise, thus, discarding those instances may lead to lost important information for the classifiers.

Table 8.1: The data distribution in the companies datasets adopted to evaluate the performance of DBBI-SMOTE.

Dataset	**Majority**	**Minority**	**Safe**	**Borderline**	**Noise (k = 5)**	**Noise (k = 10)**
Spanish Companies	2797	62	0	39	23	12
Taiwanese Companies	6599	220	0	66	154	81
Polish Companies	9797	203	32	130	41	14
US Companies	92314	558	2	217	339	244

The procedure of DBBI-SMOTE comprises several scenarios to handle the data balancing problem depending on the distribution of the majority class instances and how they are close to each minority instance. The *first scenario*: if the k nearest neighbors of a randomly selected minority instance belongs to the same class, then use SMOTE to generate new synthetics between the selected instance and a random one of its neighbors. The *second scenario*: if the nearest neighbors of the selected minority instance are a mixture of minority and majority instances, calculate the euclidean distance between the selected minority instance and each one of its neighbors, recognize the location of the nearest majority instance, then use the minority instances that closer to the selected minority instance than the majority one to generate the new synthetics. Figure 8.3 illustrate the behavior of DBBI-SMOTE to create the synthetics between the minority instances depending on the location of the nearest majority neighbor. On the other hand, if the majority instance is the nearest neighbor to the selected minority instance, then use one of the minority neighbors to generate the new synthetics with the selected minority instance taking into account placing the new instances as far as possible from the majority instance; very close to the selected minority instance or to the minority neighbor depending on which one of them is farther from the majority neighbor. Figure 8.4 illustrate the behavior of DBBI-SMOTE in case the nearest neighbor belongs to the majority class. The *third scenario*: if all of the k nearest neighbors of the selected minority instance belong to the majority class, which means that the selected minority instance is noise, then DBBI-SMOTE gives that selected minority instance more chance to avoid losing probably important information. DBBI-SMOTE increases the value of the nearest parameter k up to 10 for the noise instances, then if one new extended neighbor belongs to the minority class, that minority neighbor will be used with the selected minority instance will be used to generate new synthetics, but with taking care to locate the new synthetics as close as possible to the selected minority instance or neighbor. On the other hand, if all of the neighbors still belong to the majority class after increasing the nearest neighbors parameter k, then the selected minority instance will be discarded.

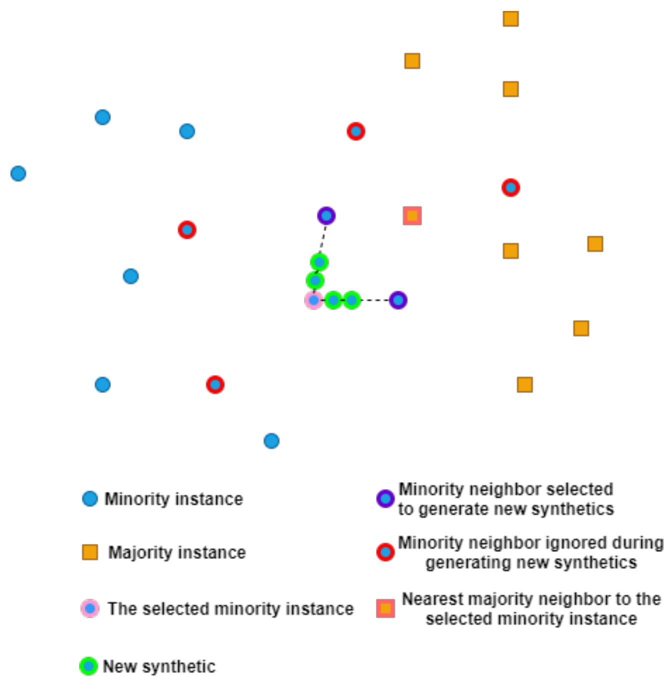
Minority instance
Majority instance
The selected minority instance
New synthetic
Minority neighbor selected to generate new synthetics
Minority neighbor ignored during generating new synthetics
Nearest majority neighbor to the selected minority instance

Figure 8.3: The procedure of DBBI-SMOTE in locating the new synthetics with respect to the nearest majority neighbor.

Suppose the training dataset T comprising of X minority and Y majority instances, and

$$X = \{x_1, x_2, x_3, ..., x_{xnum}\}, \; Y = \{y_1, y_2, y_3, ..., y_{ynum}\} \qquad (8.6)$$

where $xnum$ is the number of the minority instances, and $ynum$ is the number of majority instances. Thus, the detailed procedure of DBBI-SMOTE is as follows:

- Explore the training dataset to identify X and Y.
- Depending on the balancing ratio of T, find the amount of new synthetics to be generated.
- Randomly select x_i from X, and find the k nearest neighbors N_i of it, where $N_i = \{n_1, n_2, .., n_k\}$.

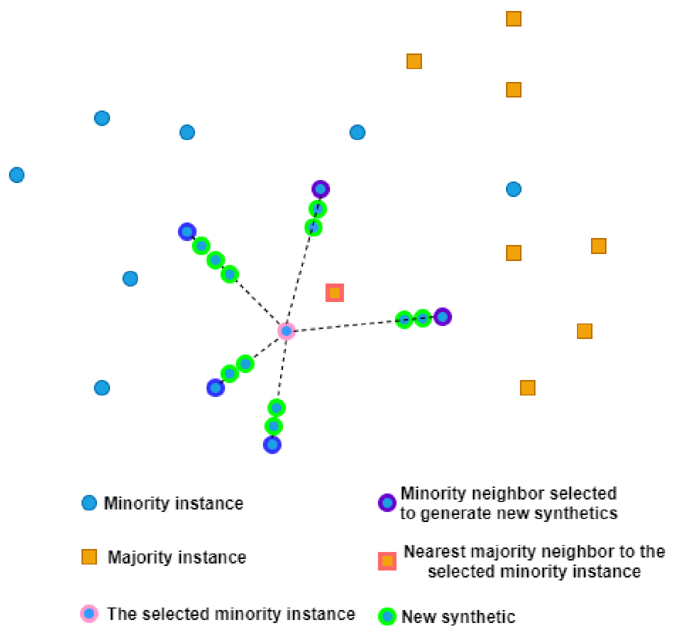
Minority instance
Majority instance
The selected minority instance
Minority neighbor selected to generate new synthetics
Nearest majority neighbor to the selected minority instance
New synthetic

Figure 8.4: The procedure of DBBI-SMOTE in locating the new synthetics if the nearest neighbor of the selected minority instance belongs to the majority class.

- If N_i contains only minority instances, then use SMOTE to generate new synthetics in the hyperplanes between x_i and randomly selected instances from N_i.

- If N_i is a mixture of minority and majority instances, identify the location of the nearest majority neighbor y_i to x_i, and then one of the following procedures will be implemented:

  – If N_i contains one or more instances belongs to X, and closer to x_i than y_i, then those minority instances will be used with x_i to generate new synthetics with attention to place it as far as possible from y_i.

  – If y_i is the nearest neighbor to x_i, then select one minority instance from N_i and use it with x_i to generate new synthetics with attention to place it very close

x_i or to the selected instance from N_i to be as far as possible from y_i.

- If all of the instances in N_i belong to Y, increase the nearest neighbor parameter k up to 10, then:
    - If the new added instances to N_i including minority instances, then select one of those minority instances and use it with x_i to generate new synthetics with attention to place it very close to x_i, or to the selected minority neighbor to be as far as possible from the majority neighbors.
    - If N_i including only majority instances after the size extension, then discard x_i.
- Insert the new generated synthetics into T.

8.4 EXPERIMENTS AND ANALYSIS

In this section, we present the performance evaluation requirements in detail. The main procedure to evaluate the performance of the proposed balancing method (DBBI-SMOTE) is to apply it as a preprocessing stage of classification using several well-known classifiers, namely: Decision trees, Naive bayes, KNN, SVM, RF, AdaBoost and XGBoost. In addition, the obtained results from DBBI-SMOTE are compared to the results yielded by 34 other balancing techniques have used as preprocessing stage to the same classifiers. However, the main reason to choose these classifiers to evaluate the performance of all balancing techniques is the variety in their behaviors during classification. In other words, we select these classifiers particularly to cover all of the machine learning classifiers categories which are: standard classifiers (Naive bays, Decision trees), Bagging-based ensemble classifiers (KNN,SVM, RF) and boosting-based ensemble classifiers (AdaBoost and XGBoost). In addition, several evaluation metrics have been adopted to evaluate the performance of each combination of a classifier and balancing method.

8.4.1 *Experimental Datasets*

Actually, to evaluate the performance and the feasibility of the new DBBI-SMOTE method comprehensively, several extremely imbalanced companies' datasets used to be processed by DBBI-SMOTE and the other balancing method as a preprocessing stage to the classifiers. Accordingly, four companies' datasets were selected carefully respecting the data type variate, balancing ratios, and the data distribution in each one of them. Thus, the first dataset is for Spanish companies, consisting of 2797 instances (98%) belonging to the majority class and the remaining 62 instances (2%) belonging to the minority class. Each instance is comprising 27 numeric attributes , and 6 categorical ones. The second dataset is for Taiwanese companies, consisting of 6599 instances (98%) belonging to the majority class and the rest corresponding to the minority class (220 instances). In addition, Each instance in the Taiwanese companies dataset is comprising 95 numerical financial attributes. The third dataset is for Polish companies, containing 9797 belonging to the majority class (97.97%), and 203 are minority instances (2.03%). Each instance is comprising 64 numerical financial attributes. More details about the Spanish, Taiwanese and Polish companies' datasets are addressed in the previous chapter, specifically, in section 7.2. Moreover, to increase the challenge for the balancing techniques and classifiers, we consider using a dataset much more complicated than the aforementioned datasets. The fourth dataset is for US companies, contains 92314 instances belonging to the majority class (99.3%), and 585 instances belonging to the minority one (0.7%). Each instance is comprised of 14 numeric financial attributes. Thus, regarding the complexity, the US companies dataset is a very complex dataset; it is approximately 10 times the size of the polish companies dataset, and it is extremely imbalanced. It can be downloaded from the *Kaggle* ML community web [1].

8.4.2 *Performance Measure*

Also in this chapter, based on the fact that the accuracy measure is not enough to evaluate the performance of the classifiers in the case of imbalanced data distribution, we have used the same evaluations measures (metrics) that devoted in the previous chapter to evaluate the performance of the Machine Learning and Deep-Learning algorithm in solving real-world problem (i.e., predicting companies financial failure). Thus, the main measures based on the confusion matrix and used in this chapter to evaluate the performance of DBBI-SMOTE and the other balancing techniques are summarized as follows:

- Accuracy [168]: Performance of the classifier in assigning the correct class to each instance.

- Recall [169]: Performance of the classifier in assigning each instance to the minority class (prediction) while it is actually minority (real status).

- Specificity [169]: Performance of the classifier in assigning every instance to the majority class (prediction) while it is actually majority (real status).

- Precision [169]: Performance of the classifier in assigning the correct class to each instance, whether majority or minority.

- Type I error [168]: The failure of the classifier to assign minority instance to the minority class (wrong prediction), while it actually belongs to the minority class (real status).

1 https://www.kaggle.com/shuvamjoy34/us-bankruptcy-prediction-data-set-19712017

- Type II error [168]: The failure of the classifier in assigning majority instances to the majority class (wrong prediction), while it actually belongs to the majority class (real status).

- Area Under ROC Curve (AUC) [182]: This is a metric adopted to evaluate the classification and regression models' performance. Receiver Operating Characteristics (ROC) curve is a graph showing the performance of classifiers and regression models by means of a series of thresholds.

However, more details about the adopted evaluation metrics are discussed in the previous chapter, specifically, in 7.3.1.

8.4.3 *Experimental Results analysis*

In this subsection, the impact of the proposed balancing method (DBBI-SMOTE) on several classifiers' behaviors will be discussed, and compared with several sampling methods covering all categories of the resampling approaches that existed in the literature [112, 185, 188, 194, 196, 214, 217]. However, as the balancing methods are considered as preprocessing stage for the data before training the classifier, we have applied DBBI-SMOTE and the other balancing technique to the datasets considered in this chapter. Table 8.2 shows the distribution of the majority and minority classes instances in the original Spanish, Taiwanese, Polish, and US companies' datasets, and also in the generated balanced dataset that obtained by each balancing method. Thus, as it can be seen in the table, the data distribution in the balanced dataset depends mainly on the type of balancing method, some balancing methods such as SMOTE, BL-SMOTE and MSMOTE are following the oversampling approach, increasing the minority instances to be equal to the majority ones; 50% of each balanced dataset is minority instances, and the other 50% is majority ones. On the other hand, it is obvious the difference of the majority and minority instances distribution in the balanced datasets obtained by the hybrid (oversampling-undersampling) approaches. Each hybrid approach has its own criteria in oversampling and undersampling, and it is not mandatory for the generated balanced datasets to be divided equally between the two classes (i.e., 50% minority and the other 50% majority). In addition, the clustering-based balancing methods such as Cluster-SMOTE and CURE-SMOTE follow only the oversampling approach to generate a balanced dataset divided equally between the two classes. Furthermore, it is important to note that two of the compared balanced methods didn't succeed in solving the data inconsistent distribution problem. SPY method failed to balance all of the considered companies' datasets. This means that the procedure of SPY is not suitable for such kind of data. On the other hand, CE-SMOTE failed to set a cluster boundary in the Polish and US companies' datasets, which means that CE-SMOTE is not convenient to process large datasets.

However, The obtained balanced dataset from each balancing technique was used individually to train several classifiers covering all of the classification methods categories, namely: standard classifiers (i.e., DT and NB), bagging-based ensemble classifiers (i.e., KNN, SVM and RF), and boosting-based ensemble classifiers (i.e., AdaBoost and XGBoost).

Table 8.2: The distribution of the majority and minority classes instances in the four datasets considered in this chapter (i.e., Spanish companies, Taiwanese companies, Polish companies and US companies datasets), and used to evaluate the performance of DBBI-SMOTE and the other resampling methods.

		Spanish companies' data		*Taiwanese companies' data*		*Polish companies' data*		*US companies' data*	
	Balancing techniques	Bankrupt	Solvent	Bankrupt	Solvent	Bankrupt	Solvent	Bankrupt	Solvent
	Original dataset	62	2797	220	6599	203	9797	558	92314
Oversampling	DBBI-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	BL-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	SVM-SMOTE	1346	2797	2308	6599	4708	9797	48224	92314
	ADASYN	2806	2797	6523	6599	9860	9797	92314	92314
	SL-SMOTE	2797	184	6599	1468	9797	1418	92314	2074
	AND-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	Distance-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	Gaussian-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	LLE-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	LN-SMOTE	2797	2797	6599	220	9797	9797	92314	92314
	LVQ-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	MSMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	MWMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	NT-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	Selected-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	SMOTE-Cosine	2797	2797	6599	6599	9797	9797	92314	92314
	SMOTE-D	2797	2797	6599	6605	9797	9798	92314	92323
	SMOTE-OUT	2797	2797	6599	6599	9797	9797	92314	92314
	SMOTE-PSO	2797	2797	6599	880	9797	812	92314	1887
	SN-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	ADOMS	2797	2797	6599	6599	9797	9797	92314	92314
Oversampling -Undersampling	SMOTE-Tomek	2765	2765	6565	6565	9794	9794	92209	92209
	SMOTE-ENN	2651	2494	6260	5433	9757	8546	88719	91941
	ISMOTE	1429	1430	3409	3410	5000	5000	46436	46436
	SDSMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	G-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	SMOTE-IPF	2797	2797	6599	6599	9797	9797	92314	92314
	SMOTE-RS*B**	2797	2797	6599	224	9797	211	-	-
	ANS	2797	2797	6599	6599	9797	9797	92314	92314
	SPY	2792	67	6596	223	9797	203	92281	591
Clustering based Oversampling	Cluster-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	CE-SMOTE	2797	2797	6599	6599	9797	203	92314	558
	CURE-SMOTE	2797	2797	6599	6599	9797	9797	92314	92314
	AHC	2797	2797	6599	439	9797	405	92314	1115

8.4.3.1 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using DT classifier.*

In this experiment, in order to evaluate the performance of DBBI-SMOTE and compare it with the other balancing techniques comprehensively, we have balanced four datasets using DBBI-SMOTE and the other 34 balancing methods. Then, the generated balanced datasets are used to train DT classifier. In addition, the main differences between the datasets are the data type and the complexity level; the complexity level of the datasets increases gradually, respectively, the Spanish, the Taiwanese, the Polish and the US companies datasets. However, regarding the Spanish companies dataset, as shown in Table 8.3, DBBI-SMOTE leads the DT classifier to obtain the highest value of the *accuracy*, *recall*, *specificity*, *precision* and *AUC*, and the lowest values of *Type I* and *error* compared to the other balancing methods. In other words, the combination of DT and DBBI-SMOTE outperforms the other combinations between DT and the other 34 balancing methods regarding predicting the minority instances and the majority instances. This means that DBBI-SMOTE represents the ideal method to balance the Spanish companies dataset before using it to train the DT classifier. In addition, with respect to the Taiwanese companies dataset, as addressed in Table 8.4, DBBI-SMOTE leads DT to obtain the highest *accuracy*, *recall* and *AUC*, whereas the combination of DT with SMOTE-D obtained the highest *precision* and *specificity*. In other words, DT classifier with conjunction with DBBI-SMOTE is the superior approach to predict the minority instances compared to the other approaches. The combination DT and SMOTE-D is the best approach to determine the majority instances in the Taiwanese companies dataset. Moreover, as shown in Table 8.5, DBBI-SMOTE in conjunction with DT is the optimum approach to predict the minority instances in the Polish companies' dataset, it obtains the highest *accuracy*, *recall* and *AUC*, and the lowest *Type II error*. The combination of DT with CE-SMOTE represents the best combination in predicting the majority class in the Polish companies dataset. This approach obtained the lowest *Type I error* and the best *specificity*.

With respect to the US companies' dataset, as stated in Table 8.6, DBBI-SMOTE leads DT classifier to the lowest minority instances misprediction according to the *Type II error* metric value obtained by this approach. In addition, the combination of DBBI-SMOTE with DT yielded the highest *accuracy*, *recall*, *precision* and *AUC* metrics values compared to the other 34 balancing methods. On the other hand, CE-SMOTE yields the best results with respect to the majority class mispredicting; it obtains the lowest *Type I error*.

Accordingly, in summary, DBBI-SMOTE in conjunction with DT classifier obtains the best results regarding predicting the minority class instances (bankrupt companies) in all of the considered datasets to evaluate the feasibility of the new proposed method (i.e., DBBI-SMOTE).

Table 8.3: The performance metrics values obtained by DT classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9896**	**0.9921**	**0.9871**	**0.9872**	**0.0129**	**0.0079**	**0.9896**
SMOTE	0.9830	0.9886	0.9775	0.9778	0.0225	0.0114	0.9830
BL-SMOTE	0.9836	0.9886	0.9786	0.9788	0.0214	0.0114	0.9836
SMOTEENN	0.9854	0.9913	0.9791	0.9807	0.0209	0.0087	0.9852
SMOTE-Tomek	0.9819	0.9884	0.9754	0.9759	0.0246	0.0116	0.9819
SVM-SMOTE	0.9773	0.9789	0.9764	0.9592	0.0236	0.0211	0.9777
ADASYN	0.9815	0.9913	0.9711	0.9734	0.0289	0.0087	0.9812
MSMOTE	0.9814	0.9864	0.9764	0.9767	0.0236	0.0136	0.9814
ISMOTE	0.9748	0.9832	0.9664	0.9671	0.0336	0.0168	0.9748
SL-SMOTE	0.9658	0.7602	0.9793	0.7081	0.0207	0.2398	0.8698
CE-SMOTE	0.9855	0.9875	0.9836	0.9837	0.0164	0.0125	0.9856
CURE-SMOTE	0.9861	0.9861	0.9861	0.9861	0.0139	0.0139	0.9861
SMOTE-D	0.9859	0.9868	0.9850	0.9851	0.0150	0.0132	0.9859
SDSMOTE	0.9837	0.9886	0.9789	0.9792	0.0211	0.0114	0.9838
G-SMOTE	0.9843	0.9903	0.9782	0.9785	0.0218	0.0097	0.9842
SMOTE-Cosine	0.9787	0.9853	0.9721	0.9726	0.0279	0.0147	0.9787
SMOTE-OUT	0.9793	0.9843	0.9743	0.9746	0.0257	0.0157	0.9793
Selected-SMOTE	0.9812	0.9882	0.9743	0.9747	0.0257	0.0118	0.9812
Gaussian-SMOTE	0.9852	0.9864	0.9839	0.9840	0.0161	0.0136	0.9851
SN-SMOTE	0.9821	0.9875	0.9768	0.9770	0.0232	0.0125	0.9822
LLE-SMOTE	0.9836	0.9846	0.9825	0.9826	0.0175	0.0154	0.9836
Cluster-SMOTE	0.9868	0.9918	0.9818	0.9820	0.0182	0.0082	0.9868
SMOTE-IPF	0.9825	0.9889	0.9760	0.9764	0.0240	0.0111	0.9825
AND-SMOTE	0.9868	0.9918	0.9818	0.9820	0.0182	0.0082	0.9868
LN-SMOTE	0.9839	0.9903	0.9775	0.9778	0.0225	0.0097	0.9839
SMOTE-PSO	0.9727	0.8458	0.9839	0.8246	0.0161	0.1542	0.9148
NT-SMOTE	0.9807	0.9857	0.9757	0.9761	0.0243	0.0143	0.9807
SMOTE-RS$B*$	0.9700	0.4333	0.9832	0.4340	0.0168	0.5667	0.7083
Distance-SMOTE	0.9855	0.9864	0.9846	0.9847	0.0154	0.0136	0.9855
MWMOTE	0.9809	0.9871	0.9746	0.9750	0.0254	0.0129	0.9808
LVQ-SMOTE	0.9873	0.9903	0.9843	0.9844	0.0157	0.0097	0.9873
ADOMS	0.9855	0.9882	0.9828	0.9830	0.0172	0.0118	0.9855
ANS	0.9843	0.9889	0.9796	0.9799	0.0204	0.0111	0.9843
AHC	0.9709	0.6744	0.9839	0.6519	0.0161	0.3256	0.8292
SPY	0.9717	0.4310	0.9846	0.4577	0.0154	0.5690	0.7078

Table 8.4: The performance metrics values obtained by DT classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9771**	**0.9847**	0.9695	0.9700	0.0305	**0.0153**	**0.9771**
SMOTE	0.9524	0.9645	0.9403	0.9417	0.0597	0.0355	0.9524
BL-SMOTE	0.9661	0.9777	0.9544	0.9555	0.0456	0.0223	0.9661
SMOTEENN	0.9637	0.9776	0.9477	0.9557	0.0523	0.0224	0.9627
SMOTE-Tomek	0.9556	0.9726	0.9386	0.9407	0.0614	0.0274	0.9556
SVM-SMOTE	0.9491	0.9207	0.9591	0.8872	0.0409	0.0793	0.9399
ADASYN	0.9556	0.9677	0.9436	0.9444	0.0564	0.0323	0.9556
MSMOTE	0.9640	0.9753	0.9527	0.9538	0.0473	0.0247	0.9640
ISMOTE	0.9318	0.9478	0.9158	0.9186	0.0842	0.0522	0.9318
SL-SMOTE	0.9084	0.7990	0.9327	0.7259	0.0673	0.2010	0.8659
CE-SMOTE	0.9657	0.9773	0.9541	0.9552	0.0459	0.0227	0.9657
CURE-SMOTE	0.9765	0.9798	0.9732	0.9734	0.0268	0.0202	0.9765
SMOTE-D	0.9764	0.9773	**0.9755**	**0.9755**	**0.0245**	0.0227	0.9764
SDSMOTE	0.9548	0.9680	0.9415	0.9430	0.0585	0.0320	0.9547
G-SMOTE	0.9682	0.9832	0.9532	0.9546	0.0468	0.0168	0.9682
SMOTE-Cosine	0.9582	0.9673	0.9491	0.9500	0.0509	0.0327	0.9582
SMOTE-OUT	0.9565	0.9611	0.9520	0.9524	0.0480	0.0389	0.9566
Selected-SMOTE	0.9636	0.9756	0.9515	0.9527	0.0485	0.0244	0.9636
Gaussian-SMOTE	0.9728	0.9747	0.9709	0.9711	0.0291	0.0253	0.9728
SN-SMOTE	0.9577	0.9709	0.9445	0.9460	0.0555	0.0291	0.9577
LLE-SMOTE	0.9716	0.9765	0.9667	0.9670	0.0333	0.0235	0.9716
Cluster-SMOTE	0.9628	0.9783	0.9473	0.9489	0.0527	0.0217	0.9628
SMOTE-IPF	0.9552	0.9694	0.9411	0.9427	0.0589	0.0306	0.9553
AND-SMOTE	0.9717	0.9815	0.9620	0.9627	0.0380	0.0185	0.9718
LN-SMOTE	0.9532	0.3000	0.9750	0.2918	0.0250	0.7000	0.6375
SMOTE-PSO	0.9497	0.8068	0.9688	0.7750	0.0312	0.1932	0.8878
NT-SMOTE	0.9612	0.9688	0.9536	0.9544	0.0464	0.0312	0.9612
SMOTE-RS*B*	0.9505	0.3441	0.9711	0.2895	0.0289	0.6559	0.6576
Distance-SMOTE	0.9711	0.9765	0.9656	0.9660	0.0344	0.0235	0.9710
MWMOTE	0.9510	0.9601	0.9418	0.9429	0.0582	0.0399	0.9509
LVQ-SMOTE	0.9743	0.9792	0.9694	0.9697	0.0306	0.0208	0.9743
ADOMS	0.9661	0.9758	0.9564	0.9572	0.0436	0.0242	0.9661
ANS	0.9503	0.9553	0.9453	0.9460	0.0547	0.0447	0.9503
AHC	0.9471	0.6079	0.9697	0.5765	0.0303	0.3921	0.7888
SPY	0.9478	0.2779	0.9704	0.2448	0.0296	0.7221	0.6241

Table 8.5: The performance metrics values obtained by DT classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9848**	**0.9938**	0.9758	0.9762	0.0242	**0.0062**	**0.9848**
SMOTE	0.9617	0.9733	0.9501	0.9512	0.0499	0.0267	0.9617
BL-SMOTE	0.9753	0.9869	0.9637	0.9645	0.0363	0.0131	0.9753
SMOTE-ENN	0.9682	0.9820	0.9524	0.9593	0.0476	0.0180	0.9672
SMOTE-Tomek	0.9639	0.9744	0.9533	0.9543	0.0467	0.0256	0.9639
SVM-SMOTE	0.9661	0.9641	0.9671	0.9338	0.0329	0.0359	0.9656
ADASYN	0.9630	0.9780	0.9478	0.9497	0.0522	0.0220	0.9629
MSMOTE	0.9776	0.9863	0.9689	0.9694	0.0311	0.0137	0.9776
ISMOTE	0.9461	0.9658	0.9264	0.9294	0.0736	0.0342	0.9461
SL-SMOTE	0.9535	0.8837	0.9637	0.7792	0.0363	0.1163	0.9237
CE-SMOTE	0.9687	0.2650	**0.9833**	0.2518	**0.0167**	0.7350	0.6241
CURE-SMOTE	0.9755	0.9808	0.9701	0.9705	0.0299	0.0192	0.9754
SMOTE-D	0.9832	0.9846	0.9817	**0.9818**	0.0183	0.0154	0.9831
SDSMOTE	0.9620	0.9745	0.9495	0.9507	0.0505	0.0255	0.9620
G-SMOTE	0.9770	0.9906	0.9634	0.9644	0.0366	0.0094	0.9770
SMOTE-Cosine	0.9595	0.9714	0.9475	0.9488	0.0525	0.0286	0.9595
SMOTE-OUT	0.9514	0.9630	0.9397	0.9412	0.0603	0.0370	0.9513
Selected-SMOTE	0.9627	0.9758	0.9497	0.9510	0.0503	0.0242	0.9627
Gaussian-SMOTE	0.9781	0.9793	0.9768	0.9769	0.0232	0.0207	0.9780
SN-SMOTE	0.9675	0.9799	0.9551	0.9562	0.0449	0.0201	0.9675
LLE-SMOTE	0.9817	0.9850	0.9785	0.9786	0.0215	0.0150	0.9818
Cluster-SMOTE	0.9742	0.9867	0.9617	0.9627	0.0383	0.0133	0.9742
SMOTE-IPF	0.9598	0.9737	0.9460	0.9476	0.0540	0.0263	0.9598
AND-SMOTE	0.9825	0.9886	0.9764	0.9767	0.0236	0.0114	0.9825
LN-SMOTE	0.9790	0.9874	0.9705	0.9710	0.0295	0.0126	0.9789
SMOTE-PSO	0.9607	0.7648	0.9769	0.7376	0.0231	0.2352	0.8709
NT-SMOTE	0.9676	0.9767	0.9586	0.9593	0.0414	0.0233	0.9677
SMOTE-RS$B*$	0.9660	0.2420	0.9816	0.2393	0.0184	0.7580	0.6118
Distance-SMOTE	0.9747	0.9858	0.9637	0.9645	0.0363	0.0142	0.9748
MWMOTE	0.9453	0.9515	0.9391	0.9399	0.0609	0.0485	0.9453
LVQ-SMOTE	0.9807	0.9840	0.9773	0.9775	0.0227	0.0160	0.9807
ADOMS	0.9647	0.9783	0.9512	0.9525	0.0488	0.0217	0.9647
ANS	0.9433	0.9509	0.9357	0.9367	0.0643	0.0491	0.9433
AHC	0.9577	0.4986	0.9766	0.4730	0.0234	0.5014	0.7376
SPY	0.9646	0.2552	0.9793	0.2013	0.0207	0.7448	0.6172

Table 8.6: The performance metrics values obtained by DT classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9956**	**0.9969**	0.9943	**0.9944**	0.0057	**0.0031**	**0.9956**
SMOTE	0.9830	0.9902	0.9757	0.9761	0.0243	0.0098	0.9829
BL-SMOTE	0.9935	0.9955	0.9915	0.9916	0.0085	0.0045	0.9935
SMOTE-ENN	0.9860	0.9922	0.9796	0.9805	0.0204	0.0078	0.9859
SMOTE-Tomek	0.9827	0.9906	0.9747	0.9751	0.0253	0.0094	0.9827
SVM-SMOTE	0.9921	0.9952	0.9890	0.9890	0.0110	0.0048	0.9921
ADASYN	0.9827	0.9903	0.9751	0.9754	0.0249	0.0097	0.9827
MSMOTE	0.9937	0.9954	0.9920	0.9920	0.0080	0.0046	0.9937
ISMOTE	0.9791	0.9890	0.9693	0.9699	0.0307	0.0110	0.9791
SL-SMOTE	0.9885	0.7685	0.9935	0.7265	0.0065	0.2315	0.8810
CE-SMOTE	0.9897	0.1793	**0.9946**	0.1613	**0.0054**	0.8207	0.5869
CURE-SMOTE	0.9915	0.9946	0.9885	0.9885	0.0115	0.0054	0.9916
SMOTE-D	0.9937	0.9939	0.9936	0.9936	0.0064	0.0061	0.9938
SDSMOTE	0.9826	0.9901	0.9750	0.9754	0.0250	0.0099	0.9826
G-SMOTE	0.9911	0.9962	0.9860	0.9861	0.0140	0.0038	0.9911
SMOTE-Cosine	0.9810	0.9871	0.9748	0.9751	0.0252	0.0129	0.9809
SMOTE-OUT	0.9796	0.9862	0.9730	0.9734	0.0270	0.0138	0.9796
Selected-SMOTE	0.9827	0.9902	0.9752	0.9755	0.0248	0.0098	0.9827
Gaussian-SMOTE	0.9937	0.9938	0.9935	0.9935	0.0065	0.0062	0.9937
SN-SMOTE	0.9859	0.9928	0.9790	0.9793	0.0210	0.0072	0.9859
LLE-SMOTE	0.9939	0.9950	0.9928	0.9929	0.0072	0.0050	0.9939
Cluster-SMOTE	0.9892	0.9952	0.9831	0.9833	0.0169	0.0048	0.9891
SMOTE-IPF	0.9828	0.9903	0.9753	0.9757	0.0247	0.0097	0.9828
AND-SMOTE	0.9947	0.9960	0.9934	0.9934	0.0066	0.0040	0.9947
LN-SMOTE	0.9891	0.1754	0.9940	0.1486	0.0060	0.8246	0.5847
SMOTE-PSO	0.9862	0.7176	0.9917	0.6384	0.0083	0.2824	0.8547
NT-SMOTE	0.9854	0.9900	0.9807	0.9809	0.0193	0.0100	0.9853
Distance-SMOTE	0.9912	0.9961	0.9863	0.9864	0.0137	0.0039	0.9912
MWMOTE	0.9689	0.9701	0.9677	0.9678	0.0323	0.0299	0.9689
LVQ-SMOTE	0.9944	0.9950	0.9938	0.9939	0.0062	0.0050	0.9944
ADOMS	0.9863	0.9937	0.9788	0.9791	0.0212	0.0063	0.9863
ANS	0.9667	0.9681	0.9652	0.9653	0.0348	0.0319	0.9667
AHC	0.9855	0.4001	0.9926	0.3990	0.0074	0.5999	0.6964
SPY	0.9889	0.1810	0.9941	0.1639	0.0059	0.8190	0.5876

8.4.3.2 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using NB classifier.*

In this experiment, DBBI-SMOTE and the other 34 resampling methods have been used as a prepossessing stage to prepare the Spanish, Taiwanese, Polish and US companies' datasets before utilizing it to train the NB classifier. Thus, with respect to the Spanish companies dataset, as stated in Table 8.7, the best evaluation metrics values are not tied with a specific balancing method combined with NB classifier. However, the conjunction of CE-SMOTE with NB yielded the highest *accuracy* and *AUC*. The combination of MWMOTE and NB obtained the best results regarding predicting the minority instances (bankrupt companies) according to the *recall* and *Type II error* metrics values, whereas SMOTE-D shows the best performance in predicting the majority instances (solvent companies) as stated in *specificity* and *type I error*. Besides, despite that the DBBI-SMOTE doesn't obtain the best results in each evaluation metric, it outperforms 13 other balancing methods in predicting the minority class instances (bankrupt companies); DBBI-SMOTE is not the worst resampling method to combine with NB classifier.

Moreover, also in the Taiwanese companies' dataset, as stated in Table 8.8, the combination of SMOTE-D with NB classifier shows the highest performance in determining the majority class instances (solvent companies) according to the *specificity* and *type I error*. Also, the same combination obtained the best *accuracy*, *precision* and *AUC* evaluation metrics values. On the other hand, AND-SMOTE yields the best results in determining the minority class instances (bankrupt companies) as stated in the *recall* and *type II error* values. Besides, also in the Taiwanese dataset, DBBI-SMOTE is not the worst resampling approach to combine with NB classifier to process the Taiwanese companies' dataset; it outperforms 29 other balancing methods in predicting the minority class instances (bankrupt companies).

With respect to the Polish companies dataset, as shown in Table 8.9, the combination of NB classifier with SMOTE-D yields the best performance in predicting the majority class instances (solvent companies). It obtains the best *accuracy*, *precision*, *AUC*, *specificity* and *type I error*, whereas LVQ-SMOTE shows the highest performance in predicting the minority class instances (bankrupt companies) according to the *recall* and *type I error*. On the other hand, DBBI-SMOTE outperforms 25 balancing method in recognizing the minority class instances (bankrupt companies).

Finally, as stated in Table 8.10, the conjunction of NB with ISMOTE shows the optimum performance in predicting the majority class instances (solvent companies) according to the *specificity* and *type I error* evaluation metrics values in the US companies' dataset. In addition, it obtains the best *precision* value. On the other hand, LVQ-SMOTE yields the best performance in predicting the minority class instances (bankrupt companies) as stated in the *recall* and *type II error* values. The combination of Gaussian-SMOTE with NB obtained the highest *accuracy* and *AUC* values. Besides, DBBI-SMOTE outperformed other 28 balancing methods in predicting the minority class instances (bankrupt companies).

Accordingly, as a firm conclusion, the NB classifier doesn't present a robust classification algorithm to process the financial datasets, its performance is the worst compared to the other classifiers. In addition, DBBI-SMOTE is not the superior method in predicting the minority or the majority classes instances compared to the other balancing methods, but it is not the worst, it outperforms a considerable number of balancing methods in predicting the minority instances (bankrupt companies).

Table 8.7: The performance metrics values obtained by NB classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.5849	0.9946	0.1752	0.5467	0.8248	0.0054	0.5849
SMOTE	0.5935	0.9989	0.1881	0.5517	0.8119	0.0011	0.5935
BL-SMOTE	0.5978	0.9989	0.1966	0.5544	0.8034	0.0011	0.5978
SMOTEENN	0.6110	0.9977	0.1988	0.5705	0.8012	0.0023	0.5983
SMOTE-Tomek	0.5950	0.9993	0.1907	0.5526	0.8093	0.0007	0.5950
SVM-SMOTE	0.4802	0.9981	0.1902	0.4084	0.8098	0.0019	0.5941
ADASYN	0.6104	0.9977	0.1976	0.5700	0.8024	0.0023	0.5977
MSMOTE	0.5964	0.9946	0.1981	0.5538	0.8019	0.0054	0.5964
ISMOTE	0.8181	0.6517	0.9846	0.9773	0.0154	0.3483	0.8181
SL-SMOTE	0.1865	0.9892	0.1337	0.0699	0.8663	0.0108	0.5615
CE-SMOTE	**0.9873**	0.9807	0.9939	0.9939	0.0061	0.0193	**0.9873**
CURE-SMOTE	0.6815	0.9871	0.3758	0.6129	0.6242	0.0129	0.6815
SMOTE-D	0.9870	0.9789	**0.9950**	**0.9949**	**0.0050**	0.0211	0.9869
SDSMOTE	0.5937	0.9982	0.1891	0.5518	0.8109	0.0018	0.5937
G-SMOTE	0.6001	0.9989	0.2013	0.5557	0.7987	0.0011	0.6001
SMOTE-Cosine	0.5887	0.9921	0.1852	0.5491	0.8148	0.0079	0.5887
SMOTE-OUT	0.5872	0.9893	0.1852	0.5485	0.8148	0.0107	0.5872
Selected-SMOTE	0.5947	0.9982	0.1913	0.5525	0.8087	0.0018	0.5948
Gaussian-SMOTE	0.9322	0.9499	0.9145	0.9177	0.0855	0.0501	0.9322
SN-SMOTE	0.5926	0.9989	0.1863	0.5512	0.8137	0.0011	0.5926
LLE-SMOTE	0.6816	0.4401	0.9231	0.8539	0.0769	0.5599	0.6816
Cluster-SMOTE	0.6073	0.9907	0.2238	0.5608	0.7762	0.0093	0.6073
SMOTE-IPF	0.5922	0.9968	0.1877	0.5511	0.8123	0.0032	0.5923
AND-SMOTE	0.6085	0.9993	0.2178	0.5611	0.7822	0.0007	0.6085
LN-SMOTE	0.5787	0.9875	0.1699	0.5435	0.8301	0.0125	0.5787
SMOTE-PSO	0.2240	0.9960	0.1555	0.0947	0.8445	0.0040	0.5757
NT-SMOTE	0.5951	0.9971	0.1930	0.5528	0.8070	0.0029	0.5950
SMOTE-RS*B*∗	0.1996	0.9429	0.1813	0.0276	0.8187	0.0571	0.5621
Distance-SMOTE	0.6099	0.9986	0.2213	0.5619	0.7787	0.0014	0.6099
MWMOTE	0.6058	**1.0**	0.2117	0.5593	0.7883	**0.0**	0.6058
LVQ-SMOTE	0.6598	0.9971	0.3225	0.5956	0.6775	0.0029	0.6598
ADOMS	0.5842	0.9975	0.1709	0.5462	0.8291	0.0025	0.5842
ANS	0.6037	0.9996	0.2077	0.5580	0.7923	0.0004	0.6037
AHC	0.2322	0.9750	0.1995	0.0509	0.8005	0.0250	0.5873
SPY	0.2001	0.9548	0.1819	0.0273	0.8181	0.0452	0.5684

Table 8.8: The performance metrics values obtained by NB classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray..

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.6649	0.9865	0.3434	0.6010	0.6566	0.0135	0.6650
SMOTE	0.6972	0.9620	0.4325	0.6317	0.5675	0.0380	0.6972
BL-SMOTE	0.6718	0.9751	0.3684	0.6077	0.6316	0.0249	0.6717
SMOTEENN	0.8510	0.9222	0.7690	0.8229	0.2310	0.0778	0.8456
SMOTE-Tomek	0.7100	0.9634	0.4565	0.6406	0.5435	0.0366	0.7100
SVM-SMOTE	0.5538	0.9753	0.4064	0.3667	0.5936	0.0247	0.6908
ADASYN	0.7052	0.9592	0.4542	0.6380	0.5458	0.0408	0.7067
MSMOTE	0.5916	0.9877	0.1955	0.5514	0.8045	0.0123	0.5916
ISMOTE	0.7532	0.5434	0.9630	0.9351	0.0370	0.4566	0.7532
SL-SMOTE	0.2712	0.9796	0.1136	0.1975	0.8864	0.0204	0.5466
CE-SMOTE	0.6615	0.9882	0.3347	0.5986	0.6653	0.0118	0.6614
CURE-SMOTE	0.7393	0.9694	0.5092	0.6658	0.4908	0.0306	0.7393
SMOTE-D	**0.9819**	0.9684	**0.9955**	**0.9953**	**0.0045**	0.0316	**0.9820**
SDSMOTE	0.7027	0.9617	0.4437	0.6352	0.5563	0.0383	0.7027
G-SMOTE	0.7075	0.9644	0.4507	0.6406	0.5493	0.0356	0.7076
SMOTE-Cosine	0.6833	0.9597	0.4069	0.6194	0.5931	0.0403	0.6833
SMOTE-OUT	0.6801	0.9604	0.3997	0.6159	0.6003	0.0396	0.6801
Selected-SMOTE	0.7200	0.9601	0.4799	0.6508	0.5201	0.0399	0.7200
Gaussian-SMOTE	0.9736	0.9744	0.9727	0.9728	0.0273	0.0256	0.9736
SN-SMOTE	0.6993	0.9629	0.4357	0.6323	0.5643	0.0371	0.6993
LLE-SMOTE	0.8771	0.9933	0.7609	0.8064	0.2391	0.0067	0.8771
Cluster-SMOTE	0.7078	0.9633	0.4523	0.6388	0.5477	0.0367	0.7078
SMOTE-IPF	0.7004	0.9614	0.4395	0.6340	0.5605	0.0386	0.7005
AND-SMOTE	0.6876	**0.9961**	0.3791	0.6166	0.6209	**0.0039**	0.6876
LN-SMOTE	0.5579	0.8682	0.5475	0.0780	0.4525	0.1318	0.7078
SMOTE-PSO	0.9045	0.6193	0.9426	0.5923	0.0574	0.3807	0.7810
NT-SMOTE	0.6980	0.9755	0.4205	0.6287	0.5795	0.0245	0.6980
SMOTE-RS*B*∗	0.5943	0.8883	0.5843	0.0812	0.4157	0.1117	0.7363
Distance-SMOTE	0.7470	0.9720	0.5220	0.6715	0.4780	0.0280	0.7470
MWMOTE	0.7218	0.9701	0.4736	0.6527	0.5264	0.0299	0.7218
LVQ-SMOTE	0.6668	0.9971	0.3364	0.6008	0.6636	0.0029	0.6667
ADOMS	0.7062	0.9636	0.4488	0.6405	0.5512	0.0364	0.7062
ANS	0.6971	0.9733	0.4208	0.6315	0.5792	0.0267	0.6971
AHC	0.5493	0.9430	0.5231	0.1180	0.4769	0.0570	0.7330
SPY	0.5950	0.8743	0.5856	0.0807	0.4144	0.1257	0.7299

Table 8.9: The performance metrics values obtained by NB classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.5624	0.9638	0.1611	0.5346	0.8389	0.0362	0.5625
SMOTE	0.5485	0.9553	0.1418	0.5268	0.8582	0.0447	0.5485
Borderline	0.5754	0.9707	0.1802	0.5421	0.8198	0.0293	0.5755
SMOTE-ENN	0.5761	0.9581	0.1406	0.5597	0.8594	0.0419	0.5494
SMOTE-Tomek	0.5491	0.9564	0.1418	0.5271	0.8582	0.0436	0.5491
SVM-SMOTE	0.4321	0.9694	0.1739	0.3606	0.8261	0.0306	0.5716
ADASYN	0.5488	0.9533	0.1416	0.5278	0.8584	0.0467	0.5474
kmeans	0.8707	0.9842	0.7572	0.8024	0.2428	0.0158	0.8707
MSMOTE	0.5613	0.9746	0.1481	0.5336	0.8519	0.0254	0.5614
ISMOTE	0.6658	0.4160	0.9156	0.8325	0.0844	0.5840	0.6658
SL-SMOTE	0.2437	0.9774	0.1375	0.1409	0.8625	0.0226	0.5575
CE-SMOTE	0.1245	0.9210	0.1080	0.0209	0.8920	0.0790	0.5145
CURE-SMOTE	0.6314	0.9672	0.2955	0.5786	0.7045	0.0328	0.6313
SMOTE-D	**0.9890**	0.9792	**0.9989**	**0.9989**	**0.0011**	0.0208	**0.9890**
SDSMOTE	0.5471	0.9536	0.1407	0.5260	0.8593	0.0464	0.5472
G-SMOTE	0.5544	0.9588	0.1499	0.5301	0.8501	0.0412	0.5544
SMOTE-Cosine	0.5604	0.9573	0.1634	0.5337	0.8366	0.0427	0.5604
SMOTE-OUT	0.5583	0.9592	0.1575	0.5324	0.8425	0.0408	0.5584
Selected-SMOTE	0.5497	0.9561	0.1433	0.5274	0.8567	0.0439	0.5497
Gaussian-SMOTE	0.8533	0.8771	0.8295	0.8531	0.1705	0.1229	0.8533
SN-SMOTE	0.5413	0.9576	0.1249	0.5225	0.8751	0.0424	0.5413
LLE-SMOTE	0.6305	0.9915	0.2696	0.5759	0.7304	0.0085	0.6306
Cluster-SMOTE	0.5393	0.9537	0.1249	0.5215	0.8751	0.0463	0.5393
SMOTE-IPF	0.5485	0.9553	0.1417	0.5268	0.8583	0.0447	0.5485
AND-SMOTE	0.5771	0.9614	0.1928	0.5436	0.8072	0.0386	0.5771
LN-SMOTE	0.5486	0.9614	0.1359	0.5267	0.8641	0.0386	0.5486
SMOTE-PSO	0.1388	0.9372	0.0727	0.0773	0.9273	0.0628	0.5050
NT-SMOTE	0.5556	0.9585	0.1527	0.5308	0.8473	0.0415	0.5556
SMOTE-RS*B*∗	0.1217	0.9290	0.1043	0.0218	0.8957	0.0710	0.5167
Distance-SMOTE	0.5556	0.9569	0.1542	0.5309	0.8458	0.0431	0.5555
MWMOTE	0.5529	0.9556	0.1501	0.5293	0.8499	0.0444	0.5529
LVQ-SMOTE	0.5406	**0.9927**	0.0885	0.5213	0.9115	**0.0073**	0.5406
ADOMS	0.5397	0.9544	0.1249	0.5217	0.8751	0.0456	0.5396
ANS	0.5882	0.9677	0.2086	0.5502	0.7914	0.0323	0.5881
AHC	0.1615	0.9432	0.1292	0.0429	0.8708	0.0568	0.5362
SPY	0.1241	0.9119	0.1078	0.0207	0.8922	0.0881	0.5099

Table 8.10: The performance metrics values obtained by NB classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.5921	0.9797	0.2045	0.5519	0.7955	0.0203	0.5921
SMOTE	0.5204	0.9732	0.0676	0.5107	0.9324	0.0268	0.5204
BL-SMOTE	0.5955	0.9806	0.2104	0.5540	0.7896	0.0194	0.5955
SMOTE-ENN	0.5308	0.9747	0.0707	0.5208	0.9293	0.0253	0.5227
SMOTE-Tomek	0.5210	0.9736	0.0685	0.5110	0.9315	0.0264	0.5211
SVM-SMOTE	0.5988	0.9854	0.2123	0.5557	0.7877	0.0146	0.5988
ADASYN	0.5184	0.9718	0.0651	0.5097	0.9349	0.0282	0.5184
MSMOTE	0.5821	0.9740	0.1902	0.5460	0.8098	0.0260	0.5821
ISMOTE	0.6218	0.2712	**0.9725**	**0.9078**	**0.0275**	0.7288	0.6219
SL-SMOTE	0.1106	0.9672	0.0914	0.0234	0.9086	0.0328	0.5293
CE-SMOTE	0.0715	0.9624	0.0661	0.0062	0.9339	0.0376	0.5142
CURE-SMOTE	0.5571	0.9933	0.1210	0.5305	0.8790	0.0067	0.5572
SMOTE-D	0.4855	0.9385	0.0325	0.4924	0.9675	0.0615	0.4855
SDSMOTE	0.5210	0.9739	0.0681	0.5110	0.9319	0.0261	0.5210
G-SMOTE	0.5202	0.9740	0.0664	0.5106	0.9336	0.0260	0.5202
SMOTE-Cosine	0.5239	0.9734	0.0745	0.5126	0.9255	0.0266	0.5240
SMOTE-OUT	0.5232	0.9744	0.0720	0.5122	0.9280	0.0256	0.5232
Selected-SMOTE	0.5208	0.9743	0.0673	0.5109	0.9327	0.0257	0.5208
Gaussian-SMOTE	**0.7685**	0.6268	0.9101	0.8718	0.0899	0.3732	**0.7685**
SN-SMOTE	0.5186	0.9735	0.0637	0.5097	0.9363	0.0265	0.5186
LLE-SMOTE	0.5232	0.9582	0.0883	0.5124	0.9117	0.0418	0.5232
Cluster-SMOTE	0.5167	0.9719	0.0615	0.5087	0.9385	0.0281	0.5167
SMOTE-IPF	0.5197	0.9727	0.0667	0.5103	0.9333	0.0273	0.5197
AND-SMOTE	0.6191	0.9931	0.2451	0.5682	0.7549	0.0069	0.6191
LN-SMOTE	0.0719	0.9622	0.0665	0.0062	0.9335	0.0378	0.5144
SMOTE-PSO	0.0850	0.9735	0.0668	0.0209	0.9332	0.0265	0.5202
NT-SMOTE	0.5195	0.9730	0.0660	0.5102	0.9340	0.0270	0.5195
Distance-SMOTE	0.5209	0.9730	0.0688	0.5110	0.9312	0.0270	0.5209
MWMOTE	0.5500	0.9789	0.1211	0.5269	0.8789	0.0211	0.5500
LVQ-SMOTE	0.5287	**0.9998**	0.0575	0.5148	0.9425	**0.0002**	0.5287
ADOMS	0.5192	0.9725	0.0660	0.5101	0.9340	0.0275	0.5192
ANS	0.5422	0.9622	0.1222	0.5229	0.8778	0.0378	0.5422
AHC	0.0806	0.9731	0.0698	0.0125	0.9302	0.0269	0.5214
SPY	0.0727	0.9610	0.0671	0.0066	0.9329	0.0390	0.5141

8.4.3.3 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using RF classifier.*

Also in this experiment, in order to evaluate the performance of the new proposed balancing technique (DBBI-SMOTE), we have used it to prepare four datasets before using them to train one of the very high-performance classifiers (i.e., RF). In addition, as in the previous experiments, the performance of DBBI-SMOTE will be compared to the other 34 balancing methods according to several evaluation metrics values. Thus, regarding the Spanish companies' dataset, as shown in Table 8.11, the conjunction of SMOTE-ENN and ADASYN with RF is the superior approach in determining the minority instances (bankrupt companies), both of them obtains the best *recall* and *type II error*. SMOTE-ENN obtains the best *accuracy* and *AUC* as well. On the other hand, the combination of RF with SMOTE-D yields the optimum performance regarding the majority class mispredicting (solvent companies) as stated in the *specificity* and *type I error* metrics values. DBBI-SMOTE obtains very close values to the superior combinations.

With respect to the Taiwanese companies' dataset, as shown in Table 8.12, as with the Spanish companies' dataset, the combination of SMOTE-ENN with RF yields the highest performance in the minority class instances predicting and mispredicting as stated in the *recall* and *type II error*. The same combination obtains the highest *accuracy* and *AUC* as well. The combinations of CURE-SMOTE and SMOTE-RS$B*$ with RF shows the optimum performance in the majority class instances predicting and mispredicting. These combinations obtained the highest *specificity* and the lowest *type I error*, whereas CURE-SMOTE obtained the best *precision*. Also in this dataset, the results of the DBBI-SMOTE is very close to the superior combinations in determining the minority and majority classes instances.

Moreover, referring to the Polish companies dataset, Table 8.13 shows the results of the conjunction of DBBI-SMOTE with RF, and the results of the remaining 34 approaches. Thus, as it can be seen in the table, the approach of the DBBI-SMOTE obtained the best *accuracy* and *AUC* metrics values. The combination of SMOTE-ENN with RF yields the best performance in predicting the minority class instances (bankrupt companies) as stated in the *recall* and *type II error*. Nevertheless, the combinations CE-SMOTE and SMOTE-D with RF return the best performance in determining the majority class instances (solvent companies).

Besides, concerning the US companies dataset, DBBI-SMOTE outcomes the highest *accuracy* and *AUC* metrics values, whereas the combination of RF and ADOMS shows the highest performance in determining the minority class instances (bankrupt companies). In addition, CE-SMOTE returns the best performance in predicting the majority instances.

However, DBBI-SMOTE shows very high performance in preparing the four extremely imbalanced datasets as a preprocessing stage of RF classifier training.

Table 8.11: The performance metrics values obtained by RF classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9954	0.9921	0.9986	0.9986	0.0014	0.0079	0.9953
SMOTE	0.9968	0.9979	0.9957	0.9957	0.0043	0.0021	0.9968
BL-SMOTE	0.9964	0.9961	0.9968	0.9968	0.0032	0.0039	0.9965
SVM-SMOTE	0.9947	0.9930	0.9957	0.9924	0.0043	0.0070	0.9944
ADASYN	0.9971	**0.9989**	0.9952	0.9955	0.0048	**0.0011**	0.9970
SMOTE-NC	0.9955	0.9982	0.9928	0.9929	0.0072	0.0018	0.9955
SMOTE-Tomek	0.9967	0.9986	0.9949	0.9950	0.0051	0.0014	0.9968
SMOTE-ENN	**0.9979**	**0.9989**	0.9968	0.9970	0.0032	**0.0011**	**0.9979**
MSMOTE	0.9918	0.9882	0.9954	0.9953	0.0046	0.0118	0.9918
ISMOTE	0.9937	0.9958	0.9916	0.9917	0.0084	0.0042	0.9937
SL-SMOTE	0.9822	0.7553	0.9971	0.9456	0.0029	0.2447	0.8762
CE-SMOTE	0.9918	0.9843	0.9993	0.9993	0.0007	0.0157	0.9918
CURE-SMOTE	0.9918	0.9850	0.9986	0.9986	0.0014	0.0150	0.9918
SMOTE-D	0.9914	0.9832	**0.9996**	**0.9996**	**0.0004**	0.0168	0.9914
SDSMOTE	0.9968	0.9986	0.9950	0.9950	0.0050	0.0014	0.9968
G-SMOTE	0.9952	0.9939	0.9964	0.9964	0.0036	0.0061	0.9951
SMOTE-Cosine	0.9952	0.9954	0.9950	0.9950	0.0050	0.0046	0.9952
SMOTE-OUT	0.9966	0.9982	0.9950	0.9950	0.0050	0.0018	0.9966
Selected-SMOTE	0.9971	0.9982	0.9961	0.9961	0.0039	0.0018	0.9971
Gaussian-SMOTE	0.9914	0.9843	0.9986	0.9986	0.0014	0.0157	0.9914
SN-SMOTE	0.9962	0.9957	0.9968	0.9968	0.0032	0.0043	0.9963
LLE-SMOTE	0.9921	0.9846	0.9996	0.9996	0.0004	0.0154	0.9921
Cluster-SMOTE	0.9966	0.9975	0.9957	0.9957	0.0043	0.0025	0.9966
SMOTE-IPF	0.9970	0.9982	0.9957	0.9957	0.0043	0.0018	0.9970
AND-SMOTE	0.9941	0.9914	0.9968	0.9968	0.0032	0.0086	0.9941
LN-SMOTE	0.9945	0.9946	0.9943	0.9943	0.0057	0.0054	0.9945
SMOTE-PSO	0.9872	0.8673	0.9979	0.9739	0.0021	0.1327	0.9326
NT-SMOTE	0.9943	0.9907	0.9979	0.9979	0.0021	0.0093	0.9943
SMOTE-RS*B**	0.9836	0.3429	0.9996	0.9000	0.0004	0.6571	0.6713
Distance-SMOTE	0.9929	0.9868	0.9989	0.9989	0.0011	0.0132	0.9929
MWMOTE	0.9955	0.9964	0.9946	0.9947	0.0054	0.0036	0.9955
LVQ-SMOTE	0.9918	0.9886	0.9950	0.9950	0.0050	0.0114	0.9918
ADOMS	0.9941	0.9911	0.9971	0.9971	0.0029	0.0089	0.9941
ANS	0.9945	0.9964	0.9925	0.9926	0.0075	0.0036	0.9945
AHC	0.9829	0.6179	0.9989	0.9625	0.0011	0.3821	0.8084
SPY	0.9822	0.2690	0.9993	0.9167	0.0007	0.7310	0.6341

Table 8.12: The performance metrics values obtained by RF classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9895	0.9844	0.9945	0.9945	0.0055	0.0156	0.9895
SMOTE	0.9796	0.9799	0.9642	0.9653	0.0358	0.0050	0.9796
BL-SMOTE	0.9841	0.9918	0.9764	0.9768	0.0236	0.0082	0.9841
SVM-SMOTE	0.9745	0.9584	0.9801	0.9443	0.0199	0.0416	0.9692
ADASYN	0.9796	0.9966	0.9627	0.9636	0.0373	0.0034	0.9797
SMOTE-Tomek	0.9807	0.9968	0.9647	0.9658	0.0353	0.0032	0.9808
SMOTE-ENN	**0.9906**	**0.9970**	0.9834	0.9857	0.0166	**0.0030**	**0.9902**
MSMOTE	0.9817	0.9767	0.9868	0.9867	0.0132	0.0233	0.9818
ISMOTE	0.9639	0.9850	0.9428	0.9452	0.0572	0.0150	0.9639
SL-SMOTE	0.9486	0.7875	0.9844	0.9184	0.0156	0.2125	0.8860
CE-SMOTE	0.9839	0.9897	0.9780	0.9783	0.0220	0.0103	0.9839
CURE-SMOTE	0.9846	0.9723	**0.9970**	**0.9969**	**0.0030**	0.0277	0.9847
SMOTE-D	0.9838	0.9715	0.9961	0.9960	0.0039	0.0285	0.9838
SDSMOTE	0.9802	0.9958	0.9647	0.9658	0.0353	0.0042	0.9803
G-SMOTE	0.9855	0.9959	0.9751	0.9757	0.0249	0.0041	0.9855
SMOTE-Cosine	0.9800	0.9947	0.9653	0.9663	0.0347	0.0053	0.9800
SMOTE-OUT	0.9806	0.9950	0.9662	0.9672	0.0338	0.0050	0.9806
Selected-SMOTE	0.9823	0.9985	0.9662	0.9673	0.0338	0.0015	0.9824
Gaussian-SMOTE	0.9845	0.9726	0.9965	0.9964	0.0035	0.0274	0.9846
SN-SMOTE	0.9819	0.9970	0.9668	0.9678	0.0332	0.0030	0.9819
LLE-SMOTE	0.9845	0.9723	0.9967	0.9966	0.0033	0.0277	0.9845
Cluster-SMOTE	0.9840	0.9959	0.9721	0.9728	0.0279	0.0041	0.9840
SMOTE-IPF	0.9800	0.9961	0.9639	0.9651	0.0361	0.0039	0.9800
AND-SMOTE	0.9853	0.9838	0.9868	0.9868	0.0132	0.0162	0.9853
LN-SMOTE	0.9695	0.1682	0.9962	0.6337	0.0038	0.8318	0.5822
SMOTE-PSO	0.9675	0.8125	0.9882	0.9039	0.0118	0.1875	0.9003
NT-SMOTE	0.9827	0.9886	0.9768	0.9772	0.0232	0.0114	0.9827
SMOTE-RS*B*∗	0.9700	0.1747	**0.9970**	0.6628	**0.0030**	0.8253	0.5858
Distance-SMOTE	0.9848	0.9820	0.9877	0.9877	0.0123	0.0180	0.9849
MWMOTE	0.9768	0.9908	0.9629	0.9639	0.0371	0.0092	0.9768
LVQ-SMOTE	0.9845	0.9751	0.9939	0.9938	0.0061	0.0249	0.9845
ADOMS	0.9844	0.9950	0.9738	0.9744	0.0262	0.0050	0.9844
ANS	0.9762	0.9903	0.9621	0.9632	0.0379	0.0097	0.9762
AHC	0.9656	0.5560	0.9929	0.8400	0.0071	0.4440	0.7745
SPY	0.9692	0.1802	0.9959	0.5846	0.0041	0.8198	0.5880

Table 8.13: The performance metrics values obtained by RF classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9981**	0.9986	0.9976	0.9976	0.0024	0.0014	**0.9981**
SMOTE	0.9932	0.9984	0.9880	0.9881	0.0120	0.0016	0.9932
BL-SMOTE	0.9935	0.9949	0.9920	0.9921	0.0080	0.0051	0.9934
SVM-SMOTE	0.9919	0.9858	0.9948	0.9891	0.0052	0.0142	0.9903
ADASYN	0.9930	0.9986	0.9874	0.9877	0.0126	0.0014	0.9930
SMOTE-Tomek	0.9933	0.9988	0.9878	0.9879	0.0122	0.0012	0.9933
SMOTE-ENN	0.9926	**0.9992**	0.9852	0.9871	0.0148	**0.0008**	0.9922
MSMOTE	0.9892	0.9862	0.9922	0.9922	0.0078	0.0138	0.9892
ISMOTE	0.9864	0.9932	0.9796	0.9799	0.0204	0.0068	0.9864
SL-SMOTE	0.9746	0.8610	0.9910	0.9332	0.0090	0.1390	0.9260
CE-SMOTE	0.9789	0.0095	**0.9990**	0.1000	**0.0010**	0.9905	0.5042
CURE-SMOTE	0.9920	0.9863	0.9978	0.9977	0.0022	0.0137	0.9920
SMOTE-D	0.9894	0.9798	**0.9990**	**0.9990**	**0.0010**	0.0202	0.9894
SDSMOTE	0.9922	0.9977	0.9868	0.9870	0.0132	0.0023	0.9923
G-SMOTE	0.9966	0.9986	0.9947	0.9947	0.0053	0.0014	0.9967
SMOTE-Cosine	0.9916	0.9970	0.9862	0.9864	0.0138	0.0030	0.9916
SMOTE-OUT	0.9905	0.9962	0.9847	0.9849	0.0153	0.0038	0.9905
Selected-SMOTE	0.9913	0.9968	0.9858	0.9860	0.0142	0.0032	0.9913
Gaussian-SMOTE	0.9890	0.9798	0.9983	0.9982	0.0017	0.0202	0.9890
SN-SMOTE	0.9938	0.9986	0.9891	0.9892	0.0109	0.0014	0.9939
LLE-SMOTE	0.9889	0.9795	0.9983	0.9982	0.0017	0.0205	0.9889
Cluster-SMOTE	0.9952	0.9981	0.9923	0.9924	0.0077	0.0019	0.9952
SMOTE-IPF	0.9918	0.9970	0.9865	0.9867	0.0135	0.0030	0.9918
AND-SMOTE	0.9927	0.9891	0.9963	0.9963	0.0037	0.0109	0.9927
LN-SMOTE	0.9936	0.9913	0.9959	0.9959	0.0041	0.0087	0.9936
SMOTE-PSO	0.9751	0.7057	0.9974	0.9583	0.0026	0.2943	0.8516
NT-SMOTE	0.9922	0.9941	0.9903	0.9903	0.0097	0.0059	0.9922
SMOTE-RS$B*$	0.9789	0.0569	0.9988	0.5167	0.0012	0.9431	0.5279
Distance-SMOTE	0.9926	0.9918	0.9934	0.9934	0.0066	0.0082	0.9926
MWMOTE	0.9811	0.9934	0.9689	0.9696	0.0311	0.0066	0.9811
LVQ-SMOTE	0.9890	0.9796	0.9985	0.9984	0.0015	0.0204	0.9890
ADOMS	0.9934	0.9973	0.9895	0.9896	0.0105	0.0027	0.9934
ANS	0.9790	0.9908	0.9672	0.9680	0.0328	0.0092	0.9790
AHC	0.9695	0.2618	0.9988	0.9039	0.0012	0.7382	0.6303
SPY	0.9790	0.0195	0.9989	0.1667	0.0011	0.9805	0.5092

Table 8.14: The performance metrics values obtained by RF classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9980**	0.9973	0.9987	0.9987	0.0013	0.0027	**0.9980**
SMOTE	0.9933	0.9994	0.9872	0.9873	0.0128	0.0006	0.9933
BL-SMOTE	0.9963	0.9963	0.9964	0.9964	0.0036	0.0037	0.9963
SMOTE-ENN	0.9948	0.9995	0.9898	0.9903	0.0102	0.0005	0.9947
SMOTE-Tomek	0.9933	0.9994	0.9872	0.9874	0.0128	0.0006	0.9933
SVM-SMOTE	0.9964	0.9978	0.9950	0.9950	0.0050	0.0022	0.9964
ADASYN	0.9933	0.9996	0.9871	0.9873	0.0129	0.0004	0.9933
MSMOTE	0.9963	0.9956	0.9970	0.9970	0.0030	0.0044	0.9963
ISMOTE	0.9912	0.9980	0.9845	0.9847	0.0155	0.0020	0.9912
SL-SMOTE	0.9939	0.7391	0.9996	0.9753	0.0004	0.2609	0.8694
CE-SMOTE	0.9940	0.0232	**0.9999**	0.5250	**0.0001**	0.9768	0.5115
CURE-SMOTE	0.9968	0.9958	0.9977	0.9977	0.0023	0.0042	0.9968
SMOTE-D	0.9968	0.9937	0.9998	0.9998	0.0002	0.0063	0.9968
SDSMOTE	0.9933	0.9994	0.9873	0.9875	0.0127	0.0006	0.9933
G-SMOTE	0.9970	0.9994	0.9945	0.9946	0.0055	0.0006	0.9970
SMOTE-Cosine	0.9932	0.9991	0.9873	0.9875	0.0127	0.0009	0.9932
SMOTE-OUT	0.9925	0.9991	0.9860	0.9862	0.0140	0.0009	0.9926
Selected-SMOTE	0.9932	0.9995	0.9869	0.9871	0.0131	0.0005	0.9932
Gaussian-SMOTE	0.9970	0.9941	0.9998	0.9998	0.0002	0.0059	0.9970
SN-SMOTE	0.9947	0.9995	0.9898	0.9899	0.0102	0.0005	0.9947
LLE-SMOTE	0.9966	0.9942	0.9990	0.9990	0.0010	0.0058	0.9966
Cluster-SMOTE	0.9961	0.9996	0.9926	0.9926	0.0074	0.0004	0.9961
SMOTE-IPF	0.9932	0.9994	0.9871	0.9872	0.0129	0.0006	0.9932
AND-SMOTE	0.9970	0.9962	0.9978	0.9978	0.0022	0.0038	0.9970
LN-SMOTE	0.9941	0.0340	0.9999	0.7217	0.0001	0.9660	0.5170
SMOTE-PSO	0.9935	0.7037	0.9994	0.9624	0.0006	0.2963	0.8516
NT-SMOTE	0.9948	0.9984	0.9912	0.9913	0.0088	0.0016	0.9948
Distance-SMOTE	0.9968	0.9992	0.9944	0.9944	0.0056	0.0008	0.9968
MWMOTE	0.9815	0.9885	0.9745	0.9749	0.0255	0.0115	0.9815
LVQ-SMOTE	0.9970	0.9941	0.9999	**0.9999**	0.0001	0.0059	0.9970
ADOMS	0.9956	**0.9997**	0.9915	0.9916	0.0085	**0.0003**	0.9956
ANS	0.9825	0.9904	0.9745	0.9749	0.0255	0.0096	0.9825
AHC	0.9907	0.2663	0.9994	0.8458	0.0006	0.7337	0.6328
SPY	0.9937	0.0491	0.9998	0.5745	0.0002	0.9509	0.5244

8.4.3.4 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using KNN classifier.*

In this experiment, the KNN classifier is considered as an ensemble method and used to discuss the impact of the proposed balancing technique (DBBI-SMOTE) on the classifier compared to the other 34 balancing methods. Thus, as stated in Table 8.15, DBBI-SMOTE leads the KNN classifier to obtain the highest *accuracy* and *AUC* metric value compared to the other balancing methods used to balance the Spanish companies' dataset. This means that the combination of DBBI-SMOTE and KNN yields better overall performance in predicting the minority instances(bankrupt companies) and the majority instances (solvent companies) compared to the other combination. Nevertheless, SMOTE, ADASYN, SMOTE-Tomek, SMOTE-ENN, ISMOTE, SDSMOTE, G-SMOTE, SMOTE-Cosine, SMOTE-OUT, Selected-SMOTE, Cluster-SMOTE Distance-SMOTE and ADOMS show very high performance in determining the minority instances (bankrupt companies) as stated in the *recall* and *type II error* metrics values; these balancing methods lead KNN to determine 100% of the minority class instances. But, this high performance comes at the expense of the other class instances prediction. The combination of SMOTE-D with KNN yields the best performance in determining the majority instances as stated in the *specificity* and *recall* metrics values.

Moreover, Table 8.16 shows the evaluation metrics values obtained by the combinations of KNN with DBBI-SMOTE and the other 34 balancing methods that applied to the Taiwanese companies' dataset individually. As it can be seen in the Table, DBBI-SMOTE leads KNN to the optimum performance in determining the minority instances. It obtains the highest overall *accuracy* and *recall*, and the lowest *type II error* metric value. However, the combinations of SMOTE, SMOTE Tomek, SMOTE-ENN, SDSMOTE and SNSMOTE with KNN show high performance in predicting the minority class instances as stated in the *recall* and *type II error*. On the other hand, SMOTE-D shows the lowest minority instances (solvent companies) misprediction. It obtains the lowest *type I error* and the highest *specificity* metric values.

In addition, also in the Polish companies dataset, the conjunction of DBBI-SMOTE with KNN classifier shows the highest overall performance in predicting and mispredicting the majority and the minority classes instances. As shown in Table 8.17, DBBI-SMOTE obtains the best *accuracy*, *precision* and *AUC* evaluation metrics values. G-SMOTE, Cluster-SMOTE, NT-SMOTE and Distance-SMOTE lead KNN to the best performance in determining the minority instances as stated in the *recall* and *type II error* metrics values. All of these combinations determined 100% of the minority instances. On the other side, the combination of SPY with KNN shows the best performance in predicting the majority instances as shown in the *specificity* and *type I error* metrics value, but at the expense of the other class. This combination shows very low performance in predicting the minority instances, this means that it is an extremely unreliable balancing method to process the Polish companies dataset.

Furthermore, also in the US companies dataset, the conjunction of DBBI-SMOTE with KNN yields the best overall performance in determining the minority and the majority class instances. As it can be seen in Table 8.18, the combination of DBBI-SMOTE and KNN classifier obtains the best *accuracy* and *AUC* metrics values. CE-SMOTE and LN-SMOTE show the highest performance in predicting the majority instances (solvent companies) according to the *specificity* and *type I error* metrics values. Both of them determine 100% of the majority instances. Whereas the combination of ISMOTE with KNN shows the highest performance in predicting the minority instances as stated in the *recall* and *type II error* metrics values.

In summary, DBBI-SMOTE leads the KNN classifier to achieve the best overall performance in determining the minority and the majority classes instances in the Spanish, Taiwanese, Polish and the US companies dataset.

Table 8.15: The performance metrics values obtained by KNN classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9891**	0.9818	0.9964	0.9964	0.0036	0.0182	**0.9891**
SMOTE	0.9753	**1.0**	0.9507	0.9531	0.0493	**0.0**	0.9753
BL-SMOTE	0.9793	0.9986	0.9600	0.9617	0.0400	0.0014	0.9793
SVM-SMOTE	0.9704	0.9955	0.9564	0.9277	0.0436	0.0045	0.9760
ADASYN	0.9833	**1.0**	0.9655	0.9687	0.0345	**0.0**	0.9828
SMOTE-NC	0.9703	0.9950	0.9456	0.9484	0.0544	0.0050	0.9703
SMOTE-Tomek	0.9762	**1.0**	0.9523	0.9547	0.0477	**0.0**	0.9762
SMOTE-ENN	0.9835	**1.0**	0.9659	0.9691	0.0341	**0.0**	0.9829
MSMOTE	0.9809	0.9928	0.9689	0.9698	0.0311	0.0072	0.9808
ISMOTE	0.9755	**1.0**	0.9510	0.9536	0.0490	**0.0**	0.9755
SL-SMOTE	0.9708	0.6901	0.9893	0.8197	0.0107	0.3099	0.8397
CE-SMOTE	0.9864	0.9936	0.9793	0.9796	0.0207	0.0064	0.9865
CURE-SMOTE	0.9852	0.9950	0.9753	0.9759	0.0247	0.0050	0.9851
SMOTE-D	0.9887	0.9800	**0.9975**	**0.9975**	**0.0025**	0.0200	0.9888
SDSMOTE	0.9757	**1.0**	0.9514	0.9538	0.0486	**0.0**	0.9757
G-SMOTE	0.9802	**1.0**	0.9603	0.9619	0.0397	**0.0**	0.9802
SMOTE-Cosine	0.9775	**1.0**	0.9550	0.9570	0.0450	**0.0**	0.9775
SMOTE-OUT	0.9768	**1.0**	0.9535	0.9557	0.0465	**0.0**	0.9768
Selected-SMOTE	0.9759	**1.0**	0.9517	0.9541	0.0483	**0.0**	0.9758
Gaussian-SMOTE	0.6893	0.3886	0.9900	0.9749	0.0100	0.6114	0.6893
SN-SMOTE	0.9787	0.9996	0.9578	0.9596	0.0422	0.0004	0.9787
LLE-SMOTE	0.9886	0.9843	0.9928	0.9928	0.0072	0.0157	0.9886
Cluster-SMOTE	0.9785	**1.0**	0.9571	0.9589	0.0429	**0.0**	0.9786
SMOTE-IPF	0.9757	0.9996	0.9517	0.9540	0.0483	0.0004	0.9757
AND-SMOTE	0.9861	0.9929	0.9793	0.9796	0.0207	0.0071	0.9861
LN-SMOTE	0.9825	0.9939	0.9710	0.9718	0.0290	0.0061	0.9825
SMOTE-PSO	0.9642	0.8753	0.9721	0.7433	0.0279	0.1247	0.9237
NT-SMOTE	0.9777	0.9993	0.9560	0.9580	0.0440	0.0007	0.9776
SMOTE-RS*B*∗	0.9777	0.1905	0.9971	0.6867	0.0029	0.8095	0.5938
Distance-SMOTE	0.9809	**1.0**	0.9618	0.9632	0.0382	**0.0**	0.9809
MWMOTE	0.9755	0.9989	0.9521	0.9544	0.0479	0.0011	0.9755
LVQ-SMOTE	0.9868	0.9828	0.9907	0.9906	0.0093	0.0172	0.9868
ADOMS	0.9771	**1.0**	0.9542	0.9563	0.0458	**0.0**	0.9771
ANS	0.9787	0.9968	0.9607	0.9622	0.0393	0.0032	0.9788
AHC	0.9774	0.6346	0.9925	0.7946	0.0075	0.3654	0.8135
SPY	0.9773	0.1286	0.9975	0.4667	0.0025	0.8714	0.5631

Table 8.16: The performance metrics values obtained by KNN classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9845**	**1.0**	0.9668	0.9714	0.0332	**0.0**	0.9834
SMOTE	0.9478	**1.0**	0.8956	0.9056	0.1044	**0.0**	0.9478
BL-SMOTE	0.9565	0.9965	0.9165	0.9227	0.0835	0.0035	0.9565
SVM-SMOTE	0.9512	0.9783	0.9417	0.8547	0.0583	0.0217	0.9600
ADASYN	0.9467	0.9998	0.8941	0.9032	0.1059	0.0002	0.9469
SMOTE-Tomek	0.9481	**1.0**	0.8963	0.9060	0.1037	**0.0**	0.9482
SMOTE-ENN	0.9843	**1.0**	0.9665	0.9714	0.0335	**0.0**	0.9832
MSMOTE	0.9617	0.9853	0.9380	0.9410	0.0620	0.0147	0.9616
ISMOTE	0.9207	0.9903	0.8510	0.8699	0.1490	0.0097	0.9206
SL-SMOTE	0.9239	0.8535	0.9395	0.7588	0.0605	0.1465	0.8965
CE-SMOTE	0.9625	0.9911	0.9339	0.9376	0.0661	0.0089	0.9625
CURE-SMOTE	0.9576	0.9976	0.9177	0.9238	0.0823	0.0024	0.9577
SMOTE-D	0.9830	0.9678	**0.9983**	**0.9983**	**0.0017**	0.0322	0.9830
SDSMOTE	0.9479	**1.0**	0.8959	0.9058	0.1041	**0.0**	0.9480
G-SMOTE	0.9567	0.9994	0.9139	0.9208	0.0861	0.0006	0.9567
SMOTE-Cosine	0.9501	0.9995	0.9007	0.9097	0.0993	0.0005	0.9501
SMOTE-OUT	0.9520	0.9998	0.9042	0.9127	0.0958	0.0002	0.9520
Selected-SMOTE	0.9513	0.9997	0.9029	0.9115	0.0971	0.0003	0.9513
Gaussian-SMOTE	0.6481	0.3067	0.9895	0.9669	0.0105	0.6933	0.6481
SN-SMOTE	0.9512	**1.0**	0.9024	0.9112	0.0976	**0.0**	0.9512
LLE-SMOTE	0.9798	0.9820	0.9776	0.9777	0.0224	0.0180	0.9798
Cluster-SMOTE	0.9532	0.9998	0.9065	0.9146	0.0935	0.0002	0.9531
SMOTE-IPF	0.9476	0.9997	0.8956	0.9055	0.1044	0.0003	0.9476
AND-SMOTE	0.9734	0.9867	0.9601	0.9612	0.0399	0.0133	0.9734
LN-SMOTE	0.9691	0.2000	0.9947	0.5758	0.0053	0.8000	0.5974
SMOTE-PSO	0.9535	0.7182	0.9848	0.8635	0.0152	0.2818	0.8515
NT-SMOTE	0.9530	0.9992	0.9068	0.9148	0.0932	0.0008	0.9530
SMOTE-RS$B*$	0.9678	0.2237	0.9930	0.5519	0.0070	0.7763	0.6083
Distance-SMOTE	0.9558	0.9998	0.9117	0.9189	0.0883	0.0002	0.9557
MWMOTE	0.9492	0.9985	0.9000	0.9090	0.1000	0.0015	0.9493
LVQ-SMOTE	0.9842	0.9770	0.9914	0.9912	0.0086	0.0230	0.9842
ADOMS	0.9589	0.9998	0.9180	0.9244	0.0820	0.0002	0.9589
ANS	0.9526	0.9974	0.9077	0.9154	0.0923	0.0026	0.9526
AHC	0.9609	0.6765	0.9798	0.6909	0.0202	0.3235	0.8281
SPY	0.9676	0.2107	0.9932	0.5218	0.0068	0.7893	0.6019

Table 8.17: The performance metrics values obtained by KNN classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9892**	0.9793	0.9992	**0.9992**	0.0008	0.0207	**0.9892**
SMOTE	0.9455	0.9997	0.8914	0.9020	0.1086	0.0003	0.9456
BL-SMOTE	0.9649	0.9959	0.9340	0.9379	0.0660	0.0041	0.9649
SVM-SMOTE	0.9566	0.9904	0.9403	0.8886	0.0597	0.0096	0.9653
ADASYN	0.9448	0.9997	0.8895	0.9011	0.1105	0.0003	0.9446
SMOTE-Tomek	0.9453	0.9992	0.8915	0.9022	0.1085	0.0008	0.9453
SMOTE-ENN	0.9788	0.9999	0.9546	0.9618	0.0454	0.0001	0.9772
MSMOTE	0.9786	0.9890	0.9682	0.9688	0.0318	0.0110	0.9786
ISMOTE	0.9388	0.9986	0.8790	0.8922	0.1210	0.0014	0.9388
SL-SMOTE	0.9585	0.8850	0.9692	0.8061	0.0308	0.1150	0.9271
CE-SMOTE	0.9792	0.0050	0.9994	0.1000	0.0006	0.9950	0.5022
CURE-SMOTE	0.9583	0.9967	0.9199	0.9256	0.0801	0.0033	0.9583
SMOTE-D	0.9754	0.9965	0.9543	0.9562	0.0457	0.0035	0.9754
SDSMOTE	0.9444	0.9997	0.8891	0.9002	0.1109	0.0003	0.9444
G-SMOTE	0.9598	**1.0**	0.9196	0.9256	0.0804	**0.0**	0.9598
SMOTE-Cosine	0.9456	0.9993	0.8920	0.9025	0.1080	0.0007	0.9456
SMOTE-OUT	0.9461	0.9993	0.8929	0.9033	0.1071	0.0007	0.9461
Selected-SMOTE	0.9445	0.9997	0.8894	0.9004	0.1106	0.0003	0.9446
Gaussian-SMOTE	0.5091	0.0199	0.9983	0.9229	0.0017	0.9801	0.5091
SN-SMOTE	0.9519	0.9995	0.9044	0.9128	0.0956	0.0005	0.9520
LLE-SMOTE	0.9783	0.9836	0.9731	0.9733	0.0269	0.0164	0.9784
Cluster-SMOTE	0.9577	**1.0**	0.9154	0.9221	0.0846	**0.0**	0.9577
SMOTE-IPF	0.9447	0.9995	0.8900	0.9009	0.1100	0.0005	0.9447
AND-SMOTE	0.9808	0.9896	0.9720	0.9725	0.0280	0.0104	0.9808
LN-SMOTE	0.9778	0.9924	0.9632	0.9642	0.0368	0.0076	0.9778
SMOTE-PSO	0.9598	0.8301	0.9705	0.7007	0.0295	0.1699	0.9003
NT-SMOTE	0.9556	**1.0**	0.9112	0.9185	0.0888	**0.0**	0.9556
SMOTE-RS*B*∗	0.9790	0.0426	0.9992	0.3567	0.0008	0.9574	0.5209
Distance-SMOTE	0.9527	**1.0**	0.9055	0.9137	0.0945	**0.0**	0.9527
MWMOTE	0.9250	0.9860	0.8640	0.8790	0.1360	0.0140	0.9250
LVQ-SMOTE	0.9852	0.9812	0.9892	0.9891	0.0108	0.0188	0.9852
ADOMS	0.9521	0.9998	0.9045	0.9128	0.0955	0.0002	0.9522
ANS	0.9281	0.9829	0.8734	0.8860	0.1266	0.0171	0.9282
AHC	0.9679	0.5752	0.9842	0.6055	0.0158	0.4248	0.7797
SPY	0.9795	0.0098	**0.9996**	0.1500	**0.0004**	0.9902	0.5047

Table 8.18: The performance metrics values obtained by KNN classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9965**	0.9966	0.9964	0.9964	0.0036	0.0034	**0.9965**
SMOTE	0.9492	0.9574	0.9409	0.9419	0.0591	0.0426	0.9491
BL-SMOTE	0.9850	0.9918	0.9782	0.9785	0.0218	0.0082	0.9850
SMOTE-ENN	0.9505	0.9560	0.9447	0.9471	0.0553	0.0440	0.9504
SMOTE-Tomek	0.9491	0.9573	0.9408	0.9417	0.0592	0.0427	0.9490
SVM-SMOTE	0.9793	0.9857	0.9728	0.9732	0.0272	0.0143	0.9792
ADASYN	0.9488	0.9587	0.9388	0.9400	0.0612	0.0413	0.9487
MSMOTE	0.9920	0.9965	0.9876	0.9877	0.0124	0.0035	0.9921
ISMOTE	0.9618	**0.9984**	0.9252	0.9303	0.0748	**0.0016**	0.9618
SL-SMOTE	0.9906	0.6456	0.9984	0.8995	0.0016	0.3544	0.8220
CE-SMOTE	0.9939	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
CURE-SMOTE	0.9863	0.9869	0.9857	0.9858	0.0143	0.0131	0.9863
SMOTE-D	0.9895	0.9847	0.9943	0.9943	0.0057	0.0153	0.9895
SDSMOTE	0.9501	0.9587	0.9416	0.9426	0.0584	0.0413	0.9502
G-SMOTE	0.9744	0.9956	0.9533	0.9552	0.0467	0.0044	0.9745
SMOTE-Cosine	0.9545	0.9659	0.9430	0.9443	0.0570	0.0341	0.9545
SMOTE-OUT	0.9508	0.9607	0.9409	0.9420	0.0591	0.0393	0.9508
Selected-SMOTE	0.9493	0.9577	0.9409	0.9419	0.0591	0.0423	0.9493
Gaussian-SMOTE	0.9925	0.9854	0.9997	**0.9997**	0.0003	0.0146	0.9926
SN-SMOTE	0.9593	0.9720	0.9467	0.9480	0.0533	0.0280	0.9593
LLE-SMOTE	0.9958	0.9944	0.9972	0.9972	0.0028	0.0056	0.9958
Cluster-SMOTE	0.9669	0.9859	0.9480	0.9499	0.0520	0.0141	0.9669
SMOTE-IPF	0.9487	0.9578	0.9397	0.9408	0.0603	0.0422	0.9487
AND-SMOTE	0.9911	0.9966	0.9855	0.9857	0.0145	0.0034	0.9910
LN-SMOTE	0.9940	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
SMOTE-PSO	0.9844	0.5628	0.9930	0.6232	0.0070	0.4372	0.7779
NT-SMOTE	0.9568	0.9637	0.9498	0.9505	0.0502	0.0363	0.9567
Distance-SMOTE	0.9783	0.9965	0.9600	0.9614	0.0400	0.0035	0.9783
MWMOTE	0.9300	0.9076	0.9523	0.9501	0.0477	0.0924	0.9300
LVQ-SMOTE	0.9965	0.9941	0.9989	0.9989	0.0011	0.0059	0.9965
ADOMS	0.9692	0.9855	0.9530	0.9545	0.0470	0.0145	0.9692
ANS	0.9327	0.9102	0.9553	0.9532	0.0447	0.0898	0.9327
AHC	0.9885	0.0637	0.9997	0.6822	0.0003	0.9363	0.5317
SPY	0.9936	0.0017	0.9999	0.0500	0.0001	0.9983	0.5008

8.4.3.5 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using SVM classifier.*

Actually, another high-performance classifier (i.e., SVM) is devoted in this experiment to evaluating the performance of the new proposed DBBI-SMOTE balancing method and comparing it with many balancing techniques used in the literature to solve the data inconsistent distribution problem. Thus, DBBI-SMOTE and the remaining balancing methods are used to generate new balanced datasets from the Spanish, Taiwanese, Polish and US companies datasets, and then these datasets have been used to train the SVM classifier. Thus, referring to the Spanish companies dataset. As shown in Table 8.19, the combination of DBBI-SMOTE with SVM shows the highest overall performance in predicting the minority and majority classes instances as stated in the *accuracy* and *AUC* evaluation metrics values. On the other hand, SMOTE-ENN leads the SVM to the lowest minority instances (bankrupt companies) misprediction. It obtains the lowest *type II error* and the highest *recall* values. Whereas CURE-SMOTE in conjunction with SVM shows the optimum performance in determining the majority instances according to the *specificity* and *type I error* evaluation metrics values.

Furthermore, with respect to the Taiwanese companies dataset, as shown in Table 8.20, DBBI-SMOTE leads SVM to obtain the highest *recall* and the lowest *type II error* metrics values. This means that the combination of DBBI-SMOTE with SVM is the optimum one to determine the minority class instances (bankrupt companies). On the other hand, SMOTE-D in conjunction with SVM shows the highest performance in determining the majority class instances in the Taiwanese companies' dataset. It obtains the highest *specificity* and *precision*, and the lowest *type I error* metrics values.

However, regarding the Polish companies dataset, DBBI-SMOTE also shows the best performance in predicting the minority class instances according to the *recall* and *type II error* metrics values in Table 8.22. On the other hand, also in this dataset, the combination of SMOTE-D with SVM returns the best performance in determining the majority class instances as stated in the *specificity* and *type I error* metrics values. Nevertheless, CE-SMOTE, SMOTE-PSO, AHC and SPY show very high performance in predicting the majority class instances, but at the expense of the minority class instances prediction; they failed to determine any minority instances. Thus, these balancing methods are unreliable to balance the Polish companies dataset to be utilized to train SVM.

Regarding to the US companies dataset, as shown in Table 8.22, Gaussian-SMOTE leads the SVM to obtain the best *accuracy*, *recall*, *AUC* and *type II error*. This means that this approach is the optimum one to predict the minority instances compared to the other 34 approaches. Nevertheless, referring to the majority class instances predicting and mispredicting, several classifiers show very high performance in it, but also in the expense of the minority instances prediction. In other words, some balancing methods lead SVM to determine 100% of the majority instances, but 0% of the minority instances.

Thus, as a firm conclusion, DBBI-SMOTE mostly leads the SVM classifier to the optimum behavior regarding determining the minority instances (bankrupt companies). It obtains the best overall performance in the Spanish companies dataset, and the best approach to determine the minority instances in the Taiwanese and Polish companies' datasets.

Table 8.19: The performance metrics values obtained by SVM classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9875**	0.9907	0.9843	0.9844	0.0157	0.0093	**0.9875**
SMOTE	0.9694	0.9796	0.9592	0.9602	0.0408	0.0204	0.9694
BL-SMOTE	0.9726	0.9782	0.9671	0.9675	0.0329	0.0218	0.9727
SVM-SMOTE	0.9697	0.9630	0.9735	0.9539	0.0265	0.0370	0.9683
ADASYN	0.9806	0.9913	0.9691	0.9716	0.0309	0.0087	0.9802
SMOTE-NC	0.9680	0.9864	0.9496	0.9515	0.0504	0.0136	0.9680
SMOTE-Tomek	0.9697	0.9791	0.9603	0.9611	0.0397	0.0209	0.9697
SMOTE-ENN	0.9825	**0.9940**	0.9703	0.9729	0.0297	**0.0060**	0.9822
MSMOTE	0.9814	0.9864	0.9764	0.9767	0.0236	0.0136	0.9814
ISMOTE	0.9748	0.9832	0.9664	0.9671	0.0336	0.0168	0.9748
SL-SMOTE	0.9658	0.7602	0.9793	0.7081	0.0207	0.2398	0.8698
CE-SMOTE	0.9855	0.9875	0.9836	0.9837	0.0164	0.0125	0.9856
CURE-SMOTE	0.9861	0.9861	**0.9861**	**0.9861**	**0.0139**	0.0139	0.9861
SMOTE-D	0.9859	0.9868	0.9850	0.9851	0.0150	0.0132	0.9859
DSMOTE	0.9830	0.9839	0.9822	0.9822	0.0178	0.0161	0.9830
SDSMOTE	0.9837	0.9886	0.9789	0.9792	0.0211	0.0114	0.9838
G-SMOTE	0.9843	0.9903	0.9782	0.9785	0.0218	0.0097	0.9842
SMOTE-Cosine	0.9787	0.9853	0.9721	0.9726	0.0279	0.0147	0.9787
SMOTE-OUT	0.9793	0.9843	0.9743	0.9746	0.0257	0.0157	0.9793
Selected-SMOTE	0.9812	0.9882	0.9743	0.9747	0.0257	0.0118	0.9812
Gaussian-SMOTE	0.9852	0.9864	0.9839	0.9840	0.0161	0.0136	0.9851
SN-SMOTE	0.9821	0.9875	0.9768	0.9770	0.0232	0.0125	0.9822
LLE-SMOTE	0.9836	0.9846	0.9825	0.9826	0.0175	0.0154	0.9836
Cluster-SMOTE	0.9868	0.9918	0.9818	0.9820	0.0182	0.0082	0.9868
SMOTE-IPF	0.9825	0.9889	0.9760	0.9764	0.0240	0.0111	0.9825
AND-SMOTE	0.9868	0.9918	0.9818	0.9820	0.0182	0.0082	0.9868
LN-SMOTE	0.9839	0.9903	0.9775	0.9778	0.0225	0.0097	0.9839
SMOTE-PSO	0.9727	0.8458	0.9839	0.8246	0.0161	0.1542	0.9148
NT-SMOTE	0.9807	0.9857	0.9757	0.9761	0.0243	0.0143	0.9807
SMOTE-RS*B**	0.9700	0.4333	0.9832	0.4340	0.0168	0.5667	0.7083
Distance-SMOTE	0.9855	0.9864	0.9846	0.9847	0.0154	0.0136	0.9855
MWMOTE	0.9809	0.9809	0.9746	0.9750	0.0254	0.0129	0.9808
LVQ-SMOTE	0.9873	0.9903	0.9843	0.9844	0.0157	0.0097	0.9873
ADOMS	0.9855	0.9882	0.9828	0.9830	0.0172	0.0118	0.9855
ANS	0.9843	0.9889	0.9796	0.9799	0.0204	0.0111	0.9843
AHC	0.9709	0.6744	0.9839	0.6519	0.0161	0.3256	0.8292
SPY	0.9717	0.4310	0.9846	0.4577	0.0154	0.5690	0.7078

Table 8.20: The performance metrics values obtained by SVM classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9783	**0.9924**	0.9729	0.9832	0.0271	**0.0076**	0.9783
SMOTE	0.9560	0.9848	0.9271	0.9311	0.0729	0.0152	0.9560
BL-SMOTE	0.9639	0.9851	0.9426	0.9450	0.0574	0.0149	0.9638
SVM-SMOTE	0.9498	0.9125	0.9629	0.8958	0.0371	0.0875	0.9377
ADASYN	**0.9885**	0.9784	0.9987	0.9986	0.0013	0.0216	**0.9886**
SMOTE-Tomek	0.9543	0.9837	0.9249	0.9291	0.0751	0.0163	0.9543
SMOTE-ENN	0.9741	0.9921	0.9535	0.9605	0.0465	0.0079	0.9728
MSMOTE	0.9601	0.9797	0.9406	0.9429	0.0594	0.0203	0.9602
ISMOTE	0.9158	0.9320	0.8997	0.9030	0.1003	0.0680	0.9159
SL-SMOTE	0.8688	0.3269	0.9894	0.8784	0.0106	0.6731	0.6582
CE-SMOTE	0.9662	0.9839	0.9485	0.9504	0.0515	0.0161	0.9662
CURE-SMOTE	0.9711	0.9785	0.9636	0.9642	0.0364	0.0215	0.9710
SMOTE-D	0.9823	0.9647	**0.9998**	**0.9998**	**0.0002**	0.0353	0.9823
SDSMOTE	0.9568	0.9862	0.9274	0.9316	0.0726	0.0138	0.9568
G-SMOTE	0.9604	0.9874	0.9335	0.9370	0.0665	0.0126	0.9605
SMOTE-Cosine	0.9573	0.9830	0.9317	0.9351	0.0683	0.0170	0.9573
SMOTE-OUT	0.9594	0.9861	0.9327	0.9362	0.0673	0.0139	0.9594
Selected-SMOTE	0.9567	0.9859	0.9274	0.9316	0.0726	0.0141	0.9567
Gaussian-SMOTE	0.9734	0.9729	0.9739	0.9740	0.0261	0.0271	0.9734
SN-SMOTE	0.9554	0.9850	0.9259	0.9301	0.0741	0.0150	0.9554
LLE-SMOTE	0.9817	0.9753	0.9880	0.9879	0.0120	0.0247	0.9816
Cluster-SMOTE	0.9586	0.9820	0.9353	0.9382	0.0647	0.0180	0.9587
SMOTE-IPF	0.9562	0.9858	0.9267	0.9309	0.0733	0.0142	0.9563
AND-SMOTE	0.9734	0.9855	0.9614	0.9623	0.0386	0.0145	0.9735
LN-SMOTE	0.9679	0.0136	0.9997	0.2500	0.0003	0.9864	0.5067
SMOTE-PSO	0.9495	0.6523	0.9891	0.8897	0.0109	0.3477	0.8207
NT-SMOTE	0.9613	0.9835	0.9391	0.9417	0.0609	0.0165	0.9613
SMOTE-RS$B*$	0.9683	0.0449	0.9997	0.6000	0.0003	0.9551	0.5223
Distance-SMOTE	0.9679	0.9838	0.9521	0.9536	0.0479	0.0162	0.9679
MWMOTE	0.9597	0.9823	0.9371	0.9399	0.0629	0.0177	0.9597
LVQ-SMOTE	0.9822	0.9777	0.9867	0.9866	0.0133	0.0223	0.9822
ADOMS	0.9407	0.9689	0.9124	0.9173	0.0876	0.0311	0.9406
ANS	0.9586	0.9805	0.9367	0.9393	0.0633	0.0195	0.9586
AHC	0.9557	0.3711	0.9945	0.8275	0.0055	0.6289	0.6828
SPY	0.9676	0.0221	0.9995	0.4500	0.0005	0.9779	0.5108

Table 8.21: The performance metrics values obtained by SVM classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9350	**0.9927**	0.8773	0.8900	0.1227	**0.0073**	0.9350
SMOTE	0.8660	0.9694	0.7626	0.8034	0.2374	0.0306	0.8660
BL-SMOTE	0.9136	0.9824	0.8447	0.8636	0.1553	0.0176	0.9136
SVM-SMOTE	0.9056	0.9023	0.9072	0.8240	0.0928	0.0977	0.9047
ADASYN	0.8629	0.9686	0.7567	0.8004	0.2433	0.0314	0.8627
SMOTE-Tomek	0.8655	0.9692	0.7618	0.8029	0.2382	0.0308	0.8655
SMOTE-ENN	0.9018	0.9846	0.8074	0.8536	0.1926	0.0154	0.8960
MSMOTE	0.8949	0.9691	0.8208	0.8440	0.1792	0.0309	0.8949
ISMOTE	0.9044	0.9382	0.8706	0.8788	0.1294	0.0618	0.9044
SL-SMOTE	0.8786	0.0473	0.9989	0.8663	0.0011	0.9527	0.5231
CE-SMOTE	0.9797	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
CURE-SMOTE	0.9228	0.9591	0.8865	0.8943	0.1135	0.0409	0.9228
SMOTE-D	**0.9890**	0.9781	**1.0**	**0.9999**	**0.0**	0.0219	**0.9890**
SDSMOTE	0.8623	0.9641	0.7604	0.8010	0.2396	0.0359	0.8622
G-SMOTE	0.8673	0.9695	0.7650	0.8049	0.2350	0.0305	0.8673
SMOTE-Cosine	0.8724	0.9664	0.7783	0.8134	0.2217	0.0336	0.8723
SMOTE-OUT	0.8712	0.9669	0.7754	0.8116	0.2246	0.0331	0.8711
Selected-SMOTE	0.8636	0.9712	0.7559	0.7992	0.2441	0.0288	0.8636
Gaussian-SMOTE	0.9828	0.9795	0.9861	0.9860	0.0139	0.0205	0.9828
SN-SMOTE	0.8617	0.9708	0.7526	0.7969	0.2474	0.0292	0.8617
LLE-SMOTE	0.9349	0.9706	0.8993	0.9060	0.1007	0.0294	0.9349
Cluster-SMOTE	0.8711	0.9543	0.7880	0.8182	0.2120	0.0457	0.8712
SMOTE-IPF	0.8648	0.9689	0.7606	0.8020	0.2394	0.0311	0.8648
AND-SMOTE	0.8803	0.9606	0.8000	0.8278	0.2000	0.0394	0.8803
LN-SMOTE	0.9019	0.9756	0.8282	0.8503	0.1718	0.0244	0.9019
SMOTE-PSO	0.9251	0.0209	**1.0**	**0.9999**	**0.0**	0.9791	0.5104
NT-SMOTE	0.8685	0.9695	0.7675	0.8066	0.2325	0.0305	0.8685
SMOTE-RS$B*$	0.9793	0.0235	0.9999	0.4000	0.0001	0.9765	0.5117
Distance-SMOTE	0.8733	0.9744	0.7722	0.8105	0.2278	0.0256	0.8733
MWMOTE	0.8829	0.9745	0.7914	0.8240	0.2086	0.0255	0.8830
LVQ-SMOTE	0.9211	0.9427	0.8995	0.9036	0.1005	0.0573	0.9211
ADOMS	0.8539	0.9606	0.7472	0.7917	0.2528	0.0394	0.8539
ANS	0.8935	0.9757	0.8114	0.8380	0.1886	0.0243	0.8936
AHC	0.9603	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
SPY	0.9797	0.0	**1.0**	0.0	**0.0**	1.0	0.5000

Table 8.22: The performance metrics values obtained by SVM classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.8571	0.9179	0.7963	0.8184	0.2037	0.0821	0.8571
SMOTE	0.7869	0.8321	0.7417	0.7632	0.2583	0.1679	0.7869
BL-SMOTE	0.8727	0.9388	0.8066	0.8292	0.1934	0.0612	0.8727
SMOTE-ENN	0.7925	0.8421	0.7411	0.7712	0.2589	0.1579	0.7916
SMOTE-Tomek	0.7865	0.8346	0.7384	0.7614	0.2616	0.1654	0.7865
SVM-SMOTE	0.8674	0.9204	0.8145	0.8322	0.1855	0.0796	0.8675
ADASYN	0.7755	0.8269	0.7242	0.7499	0.2758	0.1731	0.7755
MSMOTE	0.8628	0.9266	0.7990	0.8218	0.2010	0.0734	0.8628
ISMOTE	0.8452	0.7972	0.8932	0.8818	0.1068	0.2028	0.8452
SL-SMOTE	0.9780	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
CE-SMOTE	0.9940	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
CURE-SMOTE	0.8786	0.8872	0.8700	0.8723	0.1300	0.1128	0.8786
SMOTE-D	0.9632	0.9529	0.9735	0.9729	0.0265	0.0471	0.9632
SDSMOTE	0.7847	0.8337	0.7356	0.7592	0.2644	0.1663	0.7847
G-SMOTE	0.7843	0.8343	0.7343	0.7585	0.2657	0.1657	0.7843
SMOTE-Cosine	0.7922	0.8477	0.7367	0.7630	0.2633	0.1523	0.7922
SMOTE-OUT	0.7922	0.8415	0.7429	0.7662	0.2571	0.1585	0.7922
Selected-SMOTE	0.7836	0.8325	0.7346	0.7583	0.2654	0.1675	0.7835
Gaussian-SMOTE	**0.9949**	**0.9933**	0.9964	0.9964	0.0036	**0.0067**	**0.9949**
SN-SMOTE	0.7806	0.8264	0.7349	0.7571	0.2651	0.1736	0.7807
LLE-SMOTE	0.8797	0.9362	0.8231	0.8411	0.1769	0.0638	0.8797
Cluster-SMOTE	0.7837	0.8309	0.7365	0.7592	0.2635	0.1691	0.7837
SMOTE-IPF	0.7842	0.8333	0.7351	0.7588	0.2649	0.1667	0.7842
AND-SMOTE	0.8880	0.9570	0.8190	0.8410	0.1810	0.0430	0.8880
LN-SMOTE	0.9940	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
SMOTE-PSO	0.9800	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
NT-SMOTE	0.7865	0.8402	0.7328	0.7587	0.2672	0.1598	0.7865
Distance-SMOTE	0.7958	0.8572	0.7345	0.7635	0.2655	0.1428	0.7958
MWMOTE	0.8279	0.8993	0.7565	0.7869	0.2435	0.1007	0.8279
LVQ-SMOTE	0.9074	0.8269	0.9879	0.9855	0.0121	0.1731	0.9074
ADOMS	0.7756	0.8141	0.7371	0.7559	0.2629	0.1859	0.7756
ANS	0.8227	0.8938	0.7517	0.7826	0.2483	0.1062	0.8228
AHC	0.9881	0.0	**1.0**	0.0	**0.0**	1.0	0.5000
SPY	0.9936	0.0	**1.0**	0.0	**0.0**	1.0	0.5000

8.4.3.6 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using AdaBoost classifier.*

Basically, to evaluate the performance of DBBI-SMOTE comprehensively, it has been used to prepare several datasets before using it to train several different types of classifiers. In this experiment, to achieve more comprehensiveness during analyzing the results of the new proposed method, we have applied a boosting-based classification algorithm (i.e., AdaBoost) to the datasets generated by DBBI-SMOTE and the other 34 balancing methods to make a good judgment about the new method compared to the others. Thus, regarding the Spanish companies dataset, as shown in Table 8.23, the combination of ADASYN ad AdaBoost obtains the best *recall* and *type II error* metrics values. This means that ADASYN + AdaBoost is the best approach to determine the minority instances in the Spanish companies' dataset. Whereas the conjunction of CE-SMOTE with AdaBoost shows the highest performance in determining the majority instances as stated in the evaluation metrics. This approach obtained the highest *specificity*, *precision* and the lowest *type I error* metrics values. On the other hand, the combination of DBBI-SMOTE with AdaBoost outperforms other 15 approaches.

In addition, referring to the Taiwanese companies dataset, the combination of DBBI-SMOTE with AdaBoost classifier shows the highest performance in determining the minority instances as stated in Table 8.24. It obtains the best *recall* and *type I error* metrics values. On the other hand, LN-SMOTE with AdaBoost yields the optimum performance in determining the majority class instances (solvent companies) according to the evaluation metrics values. LN-SMOTE obtains the highest *specificity* and the lowest *type I error*. Nevertheless, SMOTE-D obtains the best *accuracy* and *AUC* metrics values.

Regarding the Polish companies dataset, as shown in Table 8.25, the combination of SMOTE-D with AdaBoost shows the best overall performance in predicting the majority and the minority classes instances. This approach obtains the highest *accuracy*, *precision* and *AUC* values. On the other hand, SMOTE-RS*B*∗ shows the optimum performance in determining the majority instances according to the evaluation metrics values. But in the expense of the minority class instances prediction. The approach of SMOTE-RS*B*∗ + AdaBoost is the superior in predicting the majority instances, but shows extremely low performance in predicting the minority instances, thus, it could be considered as an unreliable approach to predict the majority and the minority class instances in the Polish companies dataset.

Furthermore, with respect to the US companies dataset, as shown in Table 8.26, the combination of DBBI-SMOTE with AdaBoost classifier, yields the best overall performance in predicting the minority and the majority classes instances. This approach obtains the highest *accuracy*, *precision* and *AUC* metrics values compared to the other approaches. On the other hand, CE-SMOTE with AdaBoost return the optimum performance in predicting the majority class instances (solvent companies). It obtain the highest *specificity* and the lowest *type I error*. In addition, SPY obtains the same *specificity* and *type I error* of the CE-SMOTE, but SPY is not considered as a robust balancing method, because it is catastrophically misclassified the minority instances.

Table 8.23: The performance metrics values obtained by AdaBoost classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9898	0.9903	0.9893	0.9893	0.0107	0.0097	0.9898
SMOTE	0.9873	0.9946	0.9800	0.9803	0.0200	0.0054	0.9873
BL-SMOTE	0.9864	0.9954	0.9775	0.9779	0.0225	0.0046	0.9865
SVM-SMOTE	0.9828	0.9879	0.9800	0.9651	0.0200	0.0121	0.9839
ADASYN	0.9911	**0.9970**	0.9847	0.9859	0.0153	**0.0030**	0.9909
SMOTE-NC	0.9828	0.9946	0.9710	0.9718	0.0290	0.0054	0.9828
SMOTE-Tomek	0.9886	0.9957	0.9816	0.9819	0.0184	0.0043	0.9887
SMOTE-ENN	**0.9916**	0.9962	0.9867	0.9877	0.0133	0.0038	**0.9914**
MSMOTE	0.9852	0.9911	0.9793	0.9796	0.0207	0.0089	0.9852
ISMOTE	0.9839	0.9895	0.9783	0.9787	0.0217	0.0105	0.9839
SL-SMOTE	0.9758	0.7608	0.9900	0.8408	0.0100	0.2392	0.8754
CE-SMOTE	0.9911	0.9878	**0.9943**	**0.9943**	**0.0057**	0.0122	0.9910
CURE-SMOTE	0.9902	0.9903	0.9900	0.9900	0.0100	0.0097	0.9901
SMOTE-D	0.9898	0.9864	0.9932	0.9932	0.0068	0.0136	0.9898
SDSMOTE	0.9871	0.9961	0.9782	0.9786	0.0218	0.0039	0.9871
G-SMOTE	0.9862	0.9946	0.9778	0.9783	0.0222	0.0054	0.9862
SMOTE-Cosine	0.9882	0.9896	0.9868	0.9869	0.0132	0.0104	0.9882
SMOTE-OUT	0.9878	0.9918	0.9839	0.9841	0.0161	0.0082	0.9879
Selected-SMOTE	0.9877	0.9964	0.9789	0.9793	0.0211	0.0036	0.9876
Gaussian-SMOTE	0.9886	0.9850	0.9921	0.9921	0.0079	0.0150	0.9886
SN-SMOTE	0.9869	0.9950	0.9789	0.9793	0.0211	0.0050	0.9869
LLE-SMOTE	0.9868	0.9850	0.9886	0.9886	0.0114	0.0150	0.9868
Cluster-SMOTE	0.9866	0.9939	0.9793	0.9796	0.0207	0.0061	0.9866
SMOTE-IPF	0.9871	0.9968	0.9775	0.9779	0.0225	0.0032	0.9871
AND-SMOTE	0.9898	0.9928	0.9868	0.9869	0.0132	0.0072	0.9898
LN-SMOTE	0.9864	0.9950	0.9778	0.9784	0.0222	0.0050	0.9864
SMOTE-PSO	0.9777	0.7990	0.9936	0.9196	0.0064	0.2010	0.8963
NT-SMOTE	0.9891	0.9939	0.9843	0.9845	0.0157	0.0061	0.9891
SMOTE-RS*B*∗	0.9784	0.3929	0.9928	0.6255	0.0072	0.6071	0.6928
Distance-SMOTE	0.9891	0.9878	0.9903	0.9904	0.0097	0.0122	0.9890
MWMOTE	0.9870	0.9921	0.9818	0.9820	0.0182	0.0079	0.9869
LVQ-SMOTE	0.9891	0.9886	0.9896	0.9896	0.0104	0.0114	0.9891
ADOMS	0.9893	0.9893	0.9893	0.9893	0.0107	0.0107	0.9893
ANS	0.9877	0.9921	0.9832	0.9834	0.0168	0.0079	0.9876
AHC	0.9812	0.7481	0.9914	0.7943	0.0086	0.2519	0.8698
SPY	0.9783	0.3929	0.9925	0.5978	0.0075	0.6071	0.6927

Table 8.24: The performance metrics values obtained by AdaBoost classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9702	**0.9871**	0.9533	0.9549	0.0467	**0.0129**	0.9702
SMOTE	0.9343	0.9454	0.9232	0.9250	0.0768	0.0546	0.9343
BL-SMOTE	0.9545	0.9645	0.9445	0.9457	0.0555	0.0355	0.9545
SVM-SMOTE	0.9449	0.9016	0.9600	0.8878	0.0400	0.0984	0.9308
ADASYN	0.9417	0.9528	0.9307	0.9316	0.0693	0.0472	0.9417
SMOTE-Tomek	0.9440	0.9538	0.9342	0.9355	0.0658	0.0462	0.9440
SMOTE-ENN	0.9600	0.9697	0.9491	0.9559	0.0509	0.0303	0.9594
MSMOTE	0.9301	0.9445	0.9156	0.9182	0.0844	0.0555	0.9301
ISMOTE	0.9010	0.9117	0.8903	0.8929	0.1097	0.0883	0.9010
SL-SMOTE	0.8825	0.5376	0.9592	0.7477	0.0408	0.4624	0.7484
CE-SMOTE	0.9521	0.9679	0.9364	0.9383	0.0636	0.0321	0.9522
CURE-SMOTE	0.9820	0.9751	0.9889	**0.9889**	0.0111	0.0249	0.9820
SMOTE-D	**0.9821**	0.9753	0.9889	0.9888	0.0111	0.0247	**0.9821**
SDSMOTE	0.9359	0.9477	0.9241	0.9260	0.0759	0.0523	0.9359
G-SMOTE	0.9423	0.9558	0.9288	0.9307	0.0712	0.0442	0.9423
SMOTE-Cosine	0.9482	0.9491	0.9474	0.9475	0.0526	0.0509	0.9483
SMOTE-OUT	0.9518	0.9530	0.9506	0.9508	0.0494	0.0470	0.9518
Selected-SMOTE	0.9329	0.9454	0.9203	0.9224	0.0797	0.0546	0.9328
Gaussian-SMOTE	0.9820	0.9759	0.9882	0.9881	0.0118	0.0241	0.9820
SN-SMOTE	0.9320	0.9427	0.9212	0.9230	0.0788	0.0573	0.9320
LLE-SMOTE	0.9792	0.9802	0.9782	0.9782	0.0218	0.0198	0.9792
Cluster-SMOTE	0.9377	0.9479	0.9276	0.9291	0.0724	0.0521	0.9377
SMOTE-IPF	0.9374	0.9515	0.9233	0.9255	0.0767	0.0485	0.9374
AND-SMOTE	0.9673	0.9823	0.9524	0.9539	0.0476	0.0177	0.9673
LN-SMOTE	0.9652	0.2409	**0.9894**	0.4279	**0.0106**	0.7591	0.6151
SMOTE-PSO	0.9602	0.7761	0.9847	0.8713	0.0153	0.2239	0.8804
NT-SMOTE	0.9573	0.9683	0.9462	0.9475	0.0538	0.0317	0.9573
SMOTE-RS*B*	0.9650	0.2759	0.9883	0.4548	0.0117	0.7241	0.6321
Distance-SMOTE	0.9745	0.9776	0.9714	0.9716	0.0286	0.0224	0.9745
MWMOTE	0.9482	0.9550	0.9414	0.9423	0.0586	0.0450	0.9482
LVQ-SMOTE	0.9808	0.9758	0.9858	0.9856	0.0142	0.0242	0.9808
ADOMS	0.9717	0.9718	0.9715	0.9715	0.0285	0.0282	0.9717
ANS	0.9448	0.9551	0.9345	0.9360	0.0655	0.0449	0.9448
AHC	0.9604	0.6082	0.9838	0.7189	0.0162	0.3918	0.7960
SPY	0.9657	0.2958	0.9883	0.4728	0.0117	0.7042	0.6421

Table 8.25: The performance metrics values obtained by AdaBoost classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	0.9302	0.9744	0.8860	0.8954	0.1140	0.0256	0.9302
SMOTE	0.8976	0.9288	0.8664	0.8743	0.1336	0.0712	0.8976
BL-SMOTE	0.9359	0.9650	0.9068	0.9120	0.0932	0.0350	0.9359
SVM-SMOTE	0.9327	0.9157	0.9409	0.8818	0.0591	0.0843	0.9283
ADASYN	0.8972	0.9275	0.8667	0.8751	0.1333	0.0725	0.8971
SMOTE-Tomek	0.8933	0.9232	0.8633	0.8711	0.1367	0.0768	0.8932
SMOTE-ENN	0.9203	0.9513	0.8849	0.9040	0.1151	0.0487	0.9181
MSMOTE	0.9356	0.9646	0.9066	0.9117	0.0934	0.0354	0.9356
ISMOTE	0.8665	0.8780	0.8550	0.8583	0.1450	0.1220	0.8665
SL-SMOTE	0.9251	0.5931	0.9732	0.7629	0.0268	0.4069	0.7832
CE-SMOTE	0.9792	0.1036	0.9973	0.4517	0.0027	0.8964	0.5504
CURE-SMOTE	0.9772	0.9763	0.9782	0.9781	0.0218	0.0237	0.9772
SMOTE-D	**0.9896**	0.9812	0.9980	**0.9979**	0.0020	0.0188	**0.9896**
SDSMOTE	0.8935	0.9234	0.8635	0.8713	0.1365	0.0766	0.8935
G-SMOTE	0.9100	0.9408	0.8791	0.8862	0.1209	0.0592	0.9100
SMOTE-Cosine	0.9026	0.9309	0.8742	0.8811	0.1258	0.0691	0.9025
SMOTE-OUT	0.9042	0.9376	0.8707	0.8789	0.1293	0.0624	0.9042
Selected-SMOTE	0.8989	0.9315	0.8664	0.8747	0.1336	0.0685	0.8989
Gaussian-SMOTE	0.9873	0.9796	0.9950	0.9949	0.0050	0.0204	0.9873
SN-SMOTE	0.9042	0.9393	0.8690	0.8777	0.1310	0.0607	0.9042
LLE-SMOTE	0.9802	**0.9846**	0.9758	0.9760	0.0242	**0.0154**	0.9802
Cluster-SMOTE	0.9069	0.9333	0.8805	0.8865	0.1195	0.0667	0.9069
SMOTE-IPF	0.8982	0.9267	0.8697	0.8767	0.1303	0.0733	0.8982
AND-SMOTE	0.9583	0.9834	0.9331	0.9364	0.0669	0.0166	0.9583
LN-SMOTE	0.9400	0.9746	0.9054	0.9116	0.0946	0.0254	0.9400
SMOTE-PSO	0.9509	0.4742	0.9904	0.8068	0.0096	0.5258	0.7323
NT-SMOTE	0.9229	0.9477	0.8981	0.9030	0.1019	0.0523	0.9229
SMOTE-RS$B*$	0.9799	0.1325	**0.9982**	0.5812	**0.0018**	0.8675	0.5654
Distance-SMOTE	0.9478	0.9654	0.9303	0.9328	0.0697	0.0346	0.9479
MWMOTE	0.9088	0.9314	0.8862	0.8912	0.1138	0.0686	0.9088
LVQ-SMOTE	0.9854	0.9817	0.9891	0.9890	0.0109	0.0183	0.9854
ADOMS	0.9200	0.9313	0.9086	0.9108	0.0914	0.0687	0.9200
ANS	0.9157	0.9414	0.8901	0.8954	0.1099	0.0586	0.9158
AHC	0.9671	0.3235	0.9937	0.6935	0.0063	0.6765	0.6586
SPY	0.9793	0.0933	0.9977	0.4033	0.0023	0.9067	0.5455

Table 8.26: The performance metrics values obtained by AdaBoost classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9964**	0.9944	0.9984	**0.9984**	0.0016	0.0056	**0.9964**
SMOTE	0.9150	0.9220	0.9079	0.9092	0.0921	0.0780	0.9150
BL-SMOTE	0.9777	0.9855	0.9698	0.9703	0.0302	0.0145	0.9777
SMOTE-ENN	0.9235	0.9316	0.9151	0.9191	0.0849	0.0684	0.9234
SMOTE-Tomek	0.9157	0.9233	0.9080	0.9094	0.0920	0.0767	0.9157
SVM-SMOTE	0.9691	0.9728	0.9655	0.9658	0.0345	0.0272	0.9691
ADASYN	0.9119	0.9207	0.9031	0.9048	0.0969	0.0793	0.9119
MSMOTE	0.9726	0.9796	0.9657	0.9662	0.0343	0.0204	0.9727
ISMOTE	0.8903	0.8928	0.8878	0.8883	0.1122	0.1072	0.8903
SL-SMOTE	0.9822	0.3650	0.9961	0.6770	0.0039	0.6350	0.6805
CE-SMOTE	0.9935	0.0699	**0.9991**	0.3289	**0.0009**	0.9301	0.5345
CURE-SMOTE	0.9715	0.9792	0.9638	0.9643	0.0362	0.0208	0.9715
SMOTE-D	0.9944	0.9942	0.9946	0.9946	0.0054	0.0058	0.9944
SDSMOTE	0.9148	0.9223	0.9073	0.9086	0.0927	0.0777	0.9148
G-SMOTE	0.9231	0.9319	0.9142	0.9157	0.0858	0.0681	0.9230
SMOTE-Cosine	0.9201	0.9251	0.9152	0.9160	0.0848	0.0749	0.9202
SMOTE-OUT	0.9192	0.9255	0.9129	0.9139	0.0871	0.0745	0.9192
Selected-SMOTE	0.9141	0.9226	0.9056	0.9072	0.0944	0.0774	0.9141
Gaussian-SMOTE	0.9731	0.9866	0.9595	0.9606	0.0405	0.0134	0.9730
SN-SMOTE	0.9106	0.9200	0.9011	0.9030	0.0989	0.0800	0.9105
LLE-SMOTE	0.9873	0.9948	0.9797	0.9800	0.0203	0.0052	0.9872
Cluster-SMOTE	0.9126	0.9199	0.9052	0.9066	0.0948	0.0801	0.9125
SMOTE-IPF	0.9142	0.9241	0.9044	0.9063	0.0956	0.0759	0.9143
AND-SMOTE	0.9840	0.9918	0.9762	0.9766	0.0238	0.0082	0.9840
LN-SMOTE	0.9935	0.1022	0.9989	0.3454	0.0011	0.8978	0.5505
SMOTE-PSO	0.9812	0.2618	0.9959	0.5706	0.0041	0.7382	0.6289
NT-SMOTE	0.9272	0.9349	0.9195	0.9207	0.0805	0.0651	0.9272
Distance-SMOTE	0.9413	0.9511	0.9315	0.9329	0.0685	0.0489	0.9413
MWMOTE	0.9385	0.9444	0.9326	0.9334	0.0674	0.0556	0.9385
LVQ-SMOTE	0.9941	**0.9952**	0.9930	0.9930	0.0070	**0.0048**	0.9941
ADOMS	0.9162	0.9217	0.9108	0.9117	0.0892	0.0783	0.9163
ANS	0.9447	0.9506	0.9388	0.9396	0.0612	0.0494	0.9447
AHC	0.9882	0.2511	0.9971	0.5134	0.0029	0.7489	0.6241
SPY	0.9934	0.1098	**0.9991**	0.4361	**0.0009**	0.8902	0.5544

8.4.3.7 *Evaluate the Performance of DBBI-SMOTE compared to several balancing methods using XGBoost classifier.*

In this experiment, another high-performance and fast boosting-based classifier (i.e., XGBoost) is adopted to evaluate the feasibility of the new proposed balancing method compared to the other methods used in the literature. Thus, as what done in the previous experiments, DBBI-SMOTE and the other 34 balancing method utilized to balance four different datasets that belong to different companies' markets. Thus, regarding the Spanish companies dataset. As shown in Table 8.27, the conjunction of DBBI-SMOTE with XGBoost outperforms the other approaches in predicting the minority class instances, and shows the best overall performance in predicting the majority and the minority classes instances as stated in the evaluation metrics values. This approach obtains the highest *accuracy*, *recall* and *AUC* values, and the lowest *type II error*. On the other hand, SMOTE-RS$B*$ shows the optimum performance in predicting the majority class instances but at the expense of the minority class prediction; it obtains the best *specificity* and *type II error*, and also a high *Type II error* metric value.

Moreover, also in the Taiwanese companies dataset, DBBI-SMOTE shows the optimum performance in predicting the minority class instances and shows the highest overall performance in predicting the majority and the minority classes instances. As shown in Table 8.27 the conjunction of DBBI-SMOTE with XGBoost obtains the best *accuracy*, *recall* and *AUC*, and the lowest *type II error*. Nevertheless, CURE-SMOTE is the most appropriate balancing technique that leads the XGBoost to the minimal majority instances mispredicting. CURE-SMOTE + XGBoost obtains the highest *specificity* and *precision*, and the lowest *type I error* values.

In addition, referring to the Polish companies dataset, as shown in Table 8.29, the combination of DBBI-SMOTE with XGBoost yields the best overall performance in predicting the majority and the minority class instances. This approach obtains the best *accuracy* and *AUC* metrics values. On the other hand, SMOTE-ENN with XGBoost shows the optimum performance in predicting the minority class instances (bankrupt companies). This approach obtains the highest *recall* and the lowest *type II error* metrics values. Besides, CE-SMOTE leads the XGBoost to best behavior in the majority instances predicting and mispredicting. It obtains the best *specificity* and *type I error* values.

Finally, regarding the US companies dataset, as stated in Table 8.30, DBBI-SMOTE this time also shows the optimum overall performance in predicting the minority and majority class instances. The combination of DBBI-SMOTE and XGBoost obtains the highest *accuracy*, *precision* and *AUC* values. However, talking about each class prediction specifically, the combination of SMOTE-ENN with XGBoost shows the best performance in predicting the minority class instances according to the *recall* and *type II error* vales. On the other hand, CE-SMOTE is the most appropriate balancing method to preprocess the US companies' data to train the XGBoost in terms of determining the majority class instances. This approach obtains the best *specificity* and *type I error* values.

Accordingly, as a firm conclusion of this experiment, DBBI-SMOTE was the most appropriate balancing method for the four datasets, which leads the XGBoost to the best overall performance in determining the majority and the minority instances at the same time.

Table 8.27: The performance metrics values obtained by XGBoost classifier applied to the Spanish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9962**	**0.9997**	0.9961	0.9961	0.0039	**0.0003**	**0.9961**
SMOTE	0.9950	0.9989	0.9911	0.9912	0.0089	0.0011	0.9950
BL-SMOTE	0.9948	0.9968	0.9929	0.9929	0.0071	0.0032	0.9949
SVM-SMOTE	0.9931	0.9955	0.9918	0.9855	0.0082	0.0045	0.9937
ADASYN	0.9941	0.9982	0.9900	0.9901	0.0100	0.0018	0.9941
SMOTE-Tomek	0.9944	0.9993	0.9895	0.9897	0.0105	0.0007	0.9944
SMOTE-ENN	0.9955	0.9996	0.9912	0.9918	0.0088	0.0004	0.9954
MSMOTE	0.9925	0.9921	0.9928	0.9929	0.0072	0.0079	0.9925
ISMOTE	0.9948	0.9972	0.9923	0.9924	0.0077	0.0028	0.9948
SL-SMOTE	0.9805	0.7716	0.9943	0.9020	0.0057	0.2284	0.8829
CE-SMOTE	0.9921	0.9871	0.9971	**0.9971**	0.0029	0.0129	0.9921
CURE-SMOTE	0.9925	0.9893	0.9957	0.9957	0.0043	0.0107	0.9925
SMOTE-D	0.9911	0.9854	0.9968	0.9968	0.0032	0.0146	0.9911
SDSMOTE	0.9957	0.9982	0.9932	0.9933	0.0068	0.0018	0.9957
G-SMOTE	0.9932	0.9964	0.9900	0.9901	0.0100	0.0036	0.9932
SMOTE-Cosine	0.9945	0.9961	0.9929	0.9929	0.0071	0.0039	0.9945
SMOTE-OUT	0.9945	0.9957	0.9932	0.9933	0.0068	0.0043	0.9945
Selected-SMOTE	0.9943	0.9989	0.9896	0.9897	0.0104	0.0011	0.9943
Gaussian-SMOTE	0.9914	0.9868	0.9961	0.9961	0.0039	0.0132	0.9914
SN-SMOTE	0.9939	0.9964	0.9914	0.9915	0.0086	0.0036	0.9939
LLE-SMOTE	0.9916	0.9886	0.9946	0.9946	0.0054	0.0114	0.9916
Cluster-SMOTE	0.9952	0.9968	0.9936	0.9936	0.0064	0.0032	0.9952
SMOTE-IPF	0.9941	0.9986	0.9896	0.9898	0.0104	0.0014	0.9941
AND-SMOTE	0.9923	0.9918	0.9928	0.9929	0.0072	0.0082	0.9923
LN-SMOTE	0.9930	0.9954	0.9907	0.9908	0.0093	0.0046	0.9930
SMOTE-PSO	0.9911	0.9270	0.9968	0.9646	0.0032	0.0730	0.9619
NT-SMOTE	0.9930	0.9921	0.9939	0.9939	0.0061	0.0079	0.9930
SMOTE-RS$B*$	0.9839	0.4357	**0.9975**	0.8667	**0.0025**	0.5643	0.7166
Distance-SMOTE	0.9927	0.9896	0.9957	0.9957	0.0043	0.0104	0.9927
MWMOTE	0.9939	0.9982	0.9896	0.9898	0.0104	0.0018	0.9939
LVQ-SMOTE	0.9921	0.9903	0.9939	0.9939	0.0061	0.0097	0.9921
ADOMS	0.9932	0.9939	0.9925	0.9925	0.0075	0.0061	0.9932
ANS	0.9936	0.9964	0.9907	0.9908	0.0093	0.0036	0.9935
AHC	0.9839	0.7141	0.9957	0.8901	0.0043	0.2859	0.8549
SPY	0.9836	0.4167	0.9971	0.8155	0.0029	0.5833	0.7069

Table 8.28: The performance metrics values obtained by XGBoost classifier applied to the Taiwanese companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9925**	**0.9993**	0.9846	0.9868	0.0154	**0.0007**	**0.9921**
SMOTE	0.9870	0.9979	0.9762	0.9768	0.0238	0.0021	0.9870
BL-SMOTE	0.9858	0.9927	0.9788	0.9791	0.0212	0.0073	0.9858
SVM-SMOTE	0.9800	0.9692	0.9838	0.9546	0.0162	0.0308	0.9765
ADASYN	0.9872	0.9974	0.9771	0.9774	0.0229	0.0026	0.9872
SMOTE-Tomek	0.9867	0.9974	0.9761	0.9766	0.0239	**0.0026**	0.9868
SMOTE-ENN	0.9923	0.9991	0.9846	0.9867	0.0154	0.0009	0.9919
MSMOTE	0.9817	0.9801	0.9833	0.9833	0.0167	0.0199	0.9817
ISMOTE	0.9779	0.9897	0.9660	0.9669	0.0340	0.0103	0.9778
SL-SMOTE	0.9565	0.8467	0.9809	0.9082	0.0191	0.1533	0.9138
CE-SMOTE	0.9857	0.9908	0.9806	0.9808	0.0194	0.0092	0.9857
CURE-SMOTE	0.9838	0.9741	**0.9935**	**0.9934**	**0.0065**	0.0259	0.9838
SMOTE-D	0.9842	0.9752	0.9932	0.9931	0.0068	0.0248	0.9842
SDSMOTE	0.9857	0.9971	0.9742	0.9748	0.0258	0.0029	0.9856
G-SMOTE	0.9863	0.9941	0.9785	0.9788	0.0215	0.0059	0.9863
SMOTE-Cosine	0.9864	0.9929	0.9800	0.9803	0.0200	0.0071	0.9865
SMOTE-OUT	0.9861	0.9929	0.9794	0.9797	0.0206	0.0071	0.9862
Selected-SMOTE	0.9886	0.9973	0.9800	0.9803	0.0200	0.0027	0.9887
Gaussian-SMOTE	0.9836	0.9750	0.9923	0.9921	0.0077	0.0250	0.9836
SN-SMOTE	0.9870	0.9964	0.9776	0.9780	0.0224	0.0036	0.9870
LLE-SMOTE	0.9845	0.9768	0.9921	0.9920	0.0079	0.0232	0.9845
Cluster-SMOTE	0.9867	0.9961	0.9774	0.9778	0.0226	0.0039	0.9868
SMOTE-IPF	0.9860	0.9973	0.9747	0.9753	0.0253	0.0027	0.9860
AND-SMOTE	0.9847	0.9865	0.9829	0.9829	0.0171	0.0135	0.9847
LN-SMOTE	0.9695	0.2682	0.9929	0.5717	0.0071	0.7318	0.6305
SMOTE-PSO	0.9690	0.8057	0.9908	0.9216	0.0092	0.1943	0.8982
NT-SMOTE	0.9842	0.9868	0.9817	0.9818	0.0183	0.0132	0.9843
SMOTE-RS$B*$	0.9679	0.2630	0.9918	0.5163	0.0082	0.7370	0.6274
Distance-SMOTE	0.9840	0.9806	0.9874	0.9873	0.0126	0.0194	0.9840
MWMOTE	0.9861	0.9942	0.9780	0.9784	0.0220	0.0058	0.9861
LVQ-SMOTE	0.9845	0.9764	0.9926	0.9925	0.0074	0.0236	0.9845
ADOMS	0.9848	0.9836	0.9859	0.9859	0.0141	0.0164	0.9848
ANS	0.9845	0.9936	0.9755	0.9759	0.0245	0.0064	0.9846
AHC	0.9687	0.6196	0.9920	0.8382	0.0080	0.3804	0.8058
SPY	0.9682	0.2286	0.9932	0.5461	0.0068	0.7714	0.6109

Table 8.29: The performance metrics values obtained by XGBoost classifier applied to the Polish companies' dataset after balancing using DBBI-SMOTE and the other 34 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9977**	0.9988	0.9965	0.9965	0.0035	0.0012	**0.9977**
SMOTE	0.9942	0.9996	0.9889	0.9890	0.0111	0.0004	0.9943
BL-SMOTE	0.9941	0.9956	0.9925	0.9926	0.0075	0.0044	0.9941
SVM-SMOTE	0.9932	0.9887	0.9954	0.9905	0.0046	0.0113	0.9920
ADASYN	0.9941	0.9994	0.9889	0.9891	0.0111	0.0006	0.9941
SMOTE-Tomek	0.9943	0.9993	0.9894	0.9895	0.0106	0.0007	0.9943
SMOTE-ENN	0.9936	**0.9998**	0.9864	0.9883	0.0136	**0.0002**	0.9931
MSMOTE	0.9896	0.9883	0.9909	0.9909	0.0091	0.0117	0.9896
ISMOTE	0.9884	0.9956	0.9812	0.9815	0.0188	0.0044	0.9884
SL-SMOTE	0.9770	0.8801	0.9910	0.9341	0.0090	0.1199	0.9355
CE-SMOTE	0.9830	0.2024	**0.9992**	0.8417	**0.0008**	0.7976	0.6008
CURE-SMOTE	0.9935	0.9898	0.9971	0.9971	0.0029	0.0102	0.9934
SMOTE-D	0.9916	0.9843	0.9990	**0.9990**	0.0010	0.0157	0.9916
SDSMOTE	0.9938	0.9995	0.9882	0.9883	0.0118	0.0005	0.9939
G-SMOTE	0.9966	0.9987	0.9946	0.9946	0.0054	0.0013	0.9967
SMOTE-Cosine	0.9938	0.9972	0.9904	0.9905	0.0096	0.0028	0.9938
SMOTE-OUT	0.9929	0.9969	0.9889	0.9890	0.0111	0.0031	0.9929
Selected-SMOTE	0.9941	0.9995	0.9887	0.9888	0.0113	0.0005	0.9941
Gaussian-SMOTE	0.9909	0.9833	0.9986	0.9986	0.0014	0.0167	0.9909
SN-SMOTE	0.9944	0.9994	0.9895	0.9896	0.0105	0.0006	0.9945
LLE-SMOTE	0.9916	0.9845	0.9988	0.9988	0.0012	0.0155	0.9917
Cluster-SMOTE	0.9963	0.9994	0.9933	0.9933	0.0067	0.0006	0.9963
SMOTE-IPF	0.9945	0.9994	0.9897	0.9898	0.0103	0.0006	0.9946
AND-SMOTE	0.9935	0.9910	0.9960	0.9960	0.0040	0.0090	0.9935
LN-SMOTE	0.9945	0.9935	0.9956	0.9956	0.0044	0.0065	0.9946
SMOTE-PSO	0.9851	0.8312	0.9979	0.9701	0.0021	0.1688	0.9145
NT-SMOTE	0.9949	0.9962	0.9936	0.9936	0.0064	0.0038	0.9949
SMOTE-RS*B*	0.9836	0.2656	0.9991	0.8774	0.0009	0.7344	0.6323
Distance-SMOTE	0.9947	0.9955	0.9940	0.9940	0.0060	0.0045	0.9948
MWMOTE	0.9940	0.9985	0.9895	0.9896	0.0105	0.0015	0.9940
LVQ-SMOTE	0.9913	0.9838	0.9989	0.9989	0.0011	0.0162	0.9913
ADOMS	0.9939	0.9968	0.9910	0.9911	0.0090	0.0032	0.9939
ANS	0.9932	0.9969	0.9895	0.9896	0.0105	0.0031	0.9932
AHC	0.9827	0.5974	0.9986	0.9455	0.0014	0.4026	0.7980
SPY	0.9832	0.2224	0.9990	0.8324	0.0010	0.7776	0.6107

Table 8.30: The performance metrics values obtained by XGBoost classifier applied to the US companies' dataset after balancing using DBBI-SMOTE and the other 33 balancing methods. The best values are highlighted in gray.

Technique	Accuracy	Recall	Specificity	Precision	Type I error	Type II error	AUC
DBBI-SMOTE	**0.9968**	0.9945	0.9992	**0.9992**	0.0008	0.0055	**0.9969**
SMOTE	0.9909	0.9979	0.9840	0.9842	0.0160	0.0021	0.9909
BL-SMOTE	0.9953	0.9967	0.9939	0.9940	0.0061	0.0033	0.9953
SMOTE-ENN	0.9927	0.9986	0.9865	0.9871	0.0135	0.0014	0.9926
SMOTE-Tomek	0.9910	0.9980	0.9839	0.9841	0.0161	0.0020	0.9909
SVM-SMOTE	0.9952	0.9976	0.9928	0.9928	0.0072	0.0024	0.9952
ADASYN	0.9910	0.9983	0.9837	0.9839	0.0163	0.0017	0.9910
MSMOTE	0.9953	0.9967	0.9939	0.9939	0.0061	0.0033	0.9953
ISMOTE	0.9845	0.9968	0.9722	0.9729	0.0278	0.0032	0.9845
SL-SMOTE	0.9933	0.7526	0.9987	0.9299	0.0013	0.2474	0.8757
CE-SMOTE	0.9938	0.0789	**0.9993**	0.4094	**0.0007**	0.9211	0.5391
CURE-SMOTE	0.9961	0.9960	0.9963	0.9963	0.0037	0.0040	0.9961
SMOTE-D	0.9966	0.9942	0.9990	0.9990	0.0010	0.0058	0.9966
SDSMOTE	0.9911	0.9981	0.9840	0.9843	0.0160	0.0019	0.9910
G-SMOTE	0.9936	0.9985	0.9887	0.9888	0.0113	0.0015	0.9936
SMOTE-Cosine	0.9921	0.9973	0.9869	0.9870	0.0131	0.0027	0.9921
SMOTE-OUT	0.9916	0.9975	0.9857	0.9858	0.0143	0.0025	0.9916
Selected-SMOTE	0.9908	0.9980	0.9836	0.9838	0.0164	0.0020	0.9908
Gaussian-SMOTE	0.9967	0.9968	0.9966	0.9966	0.0034	0.0032	0.9967
SN-SMOTE	0.9921	0.9985	0.9858	0.9860	0.0142	0.0015	0.9922
LLE-SMOTE	0.9958	0.9951	0.9965	0.9965	0.0035	0.0049	0.9958
Cluster-SMOTE	0.9923	**0.9987**	0.9860	0.9862	0.0140	**0.0013**	0.9924
SMOTE-IPF	0.9910	0.9982	0.9837	0.9839	0.0163	0.0018	0.9909
AND-SMOTE	0.9960	0.9965	0.9955	0.9955	0.0045	0.0035	0.9960
LN-SMOTE	0.9938	0.0950	0.9993	0.4493	0.0007	0.9050	0.5472
SMOTE-PSO	0.9920	0.6948	0.9981	0.8806	0.0019	0.3052	0.8464
NT-SMOTE	0.9936	0.9972	0.9901	0.9902	0.0099	0.0028	0.9936
Distance-SMOTE	0.9947	0.9978	0.9916	0.9916	0.0084	0.0022	0.9947
MWMOTE	0.9893	0.9933	0.9852	0.9854	0.0148	0.0067	0.9892
LVQ-SMOTE	0.9967	0.9947	0.9986	0.9986	0.0014	0.0053	0.9967
ADOMS	0.9919	0.9956	0.9882	0.9883	0.0118	0.0044	0.9919
ANS	0.9898	0.9942	0.9855	0.9856	0.0145	0.0058	0.9899
AHC	0.9916	0.4108	0.9986	0.7759	0.0014	0.5892	0.7047
SPY	0.9935	0.1151	0.9992	0.4700	0.0008	0.8849	0.5572

8.5 FINAL REMARKS

Along with this chapter, a solution to a common problem mostly comes with the real data, and dramatically affects the performance of the ML and DL classifiers and leads them to outcome unreliably results is discussed. The major common problem in real data is the inconsistent distribution of the majority and minority instances in the data space. Thus, a novel data balancing technique, named Distance-Based Border Instances SMOTE (DBBI-SMOTE), has been proposed to fill the gap and avoid some drawbacks in the existing data balancing methods in the literature. Moreover, the structure of DBBI-SMOTE is based mainly on SMOTE and Borderline SMOTE methods but with some modification of their procedures to guarantee avoiding the scenarios that cause faults in the SMOTE and Borderline SMOTE procedures. In addition, due to the scarcity of bankrupt companies in the real-world, four different companies' datasets belong to different markets adopted to evaluate the performance of DBBI-SMOTE. The main difference between the dataset is the data type (i.e., numerical and categorical) and the complexity level for each one of them. The complexity level increases gradually from the Spanish companies' dataset (lowest complexity) to the US companies' dataset (Very complex), which makes a great challenge for the proposed method to prove that it is convenient to solve the problem in every data circumstance. Nevertheless, Three different types of classifiers were used as a post-stage of using DBBI-SMOTE to discuss its efficiency in solving the data problem, namely: Standard Classifiers (i.e., DT and NB), Bagging-Based Ensemble Classifiers (i.e., KNN, SVM and RF), and Boosting-Based Ensemble Classifiers (i.e., AdaBoost and XGBoost). Finally, the performance of DBBI-SMOTE is discussed compared to other 34 SMOTE variants used to balance the same four datasets as a preprocessing stage of the mentioned seven classifiers applications.

Furthermore, during training, the classifiers using imbalanced data, sticking with *accuracy* metric is not the best choice to evaluate the performance of any classifier because the classifiers tend always to predict the majority class instances and ignore the minority ones, which obtains very high *accuracy* in the expense of the classification reliability. Thus, eight different evaluation metrics devoted in this chapter to make an accurate decision about each data balancing method and classifier performance, namely: *accuracy*, *recall*, *specificity*, *precision*, *type I error*, *type II error* and *AUC*.

Accordingly, DBBI-SMOTE proved its efficiency and feasibility to solve the data inconsistent distribution problem in the four companies' datasets compared to the other 34 balancing methods, and in all of the classifiers applications. It's worth noting that DBBI-SMOTE wasn't the superior balancing method in each application, but it presented a solid alternative to solve the data problem beating many resampling methods in the literature. On the other hand, SMOTE-DBBI proved its performance with respect to reliability, not like some other balancing methods, it doesn't lead any classifier to exaggerate predicting certain class instances at the expense of the other class.

9 CONCLUSIONS

CONTENTS

This doctoral thesis addressed gradually several approaches to improve the solution of one of the critical real-world problems (i.e., companies financial failure prediction) using machine learning and deep learning algorithms. Those approaches started with improving the performance of the classical classification methods using simple sampling approaches, avoiding the overfitting and the loss of important information as well. And ended with developing a novel advanced balancing technique outperforming many other techniques addressed in the literature. Thus, taking into account the main objectives mentioned in Chapter 1, this chapter presents the main conclusions that support this thesis. In addition, it presents the future work that maintains continuing the contributions in this research field.

9.1 FINAL CONCLUSIONS

This thesis has been focused on improving the performance of Machine Learning and Deep Learning algorithms in solving a critical real-world problem (i.e., Companies financial failure forecasting). Thus, in this doctoral thesis, new simple resampling approaches have been adopted to solve the financial data inconsistent distribution problem. The first resampling approach is based mainly on partitioning the dataset into several subsets and using the balancing techniques in order to solve the problem of the balancing, this technique presents unreliable outcomes. The second sampling approach is based on splitting the original dataset in training and test set in order to avoid the reliability problem on the expense of the metrics values consistency; the classifiers select an inappropriate behavior to solve the problem while the training set is extremely unbalanced. The final approach is stands on integrating both of procedure; partitioning the dataset to several equally subsets and split test set for each partition with using the balancing techniques to make the training dataset balanced. This integration has yielded the best results regarding the reliability and the consistency of the metrics values and prove itself as the most proper style to solve the problem. Furthermore, three different well-known classical classification algorithm devoted to evaluate the performance of the proposed resampling approaches. Accordingly, using the third simple sampling approach as a preprocessing stage of applying RF shows promising performance in predicting companies financial failure.

Then, comprehensive analysis of the impact of using several different balancing methods on the performance of classical classification algorithms in predicting companies' financial failure have done. The selected balancing methods to analyse are covering the existing types in the literature. This analysis aims to conclude the best balancing technique to solve the financial data inconsistency distribution. Accordingly, the impact of these different balancing techniques on the classifiers during predicting companies financial failure has been discussed as well. However, all oversampling methods used lead to enhance results regarding predicting the bankrupt companies (minority instances) compared with classifying the original dataset without using any oversampling method. Nevertheless, SMOTE-ENN method obtained superior outcomes with with respect to identifying the minority instances (bankrupt companies).

Subsequently, the performance of three popular classification algorithms, i.e. J48, k-NN, and MLP in solving the bankruptcy prediction problem has been improved by hybridising these classifiers with the reservoir state of DLR using cascading technique. In other words, the input point for the proposed three approaches is the reservoir state of DLR, and then the data pass to train the addressed classifiers. Accordingly, the efficiency of using DLR to improve the classifiers performance demonstrated empirically. The use of DLR reservoir state can lead to performance improvements in Forecasting Business Failure over the normal ensemble voting or other single classifier models including J48, k-NN, and standard MLP.

Afterwards, several Deep Learning algorithms have been adapted specifically to predict companies' financial failure by identifying a specific number of hidden layers and hyperparameter values for each one of them. This configuration optimization has maintained the best performance in predicting bankrupt and solvent companies as well. Thus, the Deep Learning algorithms have been used in this chapter are Deep Belief Network (DBN), Multi-Layer Perceptron with 6 layers (MLP-6L), and Long-Short Term Memory (LSTM). In addition, Three bagging-based ensemble classifiers (i.e., RF, SVM and KNN) and two boosting-based ensemble classifiers (i.e., AdaBoost and XGBoost) were used as benchmarks to evaluate the performance of the Deep Learning algorithms. In addition, difficult and very imbalanced problem is faced in this study, using three different datasets belonging to different markets. In order to cope with this problem, three types of advanced balancing techniques have been devoted, namely: oversampling , Hybrid (oversampling+undersampling) methods, and clustering-based balancing method. Thus, after extensive experiments were done, MLP-6L applied to the datasets generated using SMOTE-ENN balancing technique obtained the best results regarding predicting companies' financial failure and outperforming all of the ensemble classifiers.

Finally, a novel data balancing technique named as Distance Based Border Instances SMOTE (DBBI-SMOTE) has been designed and developed to solve the inconstant distribution of the financial data. Thus, the novel technique avoids some drawbacks in the existing data balancing methods in the literature. Moreover, the structure of DBBI-SMOTE is based mainly on SMOTE and Borderline SMOTE methods but with some modification of their procedures to guarantee avoiding the scenarios that causing faults in the SMOTE and Borderline SMOTE procedures. The performance of DBBI evaluated using four different companies datasets belonging to different markets. The main difference between the dataset is the data type (i.e., numerical and categorical) and the complexity level for each one of them. In addition, three different types of classifiers have been used as a post-stage of using DBBI-SMOTE to discuss its efficiency in solving the data problem, namely: Standard Classifiers (i.e., DT and NB), Bagging-Based Ensemble Classifiers (i.e., KNN, SVM and RF), and Boosting-Based Ensemble Classifiers (i.e., AdaBoost and XGBoost). Finally, the performance of DBBI-SMOTE is discussed comparing to other 34 SMOTE variants used to balance the same four datasets as a preprocessing stage of the mentioned seven classifiers applications. Thus, DBBI-SMOTE showed high efficiency and feasibility in solving the data inconsistent distribution problem in the considered companies' datasets compared to the other 34 balancing methods, and in all of the classifiers applications.

9.2 FUTURE WORK

Deep Learning algorithms show high performance in solving the problem of predicting companies financial failure. But their performance could be improved by fine-tuning their parameters values, for instance by means of meta-optimization [221] applying Evolutionary Algorithms [222].

Another approach can improve the performance of Deep Learning algorithms and obtain promising results is considering them as ensemble models. These ensemble models could be comprised of several ensembles of one Deep Learning algorithm, or of several different algorithms

In addition, with respect to the novel data balancing technique (DBBI-SMOTE), it shows promising performance in balancing the financial binary class datasets. Thus, as a future work, it could be applied with other types of algorithms, such as ordinal classifiers.

A PUBLICATIONS

CONTENTS

A.1 INTERNATIONAL JOURNALS WITH IMPACT FACTOR

[1] H. Aljawazneh, A. M. Mora, P. García-Sánchez and P. A. Castillo-Valdivieso, "Comparing the Performance of Deep Learning Methods to Predict Companies' Financial Failure", in IEEE Access, vol. 9, pp. 97010-97038, 2021, doi: 10.1109/ACCESS.2021.3093461.

[2] H. Aljawazneh, A. M. Mora and P. A. Castillo-Valdivieso, "Solving the Financial Data Inconsistent Distribution Problem using DBBI SMOTE: A novel data balancing technique", **Work in progress**.

[3] H. Aljawazneh, A. M. Mora and P. A. Castillo-Valdivieso, "Predicting companies' financial failure using Capsule Network Algorithm", **Work in progress**.

A.2 INTERNATIONAL CONFERENCES

[1] Jawazneh H, Mora A, Castillo P. Predicting the financial status of companies using data balancing and classification methods. In International Work-Conference on Time Series (ITISE 2017). Godel Impresiones Digitales SL, Granada, Spain 2017 Sep (pp. 661-673).

[2] Rodan A, Castillo PA, Faris H, Mora AM, Jawazneh H. Forecasting Business Failure in Highly Imbalanced Distribution based on Delay Line Reservoir. In 26th European Symposium on Artificial Neural Networks, (ESANN 2018).

[3] Alswiti W, Faris H, Aljawazneh H, Safi S, Castillo P, Mora A, Abukhurma R, Alsawalqah H. Empirical evaluation of advanced oversampling methods for improving bankruptcy prediction. InProceedings of the International Conference on Time Series and Forecasting (ITISE 2018) (pp. 1495-1506).

[4] Safi S, Jawazneh H, Mora AM, García-Sánchez P, Faris H, Castillo PA. Identifying Botnets by Analysing Twitter Traffic during the Super Bowl. In 12th International Conference on Evolutionary Computation Theory and Applications (ECTA 2020) (pp. 147-154).

A.3 NATIONAL CONFERENCES

[1] Mora Antonio M., Jawazneh Huthaifa , Castillo Pedro A. Predicción de Quiebra Financiera de Empresas Mediante Equilibrado de Datos y Clasificación. In JAI 2017: III Jornadas Andaluzas de Informática.

B

GRANTS AND SPECIAL ACKNOWLEDGEMENTS

This thesis has been supported in part by the Ministerio de Ciencia, Innovación y Universidades under Project RTI2018-102002-A-I00, in part by the Ministerio de Economía y Competitividad under Projects TIN2017-85727-C4-2-P and PID2020-115570GB-C22, in part by the Fondo Europeo de Desarrollo Regional (FEDER) and Junta de Andalucía under Project B-TIC-402-UGR18, and in part by the Junta de Andalucía under Project P18-RT-4830.

BIBLIOGRAPHY

[1] Nils J Nilsson. *Principles of artificial intelligence*. Springer Science & Business Media, 1982 (Cited on page 1).

[2] John McCarthy. “What is artificial intelligence?” In: (2007) (Cited on page 1).

[3] Ethem Alpaydin. *Machine learning: the new AI*. MIT press, 2016 (Cited on page 2).

[4] Alessandra Caggiano, Jianjing Zhang, Vittorio Alfieri, Fabrizia Caiazzo, Robert Gao, and Roberto Teti. “Machine learning-based image processing for on-line defect recognition in additive manufacturing”. In: *CIRP Annals* 68.1 (2019), pp. 451–454 (Cited on page 2).

[5] David Ireri, Eisa Belal, Cedric Okinda, Nelson Makange, and Changying Ji. “A computer vision system for defect discrimination and grading in tomatoes using machine learning and image processing”. In: *Artificial Intelligence in Agriculture* 2 (2019), pp. 28–37 (Cited on page 2).

[6] Solomon Addai. “Financial forecasting using machine learning”. PhD thesis. University of Cape Town, 2016 (Cited on page 2).

[7] Harry Surden and Mary-Anne Williams. “Technological opacity, predictability, and self-driving cars”. In: *Cardozo L. Rev.* 38 (2016), p. 121 (Cited on page 2).

[8] Tingmin Wu, Shigang Liu, Jun Zhang, and Yang Xiang. “Twitter spam detection based on deep learning”. In: *Proceedings of the australasian computer science week multiconference*. 2017, pp. 1–8 (Cited on page 2).

[9] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick. “Credit card fraud detection using Bayesian and neural networks”. In: *Proceedings of the 1st international*

naiso congress on neuro fuzzy technologies. 2002, pp. 261–270 (Cited on page 2).

[10] Aravind Ganapathiraju, Jonathan Hamaker, and Joseph Picone. "Hybrid SVM/HMM architectures for speech recognition." In: *INTERSPEECH*. Citeseer. 2000, pp. 504–507 (Cited on page 2).

[11] Hamed Alqahtani, Iqbal H Sarker, Asra Kalim, Syed Md Minhaz Hossain, Sheikh Ikhlaq, and Sohrab Hossain. "Cyber intrusion detection using machine learning classification techniques". In: *International Conference on Computing Science, Communication and Security*. Springer. 2020, pp. 121–131 (Cited on page 2).

[12] Charu C Aggarwal. "Data classification". In: *Data Mining*. Springer. 2015, pp. 285–344 (Cited on page 3).

[13] Chong Ho Yu. "Resampling methods: concepts, applications, and justification". In: *Practical Assessment, Research, and Evaluation* 8.1 (2002), p. 19 (Cited on page 3).

[14] S Sarojini Devi and Y Radhika. "A survey on machine learning and statistical techniques in bankruptcy prediction". In: *International Journal of Machine Learning and Computing* 8.2 (2018), pp. 133–139 (Cited on page 9).

[15] P Ravi Kumar and Vadlamani Ravi. "Bankruptcy prediction in banks and firms via statistical and intelligent techniques–A review". In: *European journal of operational research* 180.1 (2007), pp. 1–28 (Cited on page 9).

[16] Edward I Altman. "Financial ratios, discriminant analysis and the prediction of corporate bankruptcy". In: *The journal of finance* 23.4 (1968), pp. 589–609 (Cited on page 10).

[17] Daniel Martin. "Early warning of bank failure: A logit regression approach". In: *Journal of banking & finance* 1.3 (1977), pp. 249–276 (Cited on page 10).

[18] Rashid Bhutta and Angappan Regupathi. "Predicting corporate bankruptcy: lessons from the past". In: *Asian Journal of Multidisciplinary Studies* 8.1 (2020), pp. 13–21 (Cited on page 10).

[19] James A Ohlson. "Financial ratios and the probabilistic prediction of bankruptcy". In: *Journal of accounting research* (1980), pp. 109–131 (Cited on page 10).

[20] James Kolari, Dennis Glennon, Hwan Shin, and Michele Caputo. "Predicting large US commercial bank failures". In: *Journal of Economics and Business* 54.4 (2002), pp. 361–387 (Cited on page 10).

[21] Stewart Jones and David A Hensher. "Predicting firm financial distress: A mixed logit model". In: *The accounting review* 79.4 (2004), pp. 1011–1038 (Cited on page 10).

[22] Heather Montgomery, Wimboh Santoso, Dwityapoetra S Besar, and Tran Hanh. "Coordinated failure? A cross-country bank failure prediction Model". In: *A Cross-Country Bank Failure Prediction Model (July 1, 2005). Asian Development Bank Institute Discussion Paper* 32 (2005) (Cited on page 10).

[23] Serpil Canbas, Altan Cabuk, and Suleyman Bilgin Kilic. "Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case". In: *European Journal of Operational Research* 166.2 (2005), pp. 528–546 (Cited on page 10).

[24] Gleb Lanine and Rudi Vander Vennet. "Failure prediction in the Russian bank sector with logit and trait recognition models". In: *Expert Systems with Applications* 30.3 (2006), pp. 463–478 (Cited on page 10).

[25] Natalia Konstandina. *Probability of bank failure: the Russian case*. Tech. rep. EERC Research Network, Russia and CIS, 2006 (Cited on page 10).

[26] Daniel Berg. "Bankruptcy prediction by generalized additive models". In: *Applied Stochastic Models in Business and Industry* 23.2 (2007), pp. 129–143 (Cited on page 10).

[27] Joao Minussi, DGR Soopramanien, and DJ Worthington. "Statistical modelling to predict corporate default for Brazilian companies in the context of Basel II using a new set of financial ratios". In: (2007) (Cited on page 11).

[28] Huimin Zhao, Atish P Sinha, and Wei Ge. “Effects of feature construction on classification performance: An empirical study in bank failure prediction”. In: *Expert Systems with Applications* 36.2 (2009), pp. 2633–2644 (Cited on page 11).

[29] Amalendu Bhunia and Ruchira Sarkar. “A study of financial distress based on MDA”. In: *Journal of Management Research* 3.2 (2011), pp. 1–11 (Cited on page 11).

[30] Jacek Brozyna, Grzegorz Mentel, and Tomasz Pisula. “Statistical methods of the bankruptcy prediction in the logistics sector in Poland and Slovakia”. In: *Transformations in Business & Economics* 15.1 (2016), pp. 80–96 (Cited on page 11).

[31] Jarmila Horváthová and Martina Mokrišová. “Comparison of the results of a data envelopment analysis model and logit model in assessing business financial health”. In: *Information* 11.3 (2020), p. 160 (Cited on page 11).

[32] Nicoleta Bărbuță-Mișu and Mara Madaleno. “Assessment of bankruptcy risk of large companies: European countries evolution analysis”. In: *Journal of Risk and Financial Management* 13.3 (2020), p. 58 (Cited on page 11).

[33] Jinwoo Baek and Sungzoon Cho. “Bankruptcy prediction for credit risk using an auto-associative neural network in Korean firms”. In: *2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings.* IEEE. 2003, pp. 25–29 (Cited on page 12).

[34] Xiao-Feng Hui and Jie Sun. “An application of support vector machine to companies’ financial distress prediction”. In: *International Conference on Modeling Decisions for Artificial Intelligence*. Springer. 2006, pp. 274–282 (Cited on page 12).

[35] Indranil Bose and Raktim Pal. “Predicting the survival or failure of click-and-mortar corporations: A knowledge discovery approach”. In: *European Journal of Operational Research* 174.2 (2006), pp. 959–982 (Cited on page 12).

[36] Hui Li and Jie Sun. “Predicting business failure using support vector machines with straightforward wrapper: A re-sampling study”. In: *Expert Systems with Applications* 38.10 (2011), pp. 12747–12756 (Cited on page 12).

[37] Junyoung Heo and Jin Yong Yang. “AdaBoost based bankruptcy forecasting of Korean construction companies”. In: *Applied soft computing* 24 (2014), pp. 494–499 (Cited on page 12).

[38] Deron Liang, Chia-Chi Lu, Chih-Fong Tsai, and Guan-An Shih. “Financial ratios and corporate governance indicators in bankruptcy prediction: A comprehensive study”. In: *European Journal of Operational Research* 252.2 (2016), pp. 561–572 (Cited on pages 12, 49, 89, 130).

[39] Maciej Zięba, Sebastian K Tomczak, and Jakub M Tomczak. “Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction”. In: *Expert Systems with Applications* 58 (2016), pp. 93–101 (Cited on pages 12, 86, 89, 130, 131, 132).

[40] Michal Karas and Maria Reznakova. “Predicting the bankruptcy of construction companies: a CART-based model”. In: *Engineering Economics* 28.2 (2017), pp. 145–154 (Cited on page 13).

[41] Tuong Le, Mi Young Lee, Jun Ryeol Park, and Sung Wook Baik. “Oversampling techniques for bankruptcy prediction: novel features from a transaction dataset”. In: *Symmetry* 10.4 (2018), p. 79 (Cited on pages 13, 63, 86, 99).

[42] Yi Qu, Pei Quan, Minglong Lei, and Yong Shi. “Review of bankruptcy prediction using machine learning and deep learning techniques”. In: *Procedia Computer Science* 162 (2019), pp. 895–899 (Cited on page 13).

[43] Shu Hao Yeh, Chuan Ju Wang, and Ming Feng Tsai. “Deep belief networks for predicting corporate defaults”. In: *2015 24th Wireless and Optical Communication Conference (WOCC)*. IEEE. 2015, pp. 159–163 (Cited on pages 13, 85, 125).

[44] Tadaaki Hosaka. “Bankruptcy prediction using imaged financial ratios and convolutional neural networks”. In: *Expert systems with applications* 117 (2019), pp. 287–299 (Cited on pages 13, 85).

[45] Joanna Wyrobek et al. “Predicting bankruptcy at polish companies: A comparison of selected machine learning and deep learning algorithms”. In: *Zeszyty Naukowe Uniwersytetu Ekonomicznego w Krakowie* 978.6 (2018), pp. 41–60 (Cited on page 14).

[46] Youjin Jang, In-Bae Jeong, Yong K Cho, and Yonghan Ahn. “Predicting business failure of construction contractors using long short-term memory recurrent neural network”. In: *Journal of Construction Engineering and Management* 145.11 (2019), p. 04019067 (Cited on pages 14, 86).

[47] Youjin Jang, Inbae Jeong, and Yong Cho. “Business Failure Prediction of Construction Contractors Using a LSTM RNN with Accounting, Construction Market, and Macroeconomic Variables”. In: *Journal of Management in Engineering* 36 (November 2019). DOI: `10.1061/(ASCE)ME.1943-5479.0000733` (Cited on pages 14, 86).

[48] Marek Vochozka, Jaromir Vrbka, and Petr Suler. “Bankruptcy or success? the effective prediction of a company’s financial development using LSTM”. In: *Sustainability* 12.18 (2020), p. 7529 (Cited on page 14).

[49] PPM Pompe and AJ Feelders. “Using machine learning, neural networks, and statistics to predict corporate bankruptcy”. In: *Computer-Aided Civil and Infrastructure Engineering* 12.4 (1997), pp. 267–276 (Cited on page 14).

[50] Jae H Min and Young-Chan Lee. “Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters”. In: *Expert systems with applications* 28.4 (2005), pp. 603–614 (Cited on pages 14, 76).

[51] Xiaoyan Xu and Yu Wang. “Financial failure prediction using efficiency as a predictor”. In: *Expert Systems with Applications* 36.1 (2009), pp. 366–373 (Cited on page 14).

[52] Tzong-Huei Lin. "A cross model study of corporate financial distress prediction in Taiwan: Multiple discriminant analysis, logit, probit and neural networks models". In: *Neurocomputing* 72.16-18 (2009), pp. 3507–3516 (Cited on page 15).

[53] Ali Mansouri, Arezoo Nazari, and Morteza Ramazani. "A comparison of artificial neural network model and logistics regression in prediction of companies' bankruptcy (A case study of Tehran stock exchange)". In: *International Journal of Advanced Computer Research* 6.24 (2016) (Cited on page 15).

[54] Sheikh Rabiul Islam, William Eberle, Sheikh K Ghafoor, Sid C Bundy, Douglas A Talbert, and Ambareen Siraj. "Investigating bankruptcy prediction models in the presence of extreme class imbalance and multiple stages of economy". In: *arXiv preprint arXiv:1911.09858* (2019) (Cited on pages 15, 64).

[55] Leila Bateni and Farshid Asghari. "Bankruptcy prediction using logit and genetic algorithm models: A comparative analysis". In: *Computational Economics* 55.1 (2020), pp. 335–348 (Cited on page 15).

[56] Kun Chang Lee, Ingoo Han, and Youngsig Kwon. "Hybrid neural network models for bankruptcy predictions". In: *Decision Support Systems* 18.1 (1996), pp. 63–72 (Cited on pages 16, 77).

[57] Sung-Hwan Min, Jumin Lee, and Ingoo Han. "Hybrid genetic algorithms and support vector machines for bankruptcy prediction". In: *Expert systems with applications* 31.3 (2006), pp. 652–660 (Cited on page 16).

[58] Chih-Hung Wu, Gwo-Hshiung Tzeng, Yeong-Jia Goo, and Wen-Chang Fang. "A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy". In: *Expert systems with applications* 32.2 (2007), pp. 397–408 (Cited on page 16).

[59] Yi-Chung Hu. "Incorporating a non-additive decision making method into multi-layer neural networks and its

application to financial distress analysis”. In: *Knowledge-Based Systems* 21.5 (2008), pp. 383–390 (Cited on page 16).

[60] Jie Sun and Hui Li. “Listed companies’ financial distress prediction based on weighted majority voting combination of multiple classifiers”. In: *Expert Systems with Applications* 35.3 (2008), pp. 818–827 (Cited on page 16).

[61] D Karthik Chandra, Vadlamani Ravi, and Indranil Bose. “Failure prediction of dotcom companies using hybrid intelligent techniques”. In: *Expert Systems with applications* 36.3 (2009), pp. 4830–4837 (Cited on page 16).

[62] Nazeeh Ghatasheh et al. “Cost-sensitive ensemble methods for bankruptcy prediction in a highly imbalanced data distribution: a real case from the Spanish market”. In: *Progress in Artificial Intelligence* 9.4 (2020), pp. 361–375 (Cited on pages 17, 129, 130).

[63] Hossam Faris et al. “Improving financial bankruptcy prediction in a highly imbalanced class distribution using oversampling and ensemble learning: a case from the Spanish market”. In: *Progress in Artificial Intelligence* 9.1 (2020), pp. 31–53 (Cited on pages 17, 117, 129, 130).

[64] Oded Maimon and Lior Rokach. “Data mining and knowledge discovery handbook”. In: (2005) (Cited on pages 20, 22, 24).

[65] Sotiris B Kotsiantis. “Decision trees: a recent overview”. In: *Artificial Intelligence Review* 39.4 (2013), pp. 261–283 (Cited on pages 20, 21, 22).

[66] Suryakanthi Tangirala. “Evaluating the impact of GINI index and information gain on classification using decision tree classifier algorithm”. In: *International Journal of Advanced Computer Science and Applications* 11.2 (2020), pp. 612–619 (Cited on pages 22, 23, 24).

[67] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014 (Cited on page 24).

[68] J. Ross Quinlan. “Induction of decision trees”. In: *Machine learning* 1.1 (1986), pp. 81–106 (Cited on page 24).

[69] Sonia Singh and Priyanka Gupta. “Comparative study ID3, cart and C4. 5 decision tree algorithm: a survey”. In: *International Journal of Advanced Information Science and Technology (IJAIST)* 27.27 (2014), pp. 97–103 (Cited on page 24).

[70] Tina R. Patil and S. Sherekar. “Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification”. In: *International Journal Of Computer Science And Applications* 6.2 (2013) (Cited on page 25).

[71] Thales Sehn Korting. “C4. 5 algorithm and multivariate decision trees”. In: *Image Processing Division, National Institute for Space Research–INPE Sao Jose dos Campos–SP, Brazil* (2006), p. 22 (Cited on page 25).

[72] Kevin P Murphy et al. “Naive bayes classifiers”. In: *University of British Columbia* 18.60 (2006), pp. 1–8 (Cited on page 26).

[73] Xindong Wu et al. “Top 10 algorithms in data mining”. In: *Knowledge and information systems* 14.1 (2008), pp. 1–37 (Cited on page 26).

[74] Harry Zhang. “The optimality of naive Bayes”. In: *AA* 1.2 (2004), p. 3 (Cited on pages 26, 27).

[75] Leo Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32 (Cited on pages 27, 87).

[76] Jianguo Chen et al. “A parallel random forest algorithm for big data in a Spark cloud computing environment”. In: *IEEE Transactions on Parallel and Distributed Systems* 28.4 (2017), pp. 919–933 (Cited on page 28).

[77] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. In: *Machine learning* 20.3 (1995), pp. 273–297 (Cited on pages 29, 87).

[78] Abhisek Ukil. “Support vector machine”. In: *Intelligent Systems and Signal Processing in Power Engineering*. Springer, 2007, pp. 161–226 (Cited on page 29).

[79] Jie Yan. “Ensemble svm regression based multi-view face detection system”. In: *2007 IEEE Workshop on Machine Learning for Signal Processing*. IEEE. 2007, pp. 163–169 (Cited on pages 29, 30, 31).

[80] Chi Xie, Changqing Luo, and Xiang Yu. “Financial distress prediction based on SVM and MDA methods: the case of Chinese listed companies”. In: *Quality & Quantity* 45.3 (2011), pp. 671–686 (Cited on page 29).

[81] Arti Patle and Deepak Singh Chouhan. “SVM kernel functions for classification”. In: *2013 International Conference on Advances in Technology and Engineering (ICATE)*. IEEE. 2013, pp. 1–9 (Cited on pages 30, 97).

[82] Leo Breiman. “Bagging predictors”. In: *Machine learning* 24.2 (1996), pp. 123–140 (Cited on pages 32, 87).

[83] Thomas Cover and Peter Hart. “Nearest neighbor pattern classification”. In: *IEEE transactions on information theory* 13.1 (1967), pp. 21–27 (Cited on pages 32, 87).

[84] Haneen Arafat Abu Alfeilat et al. “Effects of distance measure choice on k-nearest neighbor classifier performance: a review”. In: *Big data* 7.4 (2019), pp. 221–248 (Cited on page 35).

[85] Kittipong Chomboon, Pasapitch Chujai, Pongsakorn Teerarassamee, Kittisak Kerdprasop, and Nittaya Kerdprasop. “An empirical study of distance metrics for k-nearest neighbor algorithm”. In: *Proceedings of the 3rd international conference on industrial application engineering*. 2015, pp. 280–285 (Cited on page 35).

[86] Archana Singh, Avantika Yadav, and Ajay Rana. “K-means with Three different Distance Metrics”. In: *International Journal of Computer Applications* 67.10 (2013) (Cited on pages 35, 98).

[87] Yoav Freund and Robert E. Schapire. “Experiments with a New Boosting Algorithm”. In: *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*. ICML’96. Bari, Italy: Morgan Kaufmann Publishers Inc., 1996, pp. 148–156. ISBN: 1558604197 (Cited on pages 35, 87).

[88] Myoung-Jong Kim and Dae-Ki Kang. “Ensemble with neural networks for bankruptcy prediction”. In: *Expert systems with applications* 37.4 (2010), pp. 3373–3379 (Cited on page 36).

[89] Tu Chengsheng, Liu Huacheng, and Xu Bing. "AdaBoost typical Algorithm and its application research". In: *MATEC Web of Conferences*. Vol. 139. EDP Sciences. 2017, p. 00222 (Cited on page 37).

[90] Bernhard Schölkopf, Zhiyuan Luo, and Vladimir Vovk. *Empirical inference: Festschrift in honor of Vladimir N. Vapnik*. Springer Science & Business Media, 2013 (Cited on page 37).

[91] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794 (Cited on pages 38, 87).

[92] Sharmeen Binti Syazwan Lai, Nur Huda Nabihan Binti Md Shahri, Mazni Binti Mohamad, Hezlin Aryani Binti Abdul Rahman, and Adzhar Bin Rambli. "Comparing the Performance of AdaBoost, XGBoost, and Logistic Regression for Imbalanced Data". In: *Mathematics and Statistics* 9.3 (2021), pp. 379–385. DOI: `10.13189/ms.2021.090320` (Cited on pages 38, 39).

[93] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets". In: *Neural computation* 18.7 (2006), pp. 1527–1554 (Cited on pages 40, 86).

[94] Nicolas Le Roux and Yoshua Bengio. "Deep belief networks are compact universal approximators". In: *Neural computation* 22.8 (2010), pp. 2192–2207 (Cited on page 41).

[95] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780 (Cited on pages 42, 86).

[96] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. "Stock price prediction via discovering multi-frequency trading patterns". In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017, pp. 2141–2149 (Cited on page 42).

[97] Ahmad Ahmadpour Kasgari, Mehdi Divsalar, Mohamad Reza Javid, and Seyyed Javad Ebrahimian. “Prediction of bankruptcy Iranian corporations through artificial neural network and Probit-based analyses”. In: *Neural Computing and Applications* 23.3-4 (2013), pp. 927–936 (Cited on pages 43, 44, 45, 86).

[98] William Grant Hatcher and Wei Yu. “A survey of deep learning: Platforms, applications and emerging research trends”. In: *IEEE Access* 6 (2018), pp. 24411–24432 (Cited on pages 44, 94).

[99] Chih-Fong Tsai and Jhen-Wei Wu. “Using neural network ensembles for bankruptcy prediction and credit scoring”. In: *Expert systems with applications* 34.4 (2008), pp. 2639–2649 (Cited on pages 44, 45).

[100] H Jawazneh, A Mora, and P Castillo. “Predicting the financial status of companies using data balancing and classification methods”. In: *International Work-Conference on Time Series (ITISE 2017). Godel Impresiones Digitales SL, Granada, Spain*. 2017, pp. 661–673 (Cited on pages 47, 99, 129, 130).

[101] Félix J López Iturriaga and Iván Pastor Sanz. “Bankruptcy visualization and prediction using neural networks: A study of US commercial banks”. In: *Expert Systems with applications* 42.6 (2015), pp. 2857–2869 (Cited on pages 48, 49).

[102] Elena Fedorova, Evgenii Gilenko, and Sergey Dovzhenko. “Bankruptcy prediction for Russian companies: Application of combined classifiers”. In: *Expert systems with applications* 40.18 (2013), pp. 7285–7293 (Cited on pages 48, 49).

[103] Antonio Miguel Mora, Luis Javier Herrera, J Urquiza, Ignacio Rojas, and JJ Merelo. “Applying support vector machines and mutual information to book losses prediction”. In: *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE. 2010, pp. 1–7 (Cited on page 50).

[104] MAH Farquad and Indranil Bose. "Preprocessing unbalanced data using support vector machine". In: *Decision Support Systems* 53.1 (2012), pp. 226–233 (Cited on page 52).

[105] Alberto FernáNdez, Victoria LóPez, Mikel Galar, MarıéA José Del Jesus, and Francisco Herrera. "Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches". In: *Knowledge-based systems* 42 (2013), pp. 97–110 (Cited on page 52).

[106] Wedyan Alswiti et al. "Empirical evaluation of advanced oversampling methods for improving bankruptcy prediction". In: *Proceedings of the International Conference on Time Series and Forecasting (ITISE), Granada, 19-21 September, 2018*. September 2018, pp. 1495–1506 (Cited on pages 61, 129, 130).

[107] Edward I Altman and Edith Hotchkiss. *Corporate financial distress and bankruptcy: Predict and avoid bankruptcy, analyze and invest in distressed debt*. Vol. 289. John Wiley & Sons, 2010 (Cited on page 62).

[108] Joseph Janer and C Schneider. "Bankruptcy prediction and its advantages". In: *Empirical Evidence from SMEs in the French Hospitality Industry* (2011) (Cited on page 62).

[109] Hamad Alsawalqah, Hossam Faris, Ibrahim Aljarah, Loai Alnemer, and Nouh Alhindawi. "Hybrid SMOTE-Ensemble Approach for Software Defect Prediction". In: *Computer Science On-line Conference*. Springer. 2017, pp. 355–366 (Cited on page 63).

[110] Alaa S. AlAgha, Hossam Faris, Bassam H. Hammo, and Ala' M. Al-Zoubi. "Identifying b-thalassemia carriers using a data mining approach: The case of the Gaza Strip, Palestine". In: *Artificial Intelligence in Medicine* 88 (2018), pp. 70–83. ISSN: 0933-3657. DOI: `https://doi.org/10.1016/j.artmed.2018.04.009`. URL: `http://www.sciencedirect.com/science/article/pii/S0933365717303305` (Cited on page 63).

[111] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. "A study of the behavior of several methods for balancing machine learning training data". In: *ACM Sigkdd Explorations Newsletter* 6.1 (2004), pp. 20–29 (Cited on pages 63, 65, 66, 88, 140, 156, 157).

[112] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357 (Cited on pages 63, 65, 88, 140, 141, 174).

[113] Myoung-Jong Kim, Dae-Ki Kang, and Hong Bae Kim. "Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction". In: *Expert Systems with Applications* 42.3 (2015), pp. 1074–1082 (Cited on page 63).

[114] Ligang Zhou. "Performance of corporate bankruptcy prediction models on imbalanced dataset: The effect of sampling methods". In: *Knowledge-Based Systems* 41 (2013), pp. 16–25 (Cited on page 63).

[115] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning". In: *Advances in intelligent computing* (2005), pp. 878–887 (Cited on pages 66, 143, 144).

[116] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem". In: *Advances in knowledge discovery and data mining* (2009), pp. 475–482 (Cited on pages 67, 147).

[117] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE. 2008, pp. 1322–1328 (Cited on pages 67, 88, 140, 145).

[118] Sheng Tang and Si-Ping Chen. “The generation mechanism of synthetic minority class examples”. In: *Information Technology and Applications in Biomedicine, 2008. ITAB 2008. International Conference on*. IEEE. 2008, pp. 444–447 (Cited on pages 67, 155).

[119] Jerzy Stefanowski and Szymon Wilk. “Selective pre-processing of imbalanced data for improving classification performance”. In: *Lecture Notes in Computer Science* 5182 (2008), pp. 283–292 (Cited on page 68).

[120] Krystyna Napierała, Jerzy Stefanowski, and Szymon Wilk. “Learning from imbalanced data in presence of noisy and borderline examples”. In: *Rough sets and current trends in computing*. Springer. 2010, pp. 158–167 (Cited on page 69).

[121] Gilles Cohen, Mélanie Hilario, Hugo Sax, Stéphane Hugonnet, and Antoine Geissbuhler. “Learning from imbalanced data in surveillance of nosocomial infection”. In: *Artificial intelligence in medicine* 37.1 (2006), pp. 7–18 (Cited on pages 69, 164).

[122] Shoushan Li, Zhongqing Wang, Guodong Zhou, and Sophia Yat Mei Lee. “Semi-supervised learning for imbalanced sentiment classification”. In: *IJCAI proceedings-international joint conference on artificial intelligence*. Vol. 22. 3. 2011, p. 1826 (Cited on pages 69, 70).

[123] Ning Chen, Armando Vieira, and Joao Duarte. “Cost-sensitive LVQ for bankruptcy prediction: An empirical study”. In: *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*. IEEE. 2009, pp. 115–119 (Cited on page 70).

[124] Steven L Salzberg. “C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993”. In: *Machine Learning* 16.3 (1994), pp. 235–240 (Cited on page 70).

[125] Lawrence A Weiss and Vedran Capkun. “The impact of incorporating the cost of errors into bankruptcy prediction models”. In: (2005) (Cited on page 72).

[126] Ali Rodan, Pedro A Castillo, Hossam Faris, Antonio Miguel Mora, and Huthaifa Jawazneh. "Forecasting Business Failure in Highly Imbalanced Distribution based on Delay Line Reservoir." In: *ESANN*. 2018 (Cited on page 75).

[127] Byeong Seok Ahn, SS Cho, and CY Kim. "The integrated methodology of rough set theory and artificial neural network for business failure prediction". In: *Expert systems with applications* 18.2 (2000), pp. 65–74 (Cited on page 76).

[128] Linda M Salchenberger, E Mine Cinar, and Nicholas A Lash. "Neural networks: A new tool for predicting thrift failures". In: *Decision Sciences* 23.4 (1992), pp. 899–916 (Cited on page 76).

[129] Clarence NW Tan and Herlina Dihardjo. "A study of using artificial neural networks to develop an early warning predictor for credit union financial distress with comparison to the probit model". In: *Managerial Finance* (2001) (Cited on page 76).

[130] Kar Yan Tam and Melody Kiang. "Predicting bank failures: A neural network approach". In: *Applied Artificial Intelligence an International Journal* 4.4 (1990), pp. 265–282 (Cited on page 76).

[131] Tae Kyung Sung, Namsik Chang, and Gunhee Lee. "Dynamics of modeling in data mining: interpretive approach to bankruptcy prediction". In: *Journal of management information systems* 16.1 (1999), pp. 63–85 (Cited on page 76).

[132] J Efrim Boritz and Duane B Kennedy. "Effectiveness of neural network types for prediction of business failure". In: *Expert Systems with Applications* 9.4 (1995), pp. 503–512 (Cited on page 76).

[133] R Christopher Lacher, Pamela K Coats, Shanker C Sharma, and L Franklin Fant. "A neural network for classifying the financial health of a firm". In: *European Journal of Operational Research* 85.1 (1995), pp. 53–65 (Cited on page 76).

[134] Moshe Leshno and Yishay Spector. "Neural network prediction analysis: The bankruptcy case". In: *Neurocomputing* 10.2 (1996), pp. 125–147 (Cited on page 76).

[135] Shu Ling Lin. “A new two-stage hybrid approach of credit risk in banking industry”. In: *Expert Systems with Applications* 36.4 (2009), pp. 8333–8341 (Cited on page 77).

[136] Chih-Fong Tsai and Ming-Lun Chen. “Credit rating by hybrid machine learning techniques”. In: *Applied soft computing* 10.2 (2010), pp. 374–380 (Cited on page 77).

[137] Tian-Shyug Lee, Chih-Chou Chiu, Chi-Jie Lu, and I-Fei Chen. “Credit scoring using the hybrid neural discriminant technique”. In: *Expert Systems with applications* 23.3 (2002), pp. 245–254 (Cited on page 77).

[138] Nan-Chen Hsieh. “Hybrid mining approach in the design of credit scoring models”. In: *Expert Systems with Applications* 28.4 (2005), pp. 655–665 (Cited on page 77).

[139] Johan Huysmans, Bart Baesens, Jan Vanthienen, and Tony Van Gestel. “Failure prediction with self organizing maps”. In: *Expert Systems with Applications* 30.3 (2006), pp. 479–487 (Cited on page 77).

[140] Cheng-Lung Huang, Mu-Chen Chen, and Chieh-Jen Wang. “Credit scoring with a data mining approach based on support vector machines”. In: *Expert systems with applications* 33.4 (2007), pp. 847–856 (Cited on page 77).

[141] Tony Van Gestel, Bart Baesens, Johan AK Suykens, Dirk Van den Poel, Dirk-Emma Baestaens, and Marleen Willekens. “Bayesian kernel based classification for financial distress detection”. In: *European journal of operational research* 172.3 (2006), pp. 979–1003 (Cited on page 77).

[142] Athanasios Tsakonas, George Dounias, Michael Doumpos, and Constantin Zopounidis. “Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming”. In: *Expert Systems with Applications* 30.3 (2006), pp. 449–461 (Cited on page 77).

[143] Ali Rodan and Peter Tino. “Minimum complexity echo state network”. In: *IEEE transactions on neural networks* 22.1 (2010), pp. 131–144 (Cited on page 77).

[144] H Aljawazneh, AM Mora, Pablo Garcıéa-Sánchez, and PA Castillo-Valdivieso. “Comparing the Performance of Deep Learning Methods to Predict Companies’ Financial Failure”. In: *IEEE Access* 9 (2021), pp. 97010–97038 (Cited on page 84).

[145] Ajay Shrestha and Ausif Mahmood. “Review of deep learning algorithms and architectures”. In: *IEEE Access* 7 (2019), pp. 53040–53065 (Cited on page 85).

[146] Doohee Lee, Jingu Lee, Jingyu Ko, Jaeyeon Yoon, Kanghyun Ryu, and Yoonho Nam. “Deep learning in MR image processing”. In: *Investigative Magnetic Resonance Imaging* 23.2 (2019), pp. 81–99 (Cited on page 85).

[147] Treesukon Treebupachatsakul and Suvit Poomrittigul. “Bacteria classification using image processing and deep learning”. In: *2019 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*. IEEE. 2019, pp. 1–3 (Cited on page 85).

[148] Marco Leo, Antonino Furnari, Gerard G Medioni, Mohan Trivedi, and Giovanni M Farinella. “Deep learning for assistive computer vision”. In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018, pp. 3–14 (Cited on page 85).

[149] Kristian M Black, Hei Law, Ali Aldoukhi, Jia Deng, and Khurshid R Ghani. “Deep learning computer vision algorithm for detecting kidney stone composition”. In: (2020) (Cited on page 85).

[150] Abhimanyu Roy, Jingyi Sun, Robert Mahoney, Loreto Alonzi, Stephen Adams, and Peter Beling. “Deep learning detecting fraud in credit card transactions”. In: *2018 Systems and Information Engineering Design Symposium (SIEDS)*. IEEE. 2018, pp. 129–134 (Cited on page 85).

[151] Pradheepan Raghavan and Neamat El Gayar. “Fraud detection using machine learning and deep learning”. In: *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*. IEEE. 2019, pp. 334–339 (Cited on page 85).

[152] Ramesh Simhambhatla, Kevin Okiah, Shravan Kuchkula, and Robert Slater. "Self-driving cars: Evaluation of deep learning techniques for object detection in different driving conditions". In: *SMU Data Science Review* 2.1 (2019), p. 23 (Cited on page 85).

[153] Anselme Ndikumana, Nguyen H Tran, Ki Tae Kim, Choong Seon Hong, et al. "Deep learning based caching for self-driving cars in multi-access edge computing". In: *IEEE Transactions on Intelligent Transportation Systems* 22.5 (2020), pp. 2862–2877 (Cited on page 85).

[154] Li Deng and John Platt. "Ensemble deep learning for speech recognition". In: *Proc. Interspeech*. 2014 (Cited on page 85).

[155] Jing Huang and Brian Kingsbury. "Audio-visual deep learning for noise robust speech recognition". In: *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE. 2013, pp. 7596–7599 (Cited on page 85).

[156] Anwar Ul Haq, Adnan Zeb, Zhenfeng Lei, and Defu Zhang. "Forecasting daily stock trend using multi-filter feature selection and deep learning". In: *Expert Systems with Applications* 168 (2021), p. 114444 (Cited on page 85).

[157] Alexiei Dingli and Karl Sant Fournier. "Financial time series forecasting–a deep learning approach". In: *International Journal of Machine Learning and Computing* 7.5 (2017), pp. 118–122 (Cited on page 85).

[158] George E Dahl, Jack W Stokes, Li Deng, and Dong Yu. "Large-scale malware classification using random projections and neural networks". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2013, pp. 3422–3426 (Cited on page 85).

[159] Farhan Ullah et al. "Cyber security threats detection in internet of things using deep learning approach". In: *IEEE Access* 7 (2019), pp. 124379–124389 (Cited on page 85).

[160] Jodi L. Bellovary, Don E. Giacomino, and Michael D. Akers. "A Review of Bankruptcy Prediction Studies: 1930 to Present". In: *Journal of Financial Education* 33 (2007), pp. 1–42. ISSN: 00933961, 2332421X (Cited on page 85).

[161] Dang Lien Minh, Abolghasem Sadeghi-Niaraki, Huynh Duc Huy, Kyungbok Min, and Hyeonjoon Moon. "Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network". In: *Ieee Access* 6 (2018), pp. 55392–55404 (Cited on page 85).

[162] Zineb Lanbouri and Said Achchab. "A hybrid Deep belief network approach for Financial distress prediction". In: *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)*. IEEE. 2015, pp. 1–6 (Cited on page 85).

[163] David Veganzones and Eric Séverin. "An investigation of bankruptcy prediction in imbalanced datasets". In: *Decision Support Systems* 112 (2018), pp. 111–124 (Cited on page 85).

[164] Nurheri Cahyana, Siti Khomsah, and Agus Sasmito Aribowo. "Improving Imbalanced Dataset Classification Using Oversampling and Gradient Boosting". In: *2019 5th International Conference on Science in Information Technology (ICSITech)*. IEEE. 2019, pp. 217–222 (Cited on pages 88, 140, 143).

[165] Felix Last, Georgios Douzas, and Fernando Bacao. "Oversampling for imbalanced learning based on k-means and smote". In: *arXiv preprint arXiv:1711.00837* (2017) (Cited on page 88).

[166] Bei Zhou, Zongzhi Li, Shengrui Zhang, Xinfen Zhang, Xin Liu, and Qiannan Ma. "Analysis of Factors Affecting Hit-and-Run and Non-Hit-and-Run in Vehicle-Bicycle Crashes: A Non-Parametric Approach Incorporating Data Imbalance Treatment". In: *Sustainability* 11.5 (2019), p. 1327 (Cited on pages 88, 144).

[167] Qiong Gu, Li Zhu, and Zhihua Cai. "Evaluation measures of the classification performance of imbalanced data sets". In: *International symposium on intelligence computation and applications*. Springer. 2009, pp. 461–471 (Cited on page 92).

[168] David Powers. "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation". In: *J. Mach. Learn. Technol* 2 (January 2011), pp. 2229–3981. DOI: 10.9735/2229-3981 (Cited on pages 92, 93, 173, 174).

[169] Miodrag Stojanović et al. "Understanding sensitivity, specificity and predictive values". In: *Vojnosanitetski pregled* 71.11 (2014), pp. 1062–1065 (Cited on pages 93, 173).

[170] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830 (Cited on page 94).

[171] Tzu-Tsung Wong. "Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation". In: *Pattern Recognition* 48.9 (2015), pp. 2839–2846 (Cited on page 94).

[172] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958 (Cited on page 95).

[173] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. "Activation functions: Comparison of trends in practice and research for deep learning". In: *arXiv preprint arXiv:1811.03378* (2018) (Cited on page 96).

[174] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014) (Cited on page 96).

[175] Stephen M Omohundro. *Five balltree construction algorithms*. International Computer Science Institute Berkeley, 1989 (Cited on page 97).

[176] H Jabbar and Rafiqul Zaman Khan. "Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)". In: *Computer Science, Communication and Instrumentation Devices* (2015), pp. 163–172 (Cited on page 101).

[177] John Hearty. *Advanced Machine Learning with Python*. Packt Publishing Ltd, 2016 (Cited on page 101).

[178] Hongliang He, Wenyu Zhang, and Shuai Zhang. "A novel ensemble method for credit scoring: Adaption of different imbalance ratios". In: *Expert Systems with Applications* 98 (2018), pp. 105–117 (Cited on page 121).

[179] Mubashir Ahmad et al. "Deep belief network modeling for automatic liver segmentation". In: *IEEE Access* 7 (2019), pp. 20585–20595 (Cited on page 125).

[180] Dhanya Bibin, Madhu S Nair, and P Punitha. "Malaria parasite detection from peripheral blood smear images using deep belief networks". In: *IEEE Access* 5 (2017), pp. 9099–9108 (Cited on page 125).

[181] Ali Rodan, Pedro A. Castillo, Hossam Faris, Antonio Miguel Mora, and Huthaifa Jawazneh. "Forecasting Business Failure in Highly Imbalanced Distribution based on Delay Line Reservoir". In: *26th European Symposium on Artificial Neural Networks, ESANN 2018, Bruges, Belgium, April 25-27, 2018*. 2018. URL: `http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2018-105.pdf` (Cited on pages 129, 130).

[182] John Muschelli. "Roc and auc with a binary predictor: a potentially misleading metric". In: *Journal of Classification* (2019), pp. 1–13 (Cited on pages 131, 174).

[183] Lu Bao-Liang, Wang Xiao-Lin, Yang Yang, and Zhao Hai. "Learning from imbalanced data sets with a Min-Max modular support vector machine". In: *Frontiers of Electrical and Electronic Engineering in China* 6.1 (2011), pp. 56–71 (Cited on page 139).

[184] Ricardo Barandela, José Salvador Sánchez, Vicente Garcıa, and Edgar Rangel. "Strategies for learning in class imbalance problems". In: *Pattern Recognition* 36.3 (2003), pp. 849–851 (Cited on page 139).

[185] Shengguo Hu, Yanfeng Liang, Lintao Ma, and Ying He. "MSMOTE: Improving classification performance when training data is imbalanced". In: *2009 second international workshop on computer science and engineering*. Vol. 2. IEEE. 2009, pp. 13–17 (Cited on pages 140, 146, 174).

[186] Fajri Koto. "SMOTE-Out, SMOTE-Cosine, and Selected-SMOTE: An enhancement strategy to handle imbalance in data level". In: *2014 International Conference on Advanced Computer Science and Information System*. IEEE. 2014, pp. 280–284 (Cited on pages 140, 148, 149).

[187] Hansoo Lee, Jonggeun Kim, and Sungshin Kim. "Gaussian-based SMOTE algorithm for solving skewed class distributions". In: *International Journal of Fuzzy Logic and Intelligent Systems* 17.4 (2017), pp. 229–234 (Cited on pages 140, 149).

[188] Li Ma and Suohai Fan. "CURE-SMOTE algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests". In: *BMC bioinformatics* 18.1 (2017), pp. 1–18 (Cited on pages 140, 163, 174).

[189] Charles Elkan. "The foundations of cost-sensitive learning". In: *International joint conference on artificial intelligence*. Vol. 17. 1. Lawrence Erlbaum Associates Ltd. 2001, pp. 973–978 (Cited on page 140).

[190] Shadi Mahmoudi, Parham Moradi, Fardin Akhlaghian, and Rizan Moradi. "Diversity and separable metrics in over-sampling technique for imbalanced data classification". In: *2014 4th International Conference on Computer and Knowledge Engineering (ICCKE)*. IEEE. 2014, pp. 152–158 (Cited on page 140).

[191] Yanjie Dong and Xuehua Wang. "A new over-sampling approach: random-SMOTE for learning from imbalanced data sets". In: *International Conference on Knowledge Science, Engineering and Management*. Springer. 2011, pp. 343–352 (Cited on page 140).

[192] Katarzyna Borowska and Jarosław Stepaniuk. "Imbalanced data classification: A novel re-sampling approach combining versatile improved SMOTE and rough sets". In: *IFIP International Conference on Computer Information Systems and Industrial Management*. Springer. 2016, pp. 31–42 (Cited on page 140).

[193] Colin Bellinger, Nathalie Japkowicz, and Christopher Drummond. "Synthetic oversampling for advanced radioactive threat detection". In: *2015 IEEE 14th International*

Conference on Machine Learning and Applications (ICMLA). IEEE. 2015, pp. 948–953 (Cited on page 140).

[194] Kewen Li, Wenrong Zhang, Qinghua Lu, and Xianghua Fang. "An Improved SMOTE Imbalanced Data Classification Method Based on Support Degree". In: *2014 International Conference on Identification, Information and Knowledge in the Internet of Things*. 2014, pp. 34–38. DOI: 10.1109/IIKI.2014.14 (Cited on pages 140, 159, 174).

[195] Wacharasak Siriseriwan and Krung Sinapiromsaran. "Adaptive neighbor synthetic minority oversampling technique under 1NN outcast handling". In: *Songklanakarin J. Sci. Technol* 39.5 (2017), pp. 565–576 (Cited on pages 140, 161).

[196] Si Chen, Gongde Guo, and Lifei Chen. "A New Over-Sampling Method Based on Cluster Ensembles". In: *2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*. 2010, pp. 599–604. DOI: 10.1109/WAINA.2010.40 (Cited on pages 140, 162, 174).

[197] David A Cieslak, Nitesh V Chawla, and Aaron Striegel. "Combating imbalance in network intrusion datasets." In: *GrC*. Citeseer. 2006, pp. 732–737 (Cited on pages 140, 163).

[198] Rok Blagus and Lara Lusa. "SMOTE for high-dimensional class-imbalanced data". In: vol. 106. 14. 2013. DOI: https://doi.org/10.1186/1471-2105-14-106 (Cited on page 142).

[199] John Tabak. *Geometry: the language of space and form*. Infobase Publishing, 2014 (Cited on page 142).

[200] Vicente Garcıéa, José Salvador Sánchez, Raúl Martıén-Félez, and Ramón Alberto Mollineda. "Surrounding neighborhood-based SMOTE for learning from imbalanced data sets". In: *Progress in Artificial Intelligence* 1.4 (2012), pp. 347–362 (Cited on page 149).

[201] BB Chaudhuri. "A new definition of neighborhood of a point in multi-dimensional space". In: *Pattern Recognition Letters* 17.1 (1996), pp. 11–17 (Cited on page 150).

[202] Juanjuan Wang, Mantao Xu, Hui Wang, and Jiwu Zhang. "Classification of imbalanced data by using the SMOTE algorithm and locally linear embedding". In: *2006 8th international Conference on Signal Processing*. Vol. 3. IEEE. 2006 (Cited on page 150).

[203] Fredy Rodrıéguez Torres, Jesús A Carrasco-Ochoa, and José Fco Martıénez-Trinidad. "SMOTE-D a deterministic version of SMOTE". In: *Mexican Conference on Pattern Recognition*. Springer. 2016, pp. 177–188 (Cited on page 150).

[204] Jaesub Yun, Jihyun Ha, and Jong-Seok Lee. "Automatic determination of neighborhood size in SMOTE". In: *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*. 2016, pp. 1–8 (Cited on page 151).

[205] Tomasz Maciejewski and Jerzy Stefanowski. "Local neighbourhood extension of SMOTE for mining imbalanced data". In: *2011 IEEE symposium on computational intelligence and data mining (CIDM)*. IEEE. 2011, pp. 104–111 (Cited on page 152).

[206] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodriguez, Asdrúbal López, José Ruiz Castilla, and Adrian Trueba. "PSO-based method for SVM classification on skewed data sets". In: *Neurocomputing* 228 (2017), pp. 187–197 (Cited on page 152).

[207] Yu Hui Xu, Hui Li, Lu Ping Le, and Xiao Yun Tian. "Neighborhood triangular synthetic minority over-sampling technique for imbalanced prediction on small samples of chinese tourism and hospitality firms". In: *2014 Seventh International Joint Conference on Computational Sciences and Optimization*. IEEE. 2014, pp. 534–538 (Cited on page 152).

[208] Jorge De La Calleja and Olac Fuentes. "A Distance-Based Over-Sampling Method for Learning from Imbalanced Data Sets." In: *FLAIRS Conference*. 2007, pp. 634–635 (Cited on page 153).

[209] Sukarna Barua, Md Monirul Islam, Xin Yao, and Kazuyuki Murase. “MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning”. In: *IEEE Transactions on knowledge and data engineering* 26.2 (2012), pp. 405–425 (Cited on page 154).

[210] Munehiro Nakamura, Yusuke Kajiwara, Atsushi Otsuka, and Haruhiko Kimura. “Lvq-smote–learning vector quantization based synthetic minority over–sampling technique for biomedical data”. In: *BioData mining* 6.1 (2013), pp. 1–10 (Cited on page 155).

[211] Hu Li, Peng Zou, Xiang Wang, and Rongze Xia. “A new combination sampling method for imbalanced data”. In: *Proceedings of 2013 Chinese Intelligent Automation Conference*. Springer. 2013, pp. 547–554 (Cited on pages 157, 158).

[212] Jorma Laurikkala. “Improving identification of difficult small classes by balancing class distribution”. In: *Conference on Artificial Intelligence in Medicine in Europe*. Springer. 2001, pp. 63–66 (Cited on page 159).

[213] Tushar Sandhan and Jin Young Choi. “Handling imbalanced datasets by partially guided hybrid sampling for pattern recognition”. In: *2014 22nd International Conference on Pattern Recognition*. IEEE. 2014, pp. 1449–1453 (Cited on page 160).

[214] José A Sáez, Julián Luengo, Jerzy Stefanowski, and Francisco Herrera. “SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering”. In: *Information Sciences* 291 (2015), pp. 184–203 (Cited on pages 160, 174).

[215] Enislay Ramentol, Yailé Caballero, Rafael Bello, and Francisco Herrera. “SMOTE-RS B*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory”. In: *Knowledge and information systems* 33.2 (2012), pp. 245–265 (Cited on page 160).

[216] Zdzisław Pawlak. “Rough sets”. In: *International journal of computer & information sciences* 11.5 (1982), pp. 341–356 (Cited on page 161).

[217] Xuan Tho Dang, Dang Hung Tran, Osamu Hirose, and Kenji Satou. "SPY: A novel resampling method for improving classification performance in imbalanced data". In: *2015 Seventh International Conference on Knowledge and Systems Engineering (KSE)*. IEEE. 2015, pp. 280–285 (Cited on pages 161, 174).

[218] Salima Smiti and Makram Soui. "Bankruptcy prediction using deep learning approach based on borderline SMOTE". In: *Information Systems Frontiers* 22.5 (2020), pp. 1067–1083 (Cited on page 165).

[219] Qingjie Wang, Jingmin Xin, Jiayi Wu, and Nanning Zheng. "SVM classification of microaneurysms with imbalanced dataset based on borderline-SMOTE and data cleaning techniques". In: *Ninth international conference on machine vision (ICMV 2016)*. Vol. 10341. International Society for Optics and Photonics. 2017, 103411S (Cited on page 165).

[220] Pullagura Indira priyadarsini. "ABC-BSRF: Artificial Bee Colony and Borderline-SMOTE RF Algorithm for Intrusion Detection System on Data Imbalanced Problem". In: *Proceedings of International Conference on Computational Intelligence and Data Engineering: ICCIDE 2020*. Springer. 2021, pp. 15–29 (Cited on page 165).

[221] Mauro Birattari and Janusz Kacprzyk. *Tuning metaheuristics: a machine learning perspective*. Vol. 197. Springer, 2009 (Cited on page 224).

[222] Thomas Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996 (Cited on page 224).