Muhammed Şafak PİNAR

A Master's Thesis

AGU 2022

A NOVEL APPROACH BASED ON BAGGING AND BOOSTING FOR IMBALANCED CLASSIFICATION PROBLEMS

A THESIS
SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE OF
ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Muhammed Şafak PİNAR
August 2022

A NOVEL APPROACH BASED ON BAGGING AND BOOSTING FOR IMBALANCED CLASSIFICATION PROBLEMS

A THESIS
SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE OF
ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Muhammed Şafak PİNAR
August 2022

SCIENTIFIC ETHICS COMPLIANCE

I hereby declare that all information in this document has been obtained in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name-Surname: Muhammed Şafak PİNAR

Signature :

REGULATORY COMPLIANCE

M.Sc. thesis titled “A Novel Approach Based on Bagging and Boosting for Imbalanced Classification Problems” has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By
Muhammed Şafak PİNAR
Signature

Advisor
Prof. Dr. İbrahim AKGÜN
Signature

Head of the Industrial Engineering Program
Prof. Dr. İbrahim AKGÜN
Signature

ACCEPTANCE AND APPROVAL

M.Sc. thesis titled “A Novel Approach Based on Bagging and Boosting for Imbalanced Classification Problems” and prepared by Muhammed Şafak PİNAR has been accepted by the jury in the Industrial Engineering Program at Abdullah Gül University, Graduate School of Engineering & Science.

18/08/2022

(Thesis Defense Exam Date)

JURY:

Advisor: Prof. Dr. İbrahim AKGÜN .....

Member: Prof. Dr. Emel KIZILKAYA AYDOĞAN .....

Member: Dr. Öğr. Üyesi Selçuk GÖREN .....

APPROVAL:

The acceptance of this M.Sc. thesis has been approved by the decision of the Abdullah Gül University, Graduate School of Engineering & Science, Executive Board dated ….. /….. / …….. and numbered ……..……. .

…….. /…….. / ……..

(Date)

Graduate School Dean
Prof. Dr. İrfan ALAN

ABSTRACT

A NOVEL APPROACH BASED ON BAGGING AND BOOSTING FOR IMBALANCED CLASSIFICATION PROBLEMS

Muhammed Şafak PİNAR
MSc. in Industrial Engineering
Advisor: Prof. Dr. İbrahim AKGÜN

August 2022

Classification algorithms are employed in a wide range of real-world problems such as obstacle detection, fraud detection, medical diagnosis, spam detection, speech recognition, image processing, intrusion detection, and so forth. However, it is not always an easy task to propose a legitimate classifier. For a classification task, there are numerous limitations of datasets. One of the most confronted limitations in real-world classification tasks is skewed class distribution, also called the class imbalance problem. When learning is employed in class imbalanced datasets without incorporating appropriate adjustments into the existing algorithms, minority classes are mostly misclassified. This study introduces a novel classification algorithm that outperforms previous studies on benchmark datasets used for the class imbalance problem. The presented novel algorithm, namely, BagBoost, involves aggregating modified bagging and modified boosting algorithms to increase the visibility of minority class instances.

The state-of-the-art algorithms in the classification of imbalanced datasets are investigated. The results of the best existing algorithms are compared with the proposed algorithm using benchmark datasets. Results show that BagBoost is a better classifier than commonly used classification algorithms in the literature for benchmark datasets according to F-measure and G-mean scores.

Keywords: Classification, Class Imbalance, Ensemble Classifiers

ÖZET

DENGESİZ SINIFLANDIRMA SORUNLARINA TORBALAMA VE ARTTIRMA ESASLI YENİ BİR YAKLAŞIM

Muhammed Şafak PİNAR
Endüstri Mühendisliği
Tez Yöneticisi: Prof. Dr. İbrahim Akgün

Ağustos 2022

Sınıflandırma algoritmaları, engel tespiti, dolandırıcılık tespiti, tıbbi teşhis, istenmeyen posta tespiti, konuşma tanıma, görüntü işleme, izinsiz giriş tespiti ve benzeri gibi çok çeşitli gerçek dünya problemlerinde kullanılır. Ancak, meşru bir sınıflandırıcı önermek her zaman kolay bir iş değildir. Bir sınıflandırma görevi için, çok sayıda veri kümesi sınırlaması vardır. Gerçek dünyadaki sınıflandırma görevlerinde en çok karşılaşılan sınırlamalardan biri, sınıf dengesizliği sorunu olarak da adlandırılan çarpık sınıf dağılımıdır. Öğrenme, sınıf dengesiz veri kümelerinde mevcut algoritmalara uygun ayarlamalar yapılmadan kullanıldığında, azınlık sınıfları çoğunlukla yanlış sınıflandırılır. Bu çalışma, sınıf dengesizliği problemi için kullanılan kıyaslama veri kümeleri üzerinde önceki çalışmalardan daha iyi performans gösteren özgün bir sınıflandırma algoritması sunmaktadır. Sunulan yeni algoritma, yani BagBoost, azınlık sınıfı örneklerinin görünürlüğünü artırmak için değiştirilmiş torbalama ve değiştirilmiş artırma algoritmalarının bir araya getirilmesini içerir.

Dengesiz veri kümelerinin sınıflandırılmasında en gelişmiş algoritmalar araştırılmıştır. Mevcut en iyi algoritmaların sonuçları, kıyaslama veri kümeleri kullanılarak önerilen algoritma ile karşılaştırılmıştır. Sonuçlar, BagBoost'un F-ölçü ve G-ortalama puanlarına göre kıyaslama veri setleri için literatürde yaygın olarak kullanılan sınıflandırma algoritmalarından daha iyi bir sınıflandırıcı olduğunu göstermektedir.

Anahtar kelimeler: Sınıflandırma, Sınıf Dengesizliği, Kolektif Sınıflandırıcılar

Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Dr. İbrahim AKGÜN for his enormous support and guidance in all my research that started from my B.Sc. degree and has continued until now.

I would like to also express my sincere gratitude to my wife Hale Nur PİNAR for giving feedbacks on this study, especially on visual elements she put an enormous effort. Other than that, she looked after our child Akif Selim PİNAR and created space for me to work on my study. It is a great pleasure to also acknowledge the continuous support and great love of my parents Ebül Fatih PİNAR and Gülhan PİNAR.

Additionally, I would like to thank my friends İbrahim Tümay GÜLBAHAR, Rahime Şeyma BEKLİ, Sami KAYA, Hasan Hüseyin BİRİNCİ, Ramis Akın KORKMAZ and Mehmet Fazıl KAPÇI for their supports of constantly reading the parts finished and giving feedbacks to idealize the written expressions of the study.

I also would like to thank my jury members Prof. Dr. Emel KIZILKAYA AYDOĞAN and Dr. Öğr. Üyesi Selçuk GÖREN for their valuable time and guidance.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AUC	Area Under Receiver Operating Characteristic Curves
BBC	Balanced Bagging Classifier
BRF	Balanced Random Forest
CIC	Canadian Institute for Cybersecurity
EBB	Exactly Balanced Bagging
$F_\beta$score	Harmonic Mean of Precision and Recall
FP	False Positive
FPR	False Positive Rate
Gmean	Geometric Mean of Precision and Recall
IDS	Intrusion Detection System
IR	Imbalance Ratio
ML	Machine Learning
NN	Neural Networks
RBB	Roughly Balanced Bagging
ROC	Receiver Operating Characteristic Curve
RUS	Random Under-Sampling
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machines
TP	True Positive
TPR	True Positive Rate

To My Family

Chapter 1

Introduction

In the data age, extracting insights from data has attracted substantial research interest. However, the capacity of the human brain is not improved enough to process a large amount of data. To achieve better accuracy for learning tasks compared to the human brain, people generated algorithms on computers that can learn like a human being. For this reason, researchers are constantly developing new methods and algorithms to learn from data.

The first imitation of the human neuron is discovered by McCulloch and Pitts in 1943 to mimic the human in a learning process [1]. This revolution is the first successful mimic of neurons. On top of the idea of imitating the human nervous system, Frank Rosenblatt discovered the Perceptron in 1957 and created the first algorithm that can recognize the letters of the alphabet by using the features of each letter [2]. Perceptron is the first "Machine Learning" (ML) algorithm. As the name implies, the aim of ML is that a computer (i.e., machine) 'learns' from the features of existing data to predict the label (i.e., the name of the letter for this example) for future, un-labeled observations. To recognize a new letter, Rosenblatt 'trained' his model by features and the names of letters. In ML, feeding the algorithm with labels of the instances to predict the classes or labels of new observations is called classification or supervised learning. In other words, using the names of letters with the features of corresponding letters in the above example is a classification problem. Even though the field of ML is enormously expanded today, and more accurate classification algorithms are developed, the very first ML algorithm Perceptron has the same idea of labeling an observation by its features. In other words, classification algorithms aim to categorize the data into previously determined classes by considering their features.

Classification algorithms are employed in a wide range of real-world problems such as obstacle detection, fraud detection, medical diagnosis, spam detection, speech recognition, image processing, intrusion detection, and so forth. However, it is not always

an easy task to come up with a legitimate classifier. For a classification task, there are numerous limitations of datasets. One of the most confronted limitations in real-world classification tasks is to have skewed class distribution, which is also called the *class imbalance problem*. In class imbalanced domains, one or more of the classes, which are called *minority classes*, are significantly less represented than other classes in the data, i.e., *majority classes*. In other words, the instances belonging to minority classes are outnumbered by instances belonging to other classes. When learning is employed in class imbalanced datasets without incorporating appropriate adjustments into the existing algorithms, minority classes are mostly misclassified to maximize the overall accuracy, i.e., the ratio of correctly classified test instances to all test instances.

In this study, the aim is to introduce a novel classification approach that outperforms previous studies on benchmark datasets used for the class imbalance problem.

1.1 Class Imbalance Problem

Class imbalance affects the performance of learning algorithms adversely. To understand the burden of classification algorithms on imbalanced domains, consider a binary case where there are only, two categories: -1 (negative) and 1 (positive). Suppose that the number of observations in the negative (majority) class is considerably high compared to the number of observations in the positive (minority) class. When algorithms do not take the class imbalance problem into account, the classification process is prone to ignore the minority class because the prior probability of the majority class is exorbitant compared to the minority class [3]. Learning algorithms aim to minimize a loss function, which is generally a misclassification rate. Since the observations of the positive class are extremely low in comparison to the negative class, learning algorithms tend to classify all examples as members of the negative class to minimize the misclassification error, and hence the positive class examples cannot be detected. To exemplify, consider a computerized network case where there are one hundred access requests from different users. Suppose that two of these requests are intrusions, i.e., unauthorized access requests. If an algorithm classifies all access requests as benign, i.e., authorized access, its overall accuracy will be 98%. However, all observations of the minority class will be misclassified. For problems such as medical diagnosis, fraud detection, intrusion detection, or obstacle detection as in autonomous cars, a misclassification may result in fatalities and/or big financial losses. In this regard, it is crucial to consider skewed class

distribution while learning from data and evaluating the results of each algorithm to have an operating learning algorithm [4].

Numerous algorithms have been proposed to mitigate the class imbalance problem. The literature on the classification of imbalanced datasets is divided into five main groups: (1) data-level methods, (2) algorithm-level methods, (3) cost-sensitive learning, (4) ensemble learning methods, and (5) ensemble of ensembles. In data-level methods, the aim is to manipulate the dataset and create a balanced classification task. This is achieved by generating new minority class examples or deleting instances from the majority class using different methods. It is the most straightforward way of tackling the class imbalance problem. Algorithm-level methods include the modifications of existing classification algorithms to mitigate the class imbalance problem. The general approach is to revise the loss function to better represent minority class instances in the learning process. Cost-sensitive learning methods aim to assign a given misclassification cost to all observations or to each class to minimize the bias towards the majority class. Cost-sensitive methods assume a cost matrix for each class or each instance that is misclassified. Ensemble learning algorithms combine several learning algorithms to obtain a more accurate model. Another family of algorithms, which is generally considered a part of ensemble learning methods, is the *ensemble of ensembles* where two or more ensemble algorithms are combined to obtain a more accurate algorithm. Even though ensemble-of-ensembles learning algorithms are proven to be appropriate for most learning tasks, they are not developed to learn in imbalanced domains. However, they can better mitigate the class imbalance problem and especially ensemble-of-ensembles algorithms are superior to the aforementioned methods with regard to their classification performance when they are hybridized with data-level methods and cost-sensitive approaches (e.g., [4],[5],[6],[7]).

The results of computational tests with the existing algorithms for imbalanced datasets show that their predictive accuracy on minority classes is too low, especially when the number of instances is comparatively less. In this regard, further investigation needs to be carried out on this family of algorithms. Motivated by this need, this study proposes a novel ensemble-of-ensembles classification approach to handle class imbalance problem that is a hybridization of bagging and boosting algorithm with better overall performance than that of previous models. The approach involves aggregating modified bagging and modified boosting algorithms to increase the visibility of minority class instances by rearranging the imbalance ratio at each bootstrap. Bagging is an ensemble learning algorithm that consists of several classification algorithms called *base classifiers*. The

base classifiers are trained on different samples of the original data then the prediction results of these classifiers are aggregated to have a superior classification algorithm to all *base classifiers.* With a similar manner Boosting is another aggregation of several classification algorithms. The difference is that, in Boosting the classifiers called *weak learners,* affects the *weak learner* by weighting the misclassified instances at each learning iteration. Computational results indicate that the proposed algorithm performs 15% better than the second-best algorithm on average of 24 datasets according to F-measure and 6% better than the second-best algorithm according to G-mean score.

1.2 Thesis Structure

The outline of the rest of the thesis is as follows.

Chapter 2 states the problem and reviews the literature addressing the class imbalance problem as well as evaluation criteria. Existing learning algorithms are investigated under five main categories, namely, (1) data-level methods, (2) algorithm-level methods, (3) cost-sensitive learning methods, (4) ensemble learning methods, and (5) ensemble of ensembles, and benchmarking methods including the measurements are overviewed. The results of computational tests with the existing algorithms for imbalanced datasets indicate that their performance varies significantly depending on the features of datasets such as sample size or the imbalance ratio, which emphasizes the need to further investigate the classification task in imbalanced domains.

Chapter 3 proposes a novel classification algorithm based on hybridization of bagging and boosting with a new approach to accurately classify imbalanced datasets and discusses logical and intuitive reasoning behind the good performance of the algorithm.

Chapter 4 gives the experimental setup, benchmark datasets, and the algorithms used to compare to the proposed approach. The performance of the proposed algorithm is tested using four benchmark datasets that are proven to perform better than other approaches in the literature. 24 benchmark datasets are trained on each of the benchmark approaches and the proposed approach, and the F-measures and G-mean scores are compared to evaluate the approaches.

Chapter 5 compares the results of existing best algorithms to those of the proposed algorithm using benchmark datasets. The results show that the proposed algorithm is a better classifier than commonly used classification algorithms in the literature for benchmark datasets according to F-measure and G-mean score.

Chapter 6 will conclude the study with a discussion on Societal Impact and Contribution to Global Sustainability as well as future research ideas.

Chapter 2

Research Background

In this chapter, we first discuss class imbalance problem as well as evaluation metrics appropriate in imbalanced domains. Secondly, we give a comprehensive review of the literature and emerging solutions to the class imbalance problem.

2.1 Problem Statement

As a part of ML algorithms, classification is one of the most used data processing methods. The aim is to categorize new instances into preassigned classes based on past data. As shown in Figure 2.1, classification algorithms consist of three main phases: Preprocessing, Training, and Prediction.

In the preprocessing phase, the data is rearranged in a way that the classification algorithm can recognize the specificities of the instances. The feature space, the observation space, or in some problems the number of observations or the labels may be rearranged according to the needs of the applied algorithm. This phase differs from algorithm to algorithm and according to the needs of the decision maker. In the training phase, the minimizer parameters of a loss function are “learned” according to the classification performance of the learning algorithm using the labeled instances. The loss function differs from algorithm to algorithm. However, in traditional classification algorithms, it is generally an error term that represents the misclassification rate. To understand the learning process better, consider the following example of three students to be classified as “successful” or “not successful”. The students are instances in the learning algorithm in which two of them are from the negative class i.e., not successful and one is from the positive class i.e., successful. Suppose that we will decide whether a student is successful or not via the scores from two courses, “X” and “Y”. and we know if the student is successful or not. The loss function in this case is fed with the scores of each student from courses X and Y and outputs -1 (not successful) or 1 (successful). Then, to optimize the parameters of loss function the outputs of loss function is compared with actual classes

of students. After that, modifies the parameters of loss function to decrease the misclassification rate of students.

In the prediction phase, the aim is to use the optimized loss function parameters during the training phase to predict the categories of new unknown and unlabeled instances. The performance of proposed models on new data is evaluated. Since both the loss function and the ranking of different models are extremely prone to changes in the evaluation method, there are several evaluation functions used with different learning tasks. To exemplify, overall accuracy, which is the ratio of the number of correctly classified instances to the number of all instances, is one of the mostly used evaluation metrics. However, it ignores the class imbalance problem. When the Imbalance Ratio (IR) is high, i.e., the majority class has much more instances than the minority class, the minority class instances are mostly misclassified even if the accuracy is high.

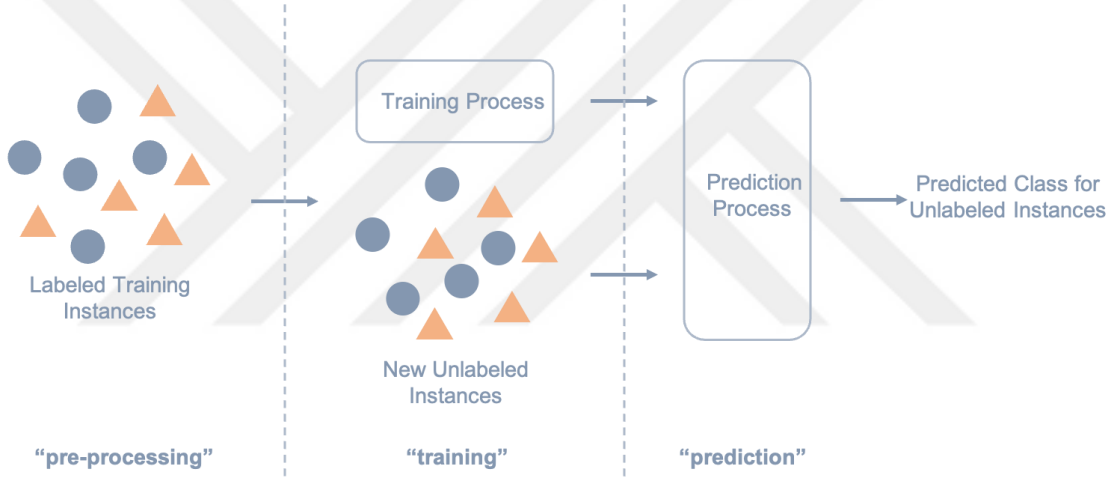

Figure 2.1 Illustration of Classification Algorithms

This generally results in a high misclassification error on minority class. In real-world problems, misclassifying almost all the minority class examples is an unacceptable flaw of the classification algorithm.

To understand the class imbalance problem, one should understand the meaning of an under-represented sample. Figure 2.2 presents three different binary-labeled datasets with under-represented classes. Blue circles represent minority class instances while orange triangles represent majority class instances because the circles are outnumbered by the triangles. Traditional classification algorithms are biased towards the majority class because they try to minimize the overall accuracy. The next discusses appropriate measurements for class imbalance problem.

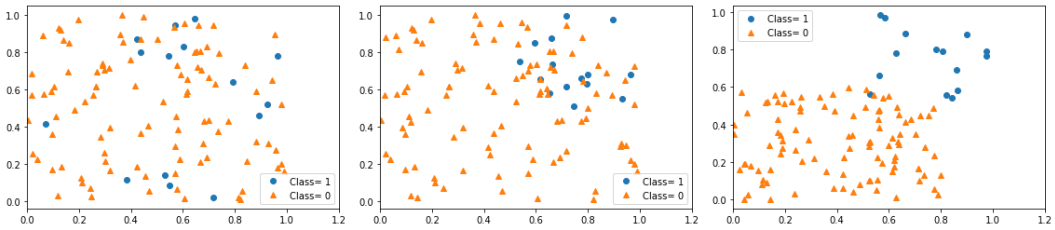

Figure 2.2 Class Imbalance Illustration

2.2 Evaluation Metrics

Several measures are used in classification algorithms. Accuracy is one of the mostly used evaluation metrics on classification tasks; however, it is not a proper performance measure for goodness of fit in imbalanced domains [8]. When the imbalance ratio (IR) is high, models trained according to accuracy tend to ignore minority class instances. In order to overcome this problem, three measures have been introduced as Area Under ROC curve (AUC), F-measure and G-mean.

ROC (Receiver Operator Characteristic Curve) is a graph showing the change in True Positive Rate (TPR) over False Positive Rate (FPR) where TPR (also called recall) and FPR are defined as in Equation (2.1) and (2.2), respectively.

$$FPR = \frac{\text{FP}}{FP + TN} \tag{2.1}$$

$$TPR\ or\ Recall = \frac{TP}{TP + FN} \tag{2.2}$$

False Positive (FP) is the number of instances that are incorrectly classified as positive. In other words, they are the instances classified as positive class observations, but they are negative class observations. True Positive (TP) is the number of instances correctly classified as positive class observations. False Negative (FN) is the number of instances that are incorrectly classified as negative. AUC is not biased towards the majority class, but it suffers from underrepresenting FN [9]. The representation of FPR which normalizes FP with TN represented in Equation 2.1 decreases the importance of FN [10]. Hence, the importance of misclassification of a positive class example as negative is decreased. In this case, minority class examples are ignored during the learning process. To that extent, using measures that do not ignore FN is crucial on imbalance domains. For this reason, using precision and recall-based metrics that do not fail on representing FN is a better

approach on imbalance domains. Hence, using precision with recall is a better way to tackle the class imbalance problem. For this reason, both F-measure and G-mean that using precision instead of FPR are better measurements on imbalanced domains.

F-measure also known as the harmonic mean of precision and recall is known to be a proper and widely used measure on imbalanced domains. Precision is defined as in Equation (2.3).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{2.3}$$

Hence, the F-measure is defined in Equation (2.4).

$$F_{\beta}\mathrm{score} = (1 + \beta^2) \times 2 \times \frac{precision\ x\ recall}{(\beta^2 \times precsion) + recall} \tag{2.4}$$

where β is a coefficient that represents the importance of recall against precision. In general, β is set to 1 to give equal importance to precision and recall.

G-mean is the geometric mean of precision and recall and defined in Equation (2.5).

$$Gmean = \sqrt{precision\ x\ recall} \tag{2.5}$$

Both F-measure and G-mean are considered to be the performance measures for imbalanced domains. Hence, in this study, to evaluate proposed model and give a comparison with existing literature F-measure and G-mean will be used.

2.3 The Literature Review

The solution techniques to learn from imbalanced data are divided into five categories according to the mitigation strategy used: (1) Data-Level Methods, (2) Algorithm-Level Methods, (3) Cost-sensitive Methods, (4) Ensemble Learning Methods and (5) Ensemble of Ensembles according to following studies [11],[12],[13],[14],[15],[16],[17],[18]. In the following, we give the literature related to these five categories.

2.3.1 Data-Level Methods – Sampling

In the data-level approaches, the idea is basically to sample data to have a balanced dataset. Since skewed data lead to an insufficient learning process, these methods

preprocess data to prepare for the actual learning process. Data-Level Approaches can be categorized into three groups: Under Sampling (US), Over Sampling, and a blend of these two, namely, Hybrid Methods.

2.3.1.1 Under-Sampling

The first method dealing with imbalanced data is Under-Sampling Method which, creates a balanced dataset by deleting instances from the majority class. In this way, the imbalanced ratio (IR) (i.e., the ratio of the cardinality of majority class to the cardinality of minority one) is decreased close to 1.

The downside of an under-sampling algorithm is the loss of potentially useful information. Moreover, in the case that the prior imbalance ratio is as high as 99, the deleted instances may be fundamental to frame the majority class.

Several algorithms were created using the under-sampling idea. These under-sampling techniques differ in the removal process to achieve less information loss. The primary idea is to remove instances from majority class examples randomly until the imbalanced ratio is 1 (i.e., Random Under-Sampling). As there is a big loss of meaningful data with this method, researchers came up with more clever ideas than randomly discarding some majority class examples. Some of the methods built on top of this idea are Tomek Links [19], One-Sided Selection [20], Under-sampling Based on Clustering [21], Class Purity Maximization [22], and so forth.

2.3.1.2 Over-Sampling

Over-Sampling method creates new instances from minority class examples. In the most primitive case, the minority class examples are replicated until the imbalanced ratio is 1. The main drawback of the over-sampling method is overfitting [3],[18]. Overfitting is the problem of fitting a too complex curve on the training data. In this case, the learning algorithm memorizes the training data instead of learning from it and thus, the evaluation of the test set fails to yield good results. In other words, at the training stage of the learning process, the learning algorithm tries to classify all the examples correctly by increasing the complexity of the fitted curve. Additionally, since the number of instances is increased enormously according to the IR, the computational complexity is increased in the training phase of learning algorithms. When IR is very high, oversampling algorithms are not compatible with other learning algorithms according to model performance in class imbalance problems.

The most promising and widely used algorithm that follows an over-sampling procedure is SMOTE (*Synthetic Minority Oversampling Technique*) [23]. In SMOTE, instead of copying minority class examples several times, new minority class examples are populated using interpolation of minority class examples. Even though SMOTE creates achieves a balanced dataset with a smart approach, there is a crucial drawback of this algorithm; interpolated new instances may be in the majority class space and thus, segregating the new instances created from the majority class is hard to handle. Having instances labeled as minority class in majority class space yields the wrong labeled training dataset, which may cause overfitting [24]. Additionally, SMOTE with imbalanced data gives a bad classification performance at the borders of minority space [25].

To solve these problems, many extensions of SMOTE have been generated. However, these extensions cannot solve the mentioned problems of SMOTE.

2.3.1.3 Hybrid Approaches

Using undersampling and oversampling in a standalone mode is not promising for imbalanced data classification tasks however, using them together enhances the performance of classification algorithm. Hybrid methods arise from the main downsides of undersampling and oversampling methods. They minimize information loss by reducing fewer data points from the majority class. Moreover, the overfitting probability is somehow decreased since algorithms copy fewer minority class examples to oversample. In this way, the results of hybrid algorithms are more promising than undersampling and oversampling. The aim is to find a balance between undersampling and oversampling that increases the model performance on imbalanced domains. For this reason, minority class examples are firstly oversampled and then, the majority (sometimes both majority and minority) class examples are undersampled. Traditional classification algorithms perform better on this preprocessed training dataset.

Even though hybrid methods are better at classification performance than their predecessors, it is not quite a solution to all the problems of data-level approaches to imbalanced problems. Because in the oversampling phase, new instances from minority classes are created by interpolation, some of the instances lie deep in the majority class space and hence, the hybrid approaches may suffer from overfitting [25].

Data-level approaches are used in a variety of studies in the literature. However, they have key deficiencies such as overfitting or loss of information.

2.3.2 Algorithm-Level Methods

Algorithm-level approaches aim to create a robust classification model that can accurately classify datasets with skewed class distribution by improving the traditional classification algorithms. To alleviate the bias towards the majority class, several existing learning algorithms are revised as Support Vector Machines (SVM), Neural Networks (NN), Decision Trees, and so forth. In each of these algorithms, several different modifications lead to different algorithms. As in data level approaches, there are several limitations of these modifications.

For algorithm-level approaches, due to flexibility and generalization ability, one of the mostly used methods for imbalanced classification tasks is the modification of SVMs. Using Kernels with SVMs provides a more flexible model that allows for specific changes in the loss function. Kernels are used to map the complex decision surface into a higher dimensional space to have a less complex decision boundary. In this way, learning gets easier and computational complexity is decreased. Additionally, by definition, Kernel functions give inner products of support vectors and hence the data itself is not important after the training phase. Feeding the Kernel function with support vectors and new instances is enough to predict the label of the new instance.

Considering the ability of Kernels on manipulating data, modifying Kernels to provide a better classification performance on imbalanced domains is not a surprise. One Kernel modifications is the conformal transformation of SVMs. The trick is to increase the precision of the ML algorithm around the decision boundary and decrease when the instances are away from the boundary [27] aims to give more priority to the instances around the decision boundary and so decrease the misclassification rate around the decision boundary. To handle class imbalance problem, Kernel Matrix which is the support vector is modified in the favor of minority class instances. Over the years, the algorithm is improved preserving the main idea of shifting the decision boundary between negative and positive classes. Two useful modifications of the shifting strategy are weighting classes with the inverse proportions of class labels [28] and optimizing the decision threshold alignment by maximizing the intra-class margin [29].

Another approach using conformal transformation of Kernel is called Kernel scaling. This approach despite the boundary shift algorithm focuses on increasing the precision on both sides of the decision boundary. Since the enlargement of the positive and negative spaces

are independent and decided according to the IR it is possible to prioritize the minority class space above the majority class space to ensure taking the imbalance data into account. This approach requires two consecutive learning phases.

Choosing the best Kernel function according to the classification problem at hand using Kernel alignment is yet another approach. To ensure the fitness of the used Kernel, inter-class and intra-class distances are compared. A higher inter-class and lower intra-class margin yield a good fit of model/Kernel. By this model/Kernel selection procedure, one may ensure to alleviate the bias toward the majority class [30].

Besides modifications, there are several weighting approaches to modify SVMs to handle imbalanced datasets. These approaches are mainly divided into two main groups: *instance weighting* and *support vector weighting.* In instance weighting the importance of instances from different classes are rearranged to mitigate with imbalance data. By the nature of imbalanced classification tasks, the minority class instances are unfavorable since their prior probabilities are less than majority class instances. Arranging the class weights, i.e., assuming the same weight for each instance in a class according to the likelihood of each class in a reverse manner is proposed in [31]. Weighting outliers, borderline instances, support vectors, or small disjuncts are other developments of the instance weighting approach. One of the most popular developments is using boosting algorithms to assign weights. In that case, boosting algorithms are used to optimize the instance importance according to the base classifier [32]–[35]. Another division of weighting approaches is called support vector weighting which, it aims to alleviate the impact of skewed class distribution by multiplying the margin of minority class support vectors with a positive coefficient [36]. Since the importance of minority class support vectors is increased, the bias towards the majority class is decreased.

Since in data level approaches the data to feed the ML algorithm is preprocessed, it enables us to use any ML algorithm afterward. However, the training algorithm itself is revised in algorithm-level approaches and hence we sacrifice that ability. In counter, algorithm-level methods may be tuned specifically to the problem better than sampling methods [26] and hence in general, algorithm-level approaches yield a better classification performance on imbalanced domains.

2.3.3 Cost-Sensitive Learning

Classification is (in most cases) minimization of misclassification error by assigning instances to classes. In imbalanced cases, the main problem is the cost functions inability properly to explaining that the minority class is misclassified so that even though all the instances of minority class are misclassified the error is reasonable. Cost-sensitive methods are investigated under two main groups, namely, *direct approaches* and *meta-learning approaches* [37]. In the direct approaches misclassification cost for each class is introduced. In this case, the cost function of the classification algorithm is manipulated, hence this type of cost-sensitive method resembles algorithm-level approaches. In meta-learning approaches, a misclassification cost is assigned to all instances to prevent bias towards the majority class. Since assigning a cost to instances is preprocessing and no structural change is made to the classification algorithm, this type of cost-sensitive approach is like a data-level approach. Since it is a data manipulation as in data-level approaches, most of the classification algorithms can be trained after preprocessing phase. However, since the algorithm is modified (mostly the cost function) with direct approaches, it is harder to apply compared to meta-learning approaches.

The main contribution of cost-sensitive approaches is to introduce the cost of misclassification in the favor of minority class. This idea is applied to various algorithms such as SVMs [38], Neural Networks ([39], [40]) and decision trees ([41]–[43]) to prevent bias toward the majority class. In this family of algorithms, the most promising methods to tackle skewed class distribution problems are the modifications of decision trees. While applying meta-learning approaches, the instance weights are calculated, and a decision tree classifier is trained on the weighted dataset. While applying direct approaches, the cost function of the decision tree algorithm is rearranged to alleviate the effect of imbalance. By assigning a higher misclassification cost is assigned to the minority class during the training of the algorithm the bias towards the majority class is decreased.

Even though cost-sensitive algorithms are promising, producing a proper cost matrix is hard. Since the algorithm learns the classes according to the given cost matrix in both meta-learning and direct approaches, the cost matrix is significant for an adequate learning algorithm. The root cause of problems related to the creation of a cost matrix is that the cost matrix is dependent on the training data. There are infinitely many cost

matrices that may be assigned to the classes (in direct approaches) or instances (in meta-learning approaches). For this reason, it is hard to select the best cost matrix that will maximize the classification performance. Additionally, if the data has limitations such as noisy class distributions, the presence of instances that have a small inter-class margin or overlap of classes on instance space, it is even harder to procure a proper cost matrix ([44], [45]). To tackle with the aforementioned problems, hybrid methods are proposed that uses the data-level approaches as preprocessing to direct cost-sensitive approaches. Hence, the preprocessing phase is used to eliminate the data-related limitations ([46], [47]).

2.3.4 Ensemble Learning Methods

Ensemble Learning is a fusion of several classifiers namely base classifiers to create a better performing model than the best one of base classifiers. Base classifiers should yield different outcomes called diversity of base classifiers, because having the same result in each base classifier will not improve the performance aggregate classification model. This requires, the models or the training dataset is variable at each training process of base classifiers [48]. Ensemble learning algorithms used to tackle class imbalance problems are mainly investigated in two parts: bagging and boosting. Bagging introduced by Breiman [49] is a combination of bootstrap and aggregate. In the bootstrap phase, several datasets are sampled from the original data called bags. Each of these samples is trained with a base classifier. Base classifiers are the same algorithm trained on different bags. In aggregating phase, the trained models are aggregated to make a prediction. The training sample at each iteration is different to preserve different results for each base classifier called diversity of the model. On the other hand, boosting as proposed by Schapire [50] assigns weights to instances in an iterative manner. In every iteration, weights of the instances that are correctly classified at the previous iteration are decreased and weights of the ones that are misclassified are increased. The learning process in each iteration is mediated through the weak learners. Weak learners are classification algorithms that have classification performance slightly better than a random guess. In each iteration, weak learners are fed with a weighted dataset from the previous iteration and weights are updated accordingly. In this way, in each iteration, a new result is created. Hence, as in the bagging model, the results of weak learners are aggregated to have a better learner than all the weak learners.

Ensemble-based learning methods addressing imbalanced classification problems are generally hybrid methods. As known, they are sensitive to skewed class distribution with a stand-alone setup. In a hybrid setup, ensemble-based learning algorithms are broadly used with data-level approaches and cost-sensitive learning algorithms. For this reason, they are frequently classified as hybrid methods in literature [13], [51], [52]. In the following part, the ensemble-based algorithms that are created to handle class imbalance problem will be discussed.

Ensemble classifiers may be combined with cost-sensitive methods in two ways: cost sensitive boosting approach and cost-sensitive weak learners. Cost-sensitive boosting approach rearranges the weights of instances in each iteration of boosting algorithms to handle class imbalance problems [53]. There are several applications of these algorithms e.g., AdaCost [54], RareBoost [55] and AdaC (from 1 to 3) [12]. They differ in their weighing procedure as cost-sensitive algorithms do.

In cost-sensitive weak learners in each iteration of ensemble the weak learner is cost-sensitive. They are generally differentiated on the base classifier itself. In some cases, not the classifier but the way of handling the cost matrix is changed. Some successful applications of this type of cost-sensitive ensembles are BoostedCS-SVM [56], BoostedWeightedELM [57], CS-DT-Ensemble [58], AL-BoostedCS-SVM [59] and IC-BoostedCS-SVM [32]. However, the main drawback of cost-sensitive boosting is creating a proper cost matrix by the nature of cost-sensitive algorithms as explained in Section 2.3.

Ensembles built upon data-level approaches are divided into three main categories: boosting-based, bagging-based, and hybrid methods. There may be confusion on the usage of the word hybrid twice, one for the ensemble built on top of data level approaches and the other for the hybridization of bagging and boosting methods. The hybrid approaches mentioned in this part represent the aggregation of bagging and boosting even though the main algorithm is a combination (i.e., hybridization) of data-level methods and ensembles. Boosting-based methods differ in the way they sample the data to obtain a balanced or nearly balanced dataset. Under-sampling, oversampling, and a combination of both techniques are used extensively. Moreover, the weights assigned to instances by the nature of boosting algorithms may change from one approach to another. Even in some applications, the order of weight assignment is changed to be before the training phase of boosting algorithms as in DataBoost-IM [57]. Other application using data level approaches prior to an ensemble learning algorithms are SMOTEBoost [60],

MSMOTEBoost [61], RUSBoost [62], EUSBoost [63], BalancedBoost [64], RAMOBoost [65], GESuperPBoost [66] and RB-Boost [67].

Another hybrid approach is a combination of data-level approaches and bagging classifiers. Since the bootstrapping phase of the bagging algorithm is a sampling from original data, bagging itself is like a data-level approach. Therefore, bagging is a less complex method compared to boosting methods that weigh individual instances at each iteration according to previous classifiers' accuracy on the instance. Additionally, after bootstrapping, any base classifier would be used without manipulating the algorithm. On the other hand, maintaining the diversity of bootstraps is the main issue. The algorithms of this type of hybridization are distinguished on that issue. Each algorithm has its sampling technique to preserve diversity. As explained in Section 2.1, data-level approaches are divided into three groups, namely, oversampling, undersampling, and hybrid methods. Bagging-based algorithms may be investigated under a similar terminology as over-bagging, under-bagging, and hybrid-bagging, respectively. Since bagging may be applied subsequentially to any preprocessing algorithm, all mentioned algorithms in Section 2.1 may be integrated with it. SMOTEBagging [68] is one of the best performing applications of over-bagging in imbalanced domains. Balanced Bagging Classifier (BBC) [69] created by Lemaitre et al, Roughly Balanced Bagging(RBB) [70], Balanced Random Forest (BRF) [71], Exactly Balanced Bagging [72], QuasiBagging [73], Asymmetric Bagging [74], Bagging Ensemble Variation [75], IRUS [76], α-TreeEnsembles [77], PUSB [78] are some of the mostly used under-bagging applications. UnderOverBagging [68], IIVotes [79], and RB-Bagging [67] are hybrid methods that use both undersampling and oversampling approaches. However, there are several algorithms proposed that are differentiated from this classification due to small changes in the sampling or learning algorithms such as USwitchingNED [80] and EPRENNID [81]. To the best of our knowledge, Balanced Bagging Classifier and Balanced Random Forest perform better than others where the sample size of the original dataset is small, and the IR is high.

2.3.5 Ensemble of Ensembles – Double Ensembles

Ensemble of ensemble is the hybridization of bagging and boosting algorithms. Bagging is the main classifier while the base classifiers are boosting algorithms. The main difference between these algorithms is the way that they handle bootstrap. There are

several applications adopting bagging and boosting together to increase diversity, accuracy, and robustness of ensemble models. In preparation for an imbalanced domain application, firstly we will discuss different algorithms within this context.

The first double ensemble strategy applied is Multiboosting [82] where the aim is to tackle bias and variance at the same time. The idea is built on top of Breiman's "Bagging Predictors" [49] and Freund's "Adaboost Algorithm" [83]. Breiman's search shows that bagging classifiers are strong in decreasing the variance of a classifier. Additionally, Friedman's discussion on bias/variance [84] and statistical analysis on boosting algorithms [85] show that boosting algorithms appear to be one of the most powerful bias reduction methods even though weak learners are highly biased. It is known that boosting algorithms also tackle the high variance problem on a stand-alone setup. However, bagging algorithms are known to be a more effective classifier for reducing the bias compared to boosting [86]. To that extent, training bagging classifiers using boosting classifiers as base classifiers decreases the bias and variance simultaneously [82]. The algorithm Multiboosting is a not direct combination of bagging and AdaBoost algorithm. Instead of bagging, wagging, which is admitted as a variant of bagging, is used. In wagging the idea is to give weights to each instance and then sample according to the given weights. There are several other applications that use bagging instead of wagging, e.g., Stochastic Gradient Boosting [87], MEL [88] and Iterated Bagging [89]. Cocktail Ensemble [90] is another ensemble of ensembles that combines different ensembles in parallel. The main idea is to learn without sampling by ensembles and to aggregate the results to decrease the bias and variance of the model. However, the Cocktail classifier is proposed not as a classifier but as a regressor. The aim is to predict a continuous label.

Liu et al. [4] are the first to propose a study that hybridized ensembles for imbalanced domains. They propose two algorithms, EasyEnsemble and BalanceCascade. They both originate from the major drawback of undersampling approaches. EasyEnsemble is likely to be the most straightforward aggregation of bagging and boosting. It is a bagging algorithm using AdaBoost as a base classifier. The usage of modified bagging algorithm together with AdaBoost results in an enormous decrease in bias and variance. Hence, the classification performance on minority class for imbalanced test set is superior compared to most of the other algorithms.

HardEnsemble [5] is another application of double ensembles on imbalanced domains. The proposed algorithm combines under-sampling and oversampling techniques. The bags are acquired by Reward-Punishment [90] and CSMOTE [5] and a RUSBoost model

[62] is trained on each bag. Fusion of RUSBoost classifiers is used to predict the label of new instances.

Cluster-based Under-sampling with Boosting (CUSBoost) [6] algorithm is proposed by Rayhan et al. Authors suggest a novel sampling technique for imbalanced domains. The instances of the majority class are clustered using the k-means algorithm. Then, each cluster is undersampled to form the majority class space of each balanced bag. A copy of minority class instances is used in each class as is. Each bag is trained with AdaBoost, and the results are aggregated. Since similar instances are clustered, intra-bag diversity increases but a loss of information occurs. The removed instances from the undersampling phase may be important identification of the majority class. Compared to other similar undersampling methods, information loss is decreased since the clustering phase ensures to have similar (clustered together) instances of majority class yield in the same bag and at each undersampling phase deletes instances from these bags of similar instances. In other words, there will be observations from each cluster of the majority class during the final learning process. Hence, the shape of the majority class space is preserved but the density of instances is decreased from each cluster. However, to ensure this, one should find a split in the majority class space that will yield optimal segregation in the clustering phase, which is not logical to expect. If the clusters are not segregated optimally then there may be some instances that are crucial to define majority class are removed from the training data.

EUSBoost: [7] propose by Lu et al. is another approach developed for learning in imbalanced domains. It samples a subset from the original majority class without replacement. The cardinality of subsets is equal to the minority class. Then, AdaBoost is trained on each of these samples. A weighted voting approach is used to aggregate the results of AdaBoost classifiers. Then a decision boundary search algorithm is used to increase the inter-class variance. Since the minority class is copied to each of the bags created, the inter-bag diversity while the bagging process is controversial.

Chapter 3

The Proposed Approach

In this chapter, we propose a novel classification algorithm for imbalanced datasets. Since the proposed algorithm is a hybridization of bagging and boosting algorithms, it is crucial to understand these algorithms. In the following, we first explain the Bagging and Boosting algorithms and then give the details of the proposed algorithm.

3.1 Bagging and Boosting Algorithms

Bagging, first proposed by Breiman [49], is an ensemble learning algorithm that is divided into two parts, namely, bootstrapping and aggregating as shown in Figure 3.1. The aim is to create a better learner by combining several learning algorithms. The bootstrapping is in the most basic terms randomly sampling from the original dataset with replacement and hence the new dataset is not the same as the original one.

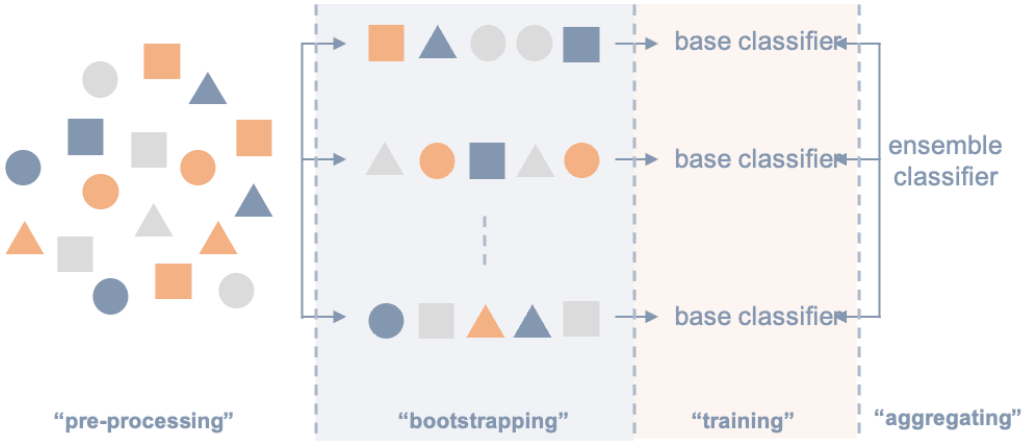

Figure 3.1 Bagging Classifiers

Because of the sampling strategy, there may be some instances represented more than once and ones not represented at all as shown in figure 3.1. After a given number of bootstraps are sampled from original data, a traditional learning algorithm chosen to optimize model performance, called base learner, is trained on each of the samples. Then,

the resulting trained models are aggregated for new unlabeled instances. The aggregation is created based on the "Wisdom of the Crowd" idea. The baseline is that having several independent and diverse learners that decide on the label of a new observation rather than a single learner is better results according to model performance. In the light of this idea, the resulting aggregated classification model is considered and proven to be better than single learners in general [49].

Boosting, developed by Schapire [50], uses the original dataset at each iteration and rearranges the weights of instances at each training step according to the classification accuracy of "weak learners" as shown in Figure 3.2. Weak learners are known to be learning algorithms slightly better than a random guess. In other words, the classification accuracy of a weak learner is expected to be slightly more than a random categorization. After a weak learner is trained on the predecessor weak learner, the instances are weighted accordingly by using the classification accuracy and used at the next iteration until the number of iterations is equal to the pre-assigned value.

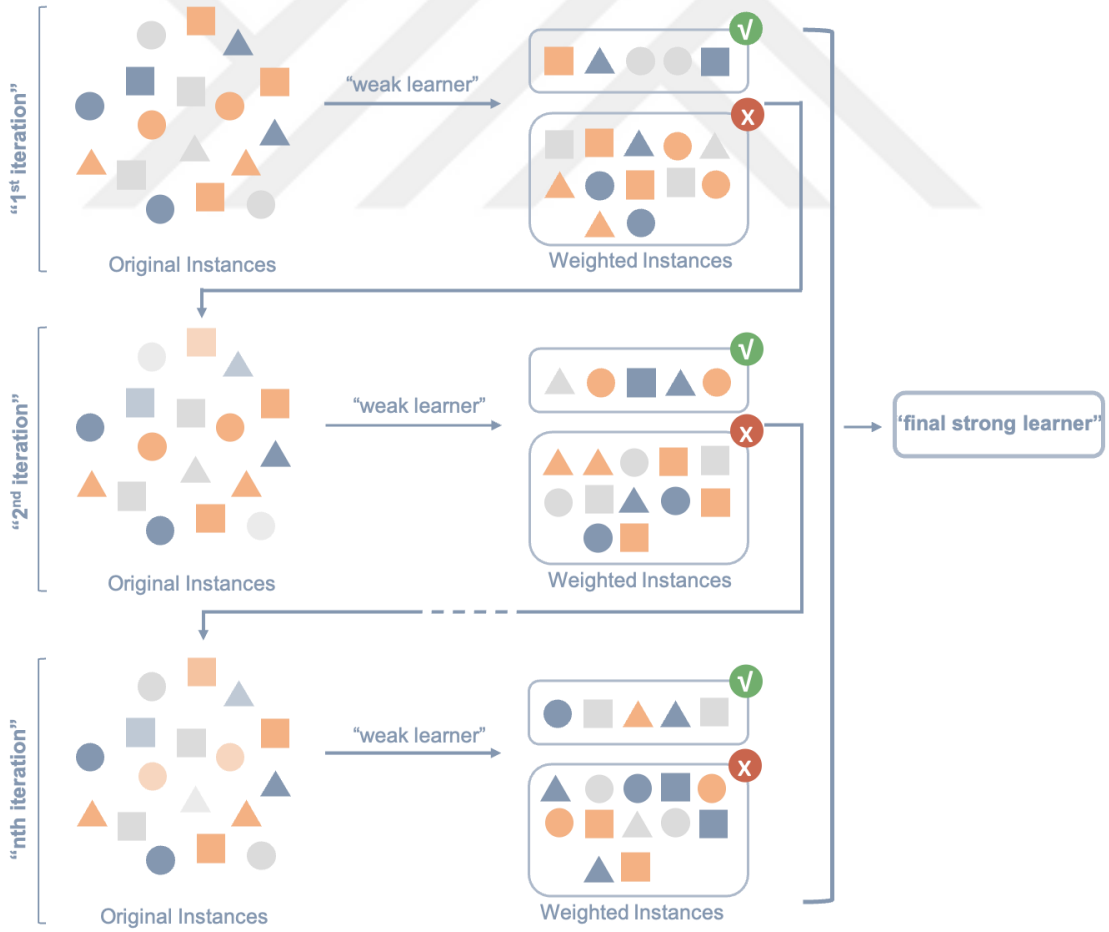

Figure 3.2 Boosting Classifiers

To explain the weighting process more intelligible, consider a small binary class classification task that uses a breast cancer dataset consisting of ten observations. At the first iteration, the weak learner is trained with the original data with an initial weight of 1 for each instance. Consider that 3 of these 10 observations are misclassified. In the next iteration, the weights of misclassified instances are increased and the weights of correctly classified instances are decreased. Let us set the weights of misclassified instances be 1.5 and the weights for correctly classified instances be 0.5. The weighted instances are used as input for the next iteration. Tweaking the weights of instances in the explained way causes an increase in the importance of misclassified instances for the next iteration. For this reason, at each iteration, the misclassification error decreases. After a given number of iterations is carried out, a linear combination of the weak learners is used to form a final strong learner by weighting according to the classification accuracy of each weak learner.

3.2 The BagBoost Algorithm

The proposed algorithm is a modified bagging classifier that uses boosting classifiers as base classifiers. Two main modifications are made to original bagging classifiers: a new bootstrapping algorithm and boosting base classifiers. The bagging classifiers have bootstrap and training phases as shown in Figure 3.1. In bootstrapping phase, the algorithm of sampling from original data is modified to handle the class imbalance problem. Additionally, the base learners of bagging are set to be tuned XGBoost classifiers to have a better-aggregated result.

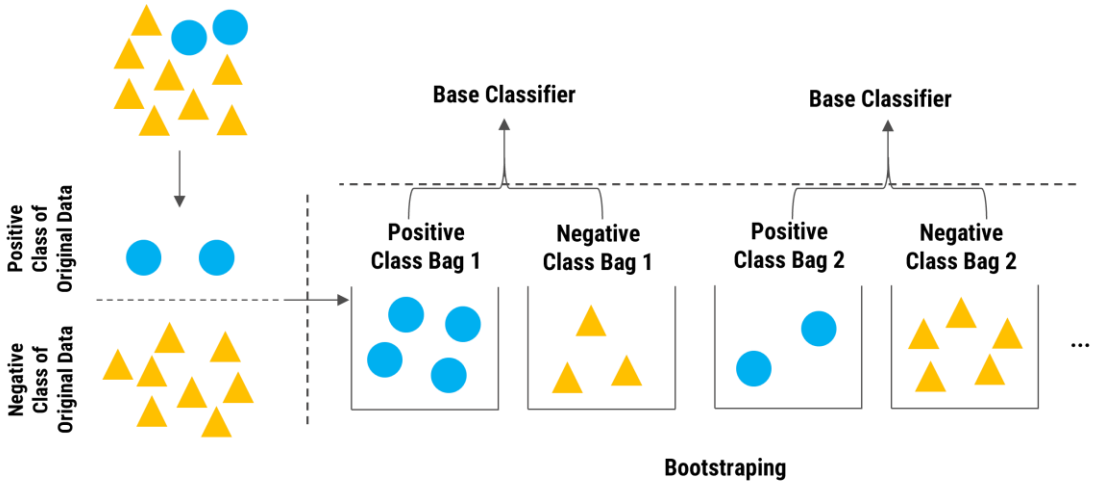

Figure 3.3 The BagBoost Algorithm

On the original bagging algorithm, the bootstraps are the same size as the original dataset. Additionally, positive and negative class instances are sampled simultaneously. Hence on the average, i.e., considering the average of all bootstraps, the IR stays the same. On the contrary, the BagBoost Algorithm first divides the original dataset into positive and negative instances as shown in Figure 3.3. Then, instances are sampled from negative and positive classes separately with different sample sizes that change at each iteration for each class. To determine the number of instances in positive and negative classes at each iteration, we generate two random numbers within a given range. The range is a parameter of the BagBoost algorithm that requires to be optimized according to the dataset and mostly affected by the sample size and IR. Two random number are selected within this range at each iteration, one number represents the positive class sample size while the other represents negative class sample size. The lower and upper bounds of the range are some ratios of minority class examples. For example, suppose that we have two positive class observations and eight negative class observations as in the Figure 3.3 with an initial IR value of 4. If the lower bound and upper bounds are chosen to be 50% and 250% of the original positive class by the parameter optimization tools, the range will be [1, 5]. Assume that, at the first iteration the random numbers for positive and negative class sample sizes are drawn from the range as four and three respectively as in the first iteration in Figure 3.3. Note that, the first bag is imbalance and the IR is smaller than one, which means that the number of instances in the positive class is higher than the number of instances in the negative class. For the second iteration assume that the sample sizes for positive and negative classes are two and five. In this case, the IR for the second bag is greater than one. This enables to increase the diversity of bags compared to the original bagging algorithm.

There are *N* bags are created according to the *number of iterations (N)* parameter of the BagBoost algorithm as stated in Algorithm 1. In the next phase of BagBoost algorithm, an XGBoost classifier, i.e., represented as base classifiers in Figure 3.3, is trained on each of these bags. In the training process of XGBoost algorithm at each iteration, the parameters of the algorithm are tuned according to the G-mean loss function instead of overall accuracy loss function. In this way, we ensure to consider the imbalance samples at each iteration to have better performing base classifier at each iteration. Note that, having different parameters of the XGBoost algorithm at each step of the BagBoost algorithm increases the diversity of base classifiers. For this reason, the BagBoost

algorithm increases the performance of classification compared to original bagging algorithm.

In the next step, the results of XGBoost algorithms are aggregated according to *majority voting approach.* Majority voting is one of the methods widely used to provide a single classifier from several base learners. The aim is to propose a better classifier than each of the base classifiers by using the "Wisdom of the Crowd" idea.

Algorithm 1 The BagBoost Algorithm

input: An input dataset: Consists of a set of minority class examples S_p and a set of majority class examples S_n, where $|S_p| < |S_n|$,
Two input parameters:
a- The number of iterations N, specifies the number of bootstraps
b- A range of the number of positive $[n_{ps}, n_{pe}]$, and negative $[n_{ns}, n_{ne}]$ class instances

output: A robust classification model

```
begin
    i = 0
    while i ≤ N do
        Randomly create n_pi ∈ [n_ps, n_pe] and n_ni ∈ [n_ns, n_ne] as the cardinality of positive and negative class
        Randomly sample S_pi from S_p, and S_ni from S_n where; |S_pi| = n_pi, and |S_ni| = n_ni
        Concatenate S_pi, and S_ni into I_i
        Train XGBoost algorithm on I_i and save the predictions of the unlabeled test set.
        i ⇐ i + 1
    end while
    Majority Voting
end
```

There are two benefits of BagBoost algorithm in imbalanced domains: increasing the diversity of bag set and using G-mean score function while training base classifiers. In the following, we will explain these benefits in detail, respectively.

As shown in Figure 3.3, the IR is change within the range [0.2, 5] according to the range parameter and will be different at each bag which means we may have IR smaller than or bigger than one in any bag. Since, in the original data the number of positive class instances are two and we may have a bag that has more than two instances, e.g., four in the first bag of Figure 3.3, the BagBoost algorithm followed an oversampling approach

in the first bag for positive class. Similarly, the original sample size of negative class is eight but the sample size of negative class in the first bag is three and hence the BagBoost algorithm followed a undersampling approach for negative class of first bag. In the bootstrapping phase both undersampling and oversampling from original data is used and hence, bootstrapping phase of BagBoost algorithm may be considered as an advanced hybridization of undersampling and oversampling algorithms that increases the diversity of bags.

Using G-mean score is important to classify the imbalanced bags accurately. Since the dataset at each bagging iteration is not also completely balanced in the proposed bootstrapping algorithm, traditional algorithms often failed to accurately classify the minority class examples. Even though the IR is decreased considerably, it is hard to build a valid classification model when the bags are imbalanced. To handle this problem at each iteration parameters of XGBoost algorithm are tuned according to the G-mean score which priorities the misclassification on minority class. In this way, the base classifiers, i.e., XGBoost algorithms, do not ignore the minority class and result well performing base learners that increases the overall performance of BagBoost algorithm.

Chapter 4

Experiments

In this chapter, we give the result of computational tests. The proposed algorithm, namely, BagBoost algorithm is compared to the state-of-the-art algorithms in the literature trained on all benchmark datasets and hence the results are compared.

4.1 Benchmark Datasets

There exist several benchmark datasets in the literature to compare the performance of classification algorithms in imbalanced domains. However, these algorithms are mostly developed for different application areas and hence proposed algorithms in the literature are tested with only some of these datasets. Accordingly, there is not a common set of benchmark datasets to be used for a fair comparison. Since the classification algorithms are highly prone to changes in the dataset, the computational results in the related studies cannot be used directly for comparison and there is need to train these algorithms in the same set of imbalanced datasets. In this regard, we use several datasets from different repositories that are used as benchmark datasets for imbalanced domains, namely, UCI, LIBSVM, and KDD to fairly compare the performances of classification algorithms. Table 4.1 shows the name, the target, the IR, and the sample size of each dataset. There are 24 datasets of varying sample sizes and IRs. The target stands for the class label for each dataset. Note that, IR varies from 9.3 to 42 and sample size varies from 450 to 145825 meaning that the chosen datasets are fair to conclude on classification performance of trained algorithms.

Table 4.1 Benchmark Datasets

Name	Target	IR	Sample Size	Name	Target	IR	Sample Size
satimage	4	9.3	6471	coil_2000	minority	16	9907
pen_digits	5	9.4	11008	arrhythmia	06	17	730
abalone	7	9.7	4187	solar_flare_m0	M>0	19	1421
sick_euthyroid	sick	9.8	3205	oil	minority	22	986
spectrometer	≥44	11	624	car_eval_4	goodness	26	1749
car_eval_34	goodness	12	1749	wine_quality	≤4	26	4909
isolet	A, B	12	8414	letter_img	Z	26	20016
us_crime	>0.65	12	2094	yeast_me2	ME2	28	1492
yeast_ml8	8	13	2520	webpage	minority	33	35080
scene	label	13	2701	ozone_level	data	34	2608
libras_move	1	14	450	mammography	minority	42	11189
thyroid_sick	sick	15	3824	protein_homo	minority	11	145825

4.2 Benchmark Algorithms

The algorithms selected to compare the performance of BagBoost algorithm according to their performances on imbalance domains are EasyEnsemble, BBC, BRF and RusBoost. EasyEnsemble is a double ensemble algorithm that is a hybridization of modified bagging and boosting algorithms. It consists of three parts: bootstrapping, training and aggregating. It ensures balanced bags while bootstrapping phase and uses AdaBoost to train on each bag at training phase. Each bag has a size equivalent to twice the size of the minority class and the minority classes are the same in each bag and hence the diversity among the bags is low compared to BagBoost algorithm. In training phase, the Adaboost algorithm is not tuned according to each bag and thus, the outputs of each base classifier, i.e., Adaboost classifiers, are similar.

EasyEnsemble is the only double ensemble algorithm used as a benchmark in this thesis. The rest of the algorithms that are compared to the BagBoost algorithm is from the family of ensemble algorithms. They all are hybridizations of data-level approaches with boosting or bagging ensembles. For this reason, there are only small changes in these algorithms.

BRF [71] is a decision tree-based ensemble classifier which has three phases (1) Bootstrap, (2) CART, and (3) Aggregate. (1) several bags are sample from original data with replacement. Each bag has a size equivalent to twice the size of minority class. (2) CART, which is a classification tree algorithm is induced without pruning the resulting

tree. (3) the results of each CART algorithm are aggregated. BRF created bags that have an IR of one in each iteration and hance, diversity of bags may be low.

RusBoost [62] is a hybridization of data-level methods and boosting algorithms. Mainly, it consists of two phases (1) RUS i.e., Random Under-sampling, and (2) AdaBoost. The RUS phase aims to under-sample from the majority class until the desired IR is ensured. In the AdaBoost phase Adaboost algorithm is trained on this new dataset. RUSBoost deletes several instances from the majority class randomly, which may cause loss of useful information that may be mandatory to define the majority class.

BBC [69] is a hybridization of bagging algorithm with data-level methods which differs from traditional bagging algorithms in the bootstrapping phase as BRF. The difference in the first phase is that BBC uses the RUS algorithm, i.e., randomly under-samples from majority class, and thus, the minority class is the same at each iteration. In other words, its under-samples from the majority class. In the second phase, it uses the Decision Tree Classifiers to learn from each bag. BBC randomly deletes several instances from negative class which may be mandatory to define majority class.

4.3 Experimental Results

This section presents the results of computational tests. Because the overall accuracy is not a good performance measure on imbalanced domains, F-measure and G-mean metrics are chosen to evaluate the performance of the algorithms.

The proposed algorithm BagBoost and benchmark algorithms are all coded using Python Programing Language. All algorithms are trained on 24 different benchmark datasets, and each dataset is split into 80% train and 20% test sets. Table 4.2 summarizes the average F-measures for all algorithms and datasets. The bold number for each dataset shows the best-performing algorithm for the dataset each row. The results show that the proposed algorithm outperforms benchmark algorithms except for three datasets, namely, *sick_euthyroid, yeast_ml8* and *ozone_level.* For two datasets, *sick_euthyroid* and *ozone_level* BagBoost performed slightly worse than the best performing algorithm, BBC. For *yeast_ml8* the performance of BagBoost is significantly worse than BBC. We think that this may be caused by class overlap or small sample size. The average of F-measure obtained by BagBoost is better than the average F-measure obtained by benchmark algorithms by about 33-53%.

Table 4.2 F-measures of Algorithms on Benchmark Datasets

Dataset	BagBoost	BBC	BRF	RusBoost	EasyEnsemble
satimage	**0.64198**	0.59307	0.54800	0.50663	0.54510
pen_digits	**0.99764**	0.91453	0.94798	0.84856	0.82191
abalone	**0.44853**	0.39464	0.39770	0.32897	0.37387
sick_euthyroid	0.80000	**0.81443**	0.75771	0.69795	0.74348
spectrometer	**0.84211**	0.61190	0.62381	0.64667	0.53357
car_eval_34	**0.88525**	0.74194	0.69305	0.82026	0.73871
isolet	**0.81679**	0.71199	0.65871	0.51268	0.70151
us_crime	**0.53488**	0.40399	0.42536	0.36087	0.40122
yeast_ml8	0.08696	**0.20359**	0.20142	0.07454	0.19391
scene	**0.28947**	0.17996	0.27216	0.15709	0.23903
libras_move	**0.66667**	0.23333	0.34000	0.50000	0.27333
thyroid_sick	**0.89583**	0.76150	0.68714	0.64034	0.72806
coil_2000	**0.19289**	0.13158	0.15190	0.12905	0.15875
arrhythmia	**0.88889**	0.50000	0.16333	0.21667	0.51667
solar_flare_m0	**0.21277**	0.19798	0.14376	0.04929	0.12560
oil	**0.18182**	0.18024	0.16639	0.15000	0.12824
car_eval_4	**0.76471**	0.58040	0.41192	0.72825	0.46654
wine_quality	**0.31461**	0.25693	0.21916	0.13051	0.19437
letter_img	**0.93197**	0.65811	0.63212	0.60822	0.60412
yeast_me2	**0.43478**	0.33690	0.26849	0.25746	0.24745
webpage	**0.78422**	0.36040	0.38022	0.19001	0.40425
ozone_level	0.17391	**0.17497**	0.13338	0.15111	0.13511
mammography	**0.76923**	0.44558	0.39634	0.40994	0.34754
protein_homo	**0.85825**	0.37635	0.40066	0.27755	0.34523
Average	**0.60059**	0.44851	0.41753	0.39136	0.41532

Table 4.3 illustrates the percentage differences between F-measure values of the proposed algorithm and those of benchmark algorithms. In nineteen benchmark datasets, the percentage change in F-measure value is greater than 10% and so, there is a significant improvement. The proposed algorithms perform better for hard classification tasks where IR is high. On the average, the percentage change in F-measure value is 38% higher than the second-best algorithm, BBC.

Table 4.3 Percentage Differences in F-measure

Dataset	BBC	BRF	RusBoost	EasyEnsemble
satimage	8.25	17.15	26.72	17.77
pen_digits	9.09	5.24	17.57	21.38
abalone	13.66	12.78	36.34	19.97
sick_euthyroid	-1.77	5.58	14.62	7.60
spectrometer	37.62	34.99	30.22	57.83
car_eval_34	19.32	27.73	7.92	19.84
isolet	14.72	24.00	59.32	16.43
us_crime	32.40	25.75	48.22	33.31
yeast_ml8	-57.29	-56.83	16.66	-55.15
scene	60.85	6.36	84.27	21.10
libras_move	185.72	96.08	33.33	143.91
thyroid_sick	17.64	30.37	39.90	23.04
coil_2000	46.60	26.98	49.47	21.51
arrhythmia	77.78	444.23	310.25	72.04
solar_flare_m0	7.47	48.00	331.67	69.40
oil	0.88	9.27	21.21	41.78
car_eval_4	31.76	85.65	5.01	63.91
wine_quality	22.45	43.55	141.06	61.86
letter_img	41.61	47.44	53.23	54.27
yeast_me2	29.05	61.94	68.87	75.70
webpage	117.60	106.25	312.73	93.99
ozone_level	-0.61	30.39	15.09	28.72
mammography	72.64	94.08	87.64	121.34
protein_homo	128.05	114.21	209.22	148.60
Average	38.14	55.88	84.19	49.17

Table 4.4 shows the G-mean scores of the algorithms for all datasets. The bold values show the best classification algorithm for the corresponding dataset. BagBoost performs better than benchmark algorithms for 14 of 24 benchmark datasets.

G-mean scores are higher than F-measure values because G-mean gives less importance to minority class than F-measure, which causes a decrease in the F-measure values resulting from the misclassification of minority class instances. Since it is crucial to detect minority class instances in many imbalanced classification tasks, F-measure may be chosen above the G-mean.

Table 4.4 G-mean of Algorithms on Benchmark Datasets

Dataset	BagBoost	BBC	BRF	RusBoost	EasyEnsemble
satimage	**0.8909**	0.8585	0.8806	0.7685	0.8671
pen_digits	**0.9919**	0.9558	0.9827	0.9303	0.9604
abalone	0.7849	0.7456	**0.8030**	0.6309	0.7839
sick_euthyroid	**0.9419**	0.9254	0.9279	0.8711	0.9346
spectrometer	**0.9742**	0.7524	0.8774	0.7051	0.8807
car_eval_34	0.9615	0.9451	0.9617	0.9557	**0.9681**
isolet	0.9529	0.9103	0.9330	0.7007	**0.9539**
us_crime	0.9118	0.7374	0.8286	0.5310	0.8142
yeast_ml8	0.5552	0.5547	0.6304	0.2067	0.6140
scene	0.6633	0.4722	0.7110	0.3685	0.6714
libras_move	0.8534	0.4897	0.5349	0.8000	0.4523
thyroid_sick	0.9652	0.9428	0.9098	0.8249	**0.9675**
coil_2000	0.6506	0.4783	0.5972	0.5200	0.6137
arrhythmia	0.9456	0.7882	0.3499	0.4881	0.7760
solar_flare_m0	0.7930	0.6105	0.5848	0.2266	0.5790
oil	**0.9018**	0.5078	0.5853	0.1694	0.4864
car_eval_4	**0.9676**	0.9406	0.9389	0.9545	0.9495
wine_quality	0.7733	0.6902	**0.7748**	0.4454	0.7628
letter_img	**0.9829**	0.9544	0.9671	0.8667	0.9664
yeast_me2	0.8325	0.7904	**0.8571**	0.5189	0.8496
webpage	**0.9382**	0.8743	0.9210	0.7525	0.9209
ozone_level	**0.7383**	0.6592	0.6609	0.3314	0.6783
mammography	0.9214	0.8985	**0.9242**	0.7833	0.9120
protein_homo	0.9519	0.9305	0.9380	0.8215	**0.9542**
Average	**0.8685**	0.7672	0.7950	0.6322	0.8049

Table 4.5 shows the differences between the G-mean scores of BagBoost algorithm and those of benchmark algorithms. On the average, EasyEnsemble is the closest to the BagBoost with a difference of 11.3%, i.e., BagBoost performs better than EasyEnsemble by about 11.3% with respect to G-mean score. BagBoost is slightly worse than the best performing benchmark algorithm for 6 of 10 datasets where BagBoost cannot outperform benchmark algorithms. In the 4 datasets there considered to be a significant change, namely, *abalone, yeast_m18, scene* and *yeast_me2* the percentage changes in G-mean score are 2.26%, 11.93%, 6.71% and 2.87%, respectively. According to these results, BagBoost algorithm performs better than the benchmark algorithms considering G-mean score.

Table 4.5 Percentage Differences in G-mean

Dataset	BBC	BRF	RusBoost	EasyEnsemble
satimage	3.78	1.17	15.93	2.74
pen_digits	3.78	0.94	6.63	3.28
abalone	5.27	-2.26	24.40	0.13
sick_euthyroid	1.78	1.51	8.13	0.78
spectrometer	29.47	11.03	38.16	10.61
car_eval_34	1.73	-0.02	0.61	-0.68
isolet	4.68	2.13	35.99	-0.10
us_crime	23.66	10.04	71.71	11.99
yeast_ml8	0.09	-11.93	168.62	-9.58
scene	40.48	-6.71	80.00	-1.21
libras_move	74.26	59.55	6.68	88.69
thyroid_sick	2.38	6.08	17.00	-0.24
coil_2000	36.03	8.94	25.12	6.01
arrhythmia	19.96	170.25	93.75	21.86
solar_flare_m0	29.89	35.61	249.89	36.96
oil	77.58	54.08	432.38	85.40
car_eval_4	2.87	3.05	1.38	1.90
wine_quality	12.04	-0.20	73.62	1.37
letter_img	2.99	1.63	13.41	1.70
yeast_me2	5.33	-2.87	60.42	-2.01
webpage	7.31	1.86	24.68	1.88
ozone_level	12.00	11.72	122.76	8.85
mammography	2.55	-0.31	17.64	1.03
protein_homo	2.30	1.48	15.87	-0.24
Average	16.76	14.87	66.87	11.30

Chapter 5

Conclusion and Future Prospects

5.1 Conclusions

In this study, we have addressed the class imbalance classification problem where majority class outnumbers minority class. Class imbalance problem is confronted in a variety of real-world classification tasks, namely, obstacle detection, fraud detection, medical diagnosis, spam detection, speech recognition, image processing and intrusion detection. The traditional classification algorithms cannot perform well for class imbalance domains because they are essentially overfitting the data to obtain a better overall accuracy and hence, ignore the minority class instances on training phase. Even though several algorithms in the literature developed for class imbalance problem, they are still not considered good enough according to the model performance on minority class instances. For this reason, we propose a novel classification algorithm that is a hybridization of modified bagging and boosting algorithms, namely, BagBoost. BagBoost algorithm differs from the previous ones in handling the bagging and boosting. While the previous state-of-the-art algorithms suffers from loss of information and/or overfitting BagBoost algorithm is developed to overcome these problems and hence, outperforms the state-of-the-art classification algorithms on most of the imbalanced benchmark datasets.

5.2 Societal Impact and Contribution to Global Sustainability

Classification algorithms are employed in a wide range of real-world problems such as obstacle detection, fraud detection, medical diagnosis, spam detection, speech recognition, image processing and intrusion detection. The BagBoost algorithm is a better

classifier than the others according to F-measure and G-mean performance metrics on imbalance domains. In this regard, in the various fields that class imbalance problem is confronted, this algorithm may contribute the sustainability of these fields. To exemplify, in health sector, may help for early detection of a cancer patient and hence, may save the life of the patient and decrease the cost of treatment. When applied in cybersecurity field, may contribute the sustainability of systems in banking, transportation, telecommunication, production, and so forth.

5.3 Future Prospects

In this study, the focus is on the classification of imbalanced datasets. Class imbalance problem is one of the most confronted issues in classification tasks. The currently proposed algorithms in the literature are unable to accurately detect minority class instances and hence, there is a need for accurate classifiers in imbalanced domains. We proposed a novel algorithm which is a hybridization of bagging and boosting classifiers that outperforms the state-of-the-art algorithms in the literature for most of the benchmark datasets. However, considering several other problems which may affect the performance of classification algorithms in imbalance domains, such as small sample size problems, class overlap problems, or within-class concepts [12], [13], the classification performance of BagBoost algorithm may be improved in other datasets. In the next stage, these problems may be incorporated in BagBoost algorithm.

BIBLIOGRAPHY

[1] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics 1943 5:4*, vol. 5, no. 4, pp. 115–133, Dec. 1943, doi: 10.1007/BF02478259.

[2] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychological Review*, vol. 65, no. 6, pp. 386–408, Nov. 1958, doi: 10.1037/H0042519.

[3] N. v. Chawla, N. Japkowicz, and A. Kotcz, “Editorial,” *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, Jun. 2004, doi: 10.1145/1007730.1007733.

[4] X. Y. Liu, J. Wu, and Z. H. Zhou, “Exploratory undersampling for class-imbalance learning,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 2, pp. 539–550, 2009, doi: 10.1109/TSMCB.2008.2007853.

[5] L. Nanni, C. Fantozzi, and N. Lazzarini, “Coupling different methods for overcoming the class imbalance problem,” *Neurocomputing*, vol. 158, pp. 48–61, Jun. 2015, doi: 10.1016/J.NEUCOM.2015.01.068.

[6] F. Rayhan, S. Ahmed, A. Mahbub, R. Jani, S. Shatabda, and D. M. Farid, “CUSBoost: Cluster-Based Under-Sampling with Boosting for Imbalanced Classification,” *2nd International Conference on Computational Systems and Information Technology for Sustainable Solutions, CSITSS 2017*, Aug. 2018, doi: 10.1109/CSITSS.2017.8447534.

[7] W. Lu, Z. Li, and J. Chu, “Adaptive Ensemble Undersampling-Boost: A novel learning framework for imbalanced data,” *Journal of Systems and Software*, vol. 132, pp. 272–282, Oct. 2017, doi: 10.1016/J.JSS.2017.07.006.

[8] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” 2018, doi: 10.5220/0006639801080116.

[9] J. Davis and M. Goadrich, “The Relationship Between Precision-Recall and ROC Curves,” *Proceedings of the 23rd international conference on Machine learning - ICML ’06*, doi: 10.1145/1143844.

[10] W. Siblini, J. Fréry, L. He-Guelton, F. Oblé, and Y. Q. Wang, “Master Your Metrics with Calibration,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12080 LNCS, pp. 457–469, 2020, doi: 10.1007/978-3-030-44584-3_36/FIGURES/7.

[11] F. Thabtah, S. Hammoud, F. Kamalov, and A. Gonsalves, “Data imbalance in classification: Experimental evaluation,” *Information Sciences*, vol. 513, pp. 429–441, Mar. 2020, doi: 10.1016/J.INS.2019.11.004.

[12] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, “Cost-sensitive boosting for classification of imbalanced data,” *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, Dec. 2007, doi: 10.1016/J.PATCOG.2007.04.009.

[13] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, “A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches,” *IEEE Transactions on Systems, Man and Cybernetics Part C:*

Applications and Reviews, vol. 42, no. 4, pp. 463–484, Jul. 2012, doi: 10.1109/TSMCC.2011.2161285.

[14] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, pp. 1–30, Dec. 2018, doi: 10.1186/S40537-018-0151-6/TABLES/5.

[15] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, Nov. 2016, doi: 10.1007/S13748-016-0094-0/TABLES/1.

[16] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, Oct. 2018, doi: 10.1016/J.NEUNET.2018.07.011.

[17] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1–54, Dec. 2019, doi: 10.1186/S40537-019-0192-5/TABLES/18.

[18] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from Imbalanced Data Sets*. Springer International Publishing, 2018. doi: 10.1007/978-3-319-98074-4.

[19] I. Tomek, "TWO MODIFICATIONS OF CNN.," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-6, no. 11, pp. 769–772, 1976, doi: 10.1109/TSMC.1976.4309452.

[20] M. Kubat, R. Holte, and S. Matwin, "Learning when negative examples abound," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1224, pp. 146–153, 1997, doi: 10.1007/3-540-62858-4_79.

[21] S. J. Yen and Y. S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5718–5727, Apr. 2009, doi: 10.1016/J.ESWA.2008.06.108.

[22] K. Yoon and S. Kwek, "An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics," *Proceedings - HIS 2005: Fifth International Conference on Hybrid Intelligent Systems*, vol. 2005, pp. 303–308, 2005, doi: 10.1109/ICHIS.2005.23.

[23] N. v. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/JAIR.953.

[24] J. Luengo, A. Fernández, S. García, and F. Herrera, "Addressing data complexity for imbalanced data sets: Analysis of SMOTE-based oversampling and evolutionary undersampling," *Soft Computing*, vol. 15, no. 10, pp. 1909–1936, Oct. 2011, doi: 10.1007/S00500-010-0625-8/TABLES/16.

[25] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, "Data Level Preprocessing Methods," *Learning from Imbalanced Data Sets*, pp. 79–121, 2018, doi: 10.1007/978-3-319-98074-4_5.

[26] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, "Algorithm-Level Approaches," *Learning from Imbalanced Data Sets*, pp. 123–146, 2018, doi: 10.1007/978-3-319-98074-4_6.

[27] Gang Wu and E. Y. Chang, "Aligning Boundary in Kernel Space for Learning Imbalanced Dataset," pp. 265–272, Mar. 2005, doi: 10.1109/ICDM.2004.10106.

[28] C. Y. Yang, J. S. Yang, and J. J. Wang, "Margin calibration in SVM class-imbalanced learning," *Neurocomputing*, vol. 73, no. 1–3, pp. 397–411, Dec. 2009, doi: 10.1016/J.NEUCOM.2009.08.006.

[29] H. Yu, C. Mu, C. Sun, W. Yang, X. Yang, and X. Zuo, “Support vector machine-based optimized decision threshold adjustment strategy for classifying imbalanced data,” *Knowledge-Based Systems*, vol. 76, pp. 67–78, Mar. 2015, doi: 10.1016/J.KNOSYS.2014.12.007.

[30] M. Pérez-Ortiz, P. A. Gutiérrez, J. Sánchez-Monedero, and C. Hervás-Martínez, “A Study on Multi-Scale Kernel Optimisation via Centered Kernel-Target Alignment,” *Neural Processing Letters*, vol. 44, no. 2, pp. 491–517, Oct. 2016, doi: 10.1007/S11063-015-9471-0.

[31] M. R. Smith, T. Martinez, and C. Giraud-Carrier, “An instance level analysis of data complexity,” *Machine Learning*, vol. 95, no. 2, pp. 225–256, 2014, doi: 10.1007/S10994-013-5422-Z.

[32] W. Lee, C. H. Jun, and J. S. Lee, “Instance categorization by support vector machines to adjust weights in AdaBoost for imbalanced data classification,” *Information Sciences*, vol. 381, pp. 92–103, Mar. 2017, doi: 10.1016/J.INS.2016.11.014.

[33] X. Wang, X. Liu, and S. Matwin, “A distributed instance-weighted SVM algorithm on large-scale imbalanced datasets,” *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, pp. 45–51, Jan. 2015, doi: 10.1109/BIGDATA.2014.7004467.

[34] X. Wang, X. Liu, S. Matwin, and N. Japkowicz, “Applying instance-weighted support vector machines to class imbalanced datasets,” *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, pp. 112–118, 2014, doi: 10.1109/BIGDATA.2014.7004364.

[35] F. Zhu, J. Yang, J. Gao, and C. Xu, “Extended nearest neighbor chain induced instance-weights for SVMs,” *Pattern Recognition*, vol. 60, pp. 863–874, Dec. 2016, doi: 10.1016/J.PATCOG.2016.07.012.

[36] T. Imam, K. M. Ting, and J. Kamruzzaman, “z-SVM: An SVM for Improved Classification of Imbalanced Data,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4304 LNAI, pp. 264–273, 2006, doi: 10.1007/11941439_30.

[37] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, “Cost-Sensitive Learning,” *Learning from Imbalanced Data Sets*, pp. 63–78, 2018, doi: 10.1007/978-3-319-98074-4_4.

[38] S. Datta and S. Das, “Near-Bayesian Support Vector Machines for imbalanced data classification with equal or unequal misclassification costs,” *Neural Networks*, vol. 70, pp. 39–52, Oct. 2015, doi: 10.1016/J.NEUNET.2015.06.005.

[39] Z. H. Zhou and X. Y. Liu, “Training cost-sensitive neural networks with methods addressing the class imbalance problem,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, pp. 63–77, Jan. 2006, doi: 10.1109/TKDE.2006.17.

[40] B. Krawczyk and M. Woźniak, “Cost-sensitive neural network with ROC-based moving threshold for imbalanced classification,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9375 LNCS, pp. 45–52, 2015, doi: 10.1007/978-3-319-24834-9_6/TABLES/3.

[41] F. Li, X. Zhang, X. Zhang, C. Du, Y. Xu, and Y. C. Tian, “Cost-sensitive and hybrid-attribute measure multi-decision tree over imbalanced data sets,” *Information Sciences*, vol. 422, pp. 242–256, Jan. 2018, doi: 10.1016/J.INS.2017.09.013.

[42] C. Qiu, L. Jiang, and C. Li, “Randomly selected decision tree for test-cost sensitive learning,” *Applied Soft Computing*, vol. 53, pp. 27–33, Apr. 2017, doi: 10.1016/J.ASOC.2016.12.047.

[43] H. Zhao, X. J. Li, Z. L. Xu, and W. Zhu, “Cost-sensitive decision tree with probabilistic pruning mechanism,” *Proceedings - International Conference on Machine Learning and Cybernetics*, vol. 1, pp. 81–87, Nov. 2015, doi: 10.1109/ICMLC.2015.7340902.

[44] J. A. Sáez, B. Krawczyk, and M. Woźniak, “Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets,” *Pattern Recognition*, vol. 57, pp. 164–178, Sep. 2016, doi: 10.1016/J.PATCOG.2016.03.012.

[45] B. Krawczyk, L. Torgo, P. Branco, and N. Moniz, “Influence of minority class instance types on SMOTE imbalanced data oversampling Przemyss law Skryjomski,” *Proceedings of Machine Learning Research*, vol. 74, pp. 7–21, 2017.

[46] S. Wang, Z. Li, W. Chao, and Q. Cao, “Applying adaptive over-sampling technique based on data density and cost-sensitive SVM to imbalanced learning,” *Proceedings of the International Joint Conference on Neural Networks*, 2012, doi: 10.1109/IJCNN.2012.6252696.

[47] W. Huang, G. Song, M. Li, W. Hu, and K. Xie, “Adaptive Weight Optimization for Classification of Imbalanced Data,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8261 LNCS, pp. 546–553, 2013, doi: 10.1007/978-3-642-42057-3_69.

[48] X. C. Yin, K. Huang, and H. W. Hao, “DE2: Dynamic ensemble of ensembles for learning nonstationary data,” *Neurocomputing*, vol. 165, pp. 14–22, Oct. 2015, doi: 10.1016/J.NEUCOM.2014.06.092.

[49] L. Breiman, “Bagging predictors,” *Machine Learning 1996 24:2*, vol. 24, no. 2, pp. 123–140, 1996, doi: 10.1007/BF00058655.

[50] R. E. Schapire, “The strength of weak learnability,” *Machine Learning 1990 5:2*, vol. 5, no. 2, pp. 197–227, Jun. 1990, doi: 10.1007/BF00116037.

[51] J. F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, and L. I. Kuncheva, “Diversity techniques improve the performance of the best imbalance learning ensembles,” *Information Sciences*, vol. 325, pp. 98–117, Dec. 2015, doi: 10.1016/J.INS.2015.07.025.

[52] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, “Ensemble Learning,” *Learning from Imbalanced Data Sets*, pp. 147–196, 2018, doi: 10.1007/978-3-319-98074-4_7.

[53] H. Masnadi-Shirazi and N. Vasconcelos, “Cost-sensitive boosting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 294–309, 2011, doi: 10.1109/TPAMI.2010.71.

[54] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, “AdaCost: Misclassification Cost-sensitive Boosting”.

[55] M. v. Joshi, V. Kumar, and R. C. Agarwal, “Evaluating boosting algorithms to classify rare classes: Comparison and improvements,” *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 257–264, 2001, doi: 10.1109/ICDM.2001.989527.

[56] B. X. Wang and N. Japkowicz, “Boosting support vector machines for imbalanced data sets,” *Knowledge and Information Systems 2009 25:1*, vol. 25, no. 1, pp. 1–20, Mar. 2009, doi: 10.1007/S10115-009-0198-Y.

[57] K. Li, X. Kong, Z. Lu, L. Wenyin, and J. Yin, “Boosting weighted ELM for imbalanced learning,” *Neurocomputing*, vol. 128, pp. 15–21, Mar. 2014, doi: 10.1016/J.NEUCOM.2013.05.051.

[58] B. Krawczyk, M. Woźniak, and G. Schaefer, “Cost-sensitive decision tree ensembles for effective imbalanced classification,” *Applied Soft Computing*, vol. 14, no. PART C, pp. 554–562, Jan. 2014, doi: 10.1016/J.ASOC.2013.08.014.

[59] M. Zięba and J. M. Tomczak, “Boosted SVM with active learning strategy for imbalanced data,” *Soft Computing*, vol. 19, no. 12, pp. 3357–3368, Dec. 2015, doi: 10.1007/S00500-014-1407-5/TABLES/5.

[60] N. v. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, “SMOTEBoost: Improving Prediction of the Minority Class in Boosting,” *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, vol. 2838, pp. 107–119, 2003, doi: 10.1007/978-3-540-39804-2_12.

[61] S. Hu, Y. Liang, L. Ma, and Y. He, “MSMOTE: Improving classification performance when training data is imbalanced,” *2nd International Workshop on Computer Science and Engineering, WCSE 2009*, vol. 2, pp. 13–17, 2009, doi: 10.1109/WCSE.2009.756.

[62] C. Seiffert, T. M. Khoshgoftaar, J. van Hulse, and A. Napolitano, “RUSBoost: A hybrid approach to alleviating class imbalance,” *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 40, no. 1, pp. 185–197, Jan. 2010, doi: 10.1109/TSMCA.2009.2029559.

[63] M. Galar, A. Fernández, E. Barrenechea, and F. Herrera, “EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling,” *Pattern Recognition*, vol. 46, no. 12, pp. 3460–3471, Dec. 2013, doi: 10.1016/J.PATCOG.2013.05.006.

[64] H. Wei, B. Sun, and M. Jing, “BalancedBoost: A hybrid approach for real-time network traffic classification,” *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, Sep. 2014, doi: 10.1109/ICCCN.2014.6911833.

[65] S. Chen, H. He, and E. A. Garcia, “RAMOBoost: Ranked minority oversampling in boosting,” *IEEE Transactions on Neural Networks*, vol. 21, no. 10, pp. 1624–1642, Oct. 2010, doi: 10.1109/TNN.2010.2066988.

[66] N. García-Pedrajas and C. García-Osorio, “Boosting for class-imbalanced datasets using genetically evolved supervised non-linear projections,” *Progress in Artificial Intelligence*, vol. 2, no. 1, pp. 29–44, Mar. 2013, doi: 10.1007/S13748-012-0028-4/TABLES/4.

[67] J. F. Díez-Pastor, J. J. Rodríguez, C. García-Osorio, and L. I. Kuncheva, “Random Balance: Ensembles of variable priors classifiers for imbalanced data,” *Knowledge-Based Systems*, vol. 85, pp. 96–111, Sep. 2015, doi: 10.1016/J.KNOSYS.2015.04.022.

[68] S. Wang and X. Yao, “Diversity analysis on imbalanced data sets by using ensemble models,” *2009 IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2009 - Proceedings*, pp. 324–331, 2009, doi: 10.1109/CIDM.2009.4938667.

[69] “BalancedBaggingClassifier — Version 0.9.1.” https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.BalancedBaggingClassifier.html#imblearn.ensemble.BalancedBaggingClassifier (accessed Aug. 11, 2022).

[70] S. Hido, H. Kashima, and Y. Takahashi, “Roughly balanced bagging for imbalanced data,” *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 2, no. 5–6, pp. 412–426, Dec. 2009, doi: 10.1002/SAM.10061.
[71] C. Chen and A. Liaw, “Using Random Forest to Learn Imbalanced Data”.
[72] “(PDF) An Empirical Evaluation of Bagging and Boosting.” https://www.researchgate.net/publication/2262068_An_Empirical_Evaluation_of_Bagging_and_Boosting (accessed Aug. 11, 2022).
[73] E. Y. Chang, B. Li, G. Wu, and K. Goh, “Statistical Learning for Effective Visual Information Retrieval,” *IEEE International Conference on Image Processing*, vol. 3, pp. 609–612, 2003, doi: 10.1109/ICIP.2003.1247318.
[74] D. Tao, X. Tang, and X. Wu, “Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1088–1099, Jul. 2006, doi: 10.1109/TPAMI.2006.134.
[75] L. Cen, “Classifying imbalanced data using a Bagging Ensemble Variation (BEV),” *Proceedings of the Annual Southeast Conference*, vol. 2007, pp. 203–208, 2007, doi: 10.1145/1233341.1233378.
[76] M. A. Tahir, J. Kittler, and F. Yan, “Inverse random under sampling for class imbalance problem and its application to multi-label classification,” *Pattern Recognition*, vol. 45, no. 10, pp. 3738–3750, Oct. 2012, doi: 10.1016/J.PATCOG.2012.03.014.
[77] Y. Park and J. Ghosh, “Ensembles of α-trees for imbalanced classification problems,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 131–143, Jan. 2014, doi: 10.1109/TKDE.2012.255.
[78] B. Krawczyk and G. Schaefer, “An improved ensemble approach for imbalanced classification problems,” *SACI 2013 - 8th IEEE International Symposium on Applied Computational Intelligence and Informatics, Proceedings*, pp. 423–426, 2013, doi: 10.1109/SACI.2013.6609011.
[79] J. Błaszczyński, M. Deckert, J. Stefanowski, and S. Wilk, “Integrating Selective Pre-processing of Imbalanced Data with Ivotes Ensemble,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6086 LNAI, pp. 148–157, 2010, doi: 10.1007/978-3-642-13529-3_17.
[80] S. Gónzalez, S. García, M. Lázaro, A. R. Figueiras-Vidal, and F. Herrera, “Class Switching according to Nearest Enemy Distance for learning from highly imbalanced data-sets,” *Pattern Recognition*, vol. 70, pp. 12–24, Oct. 2017, doi: 10.1016/J.PATCOG.2017.04.028.
[81] S. Vluymans, I. Triguero, C. Cornelis, and Y. Saeys, “EPRENNID: An evolutionary prototype reduction based ensemble for nearest neighbor classification of imbalanced data,” *Neurocomputing*, vol. 216, pp. 596–610, Dec. 2016, doi: 10.1016/J.NEUCOM.2016.08.026.
[82] G. I. Webb, “MultiBoosting: A Technique for Combining Boosting and Wagging,” *Machine Learning 2000 40:2*, vol. 40, no. 2, pp. 159–196, Aug. 2000, doi: 10.1023/A:1007659514849.
[83] Y. Freund and R. E. Schapire, “Experiments with a New Boosting Algorithm,” *Proceedings of the 13th International Conference on Machine Learning*, pp. 148–156, 1996, doi: 10.1.1.133.1040.
[84] J. H. Friedman, “On Bias, Variance, 0/1—Loss, and the Curse-of-Dimensionality,” *Data Mining and Knowledge Discovery 1997 1:1*, vol. 1, no. 1, pp. 55–77, 1997, doi: 10.1023/A:1009778005914.

[85] J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors),” *https://doi.org/10.1214/aos/1016218223*, vol. 28, no. 2, pp. 337–407, Apr. 2000, doi: 10.1214/AOS/1016218223.

[86] E. Bauer, P. Chan, S. Stolfo, and D. Wolpert, “An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants,” *Machine Learning 1999 36:1*, vol. 36, no. 1, pp. 105–139, 1999, doi: 10.1023/A:1007515423169.

[87] J. H. Friedman, “Stochastic gradient boosting,” *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, Feb. 2002, doi: 10.1016/S0167-9473(01)00065-2.

[88] G. I. Webb and Z. Zheng, “Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 8, pp. 980–991, Aug. 2004, doi: 10.1109/TKDE.2004.29.

[89] L. Breiman, “Using Iterated Bagging to Debias Regressions,” *Machine Learning 2001 45:3*, vol. 45, no. 3, pp. 261–277, Dec. 2001, doi: 10.1023/A:1017934522171.

[90] L. Nanni and A. Franco, “Reduced Reward-punishment editing for building ensembles of classifiers,” *Expert Systems with Applications*, vol. 38, no. 3, pp. 2395–2400, Mar. 2011, doi: 10.1016/J.ESWA.2010.08.028.

CURRICULUM VITAE

2013 – 2018	B.Sc., Industrial Engineering, Abdullah Gül University, Kayseri, TURKEY
2018 – Present	Present M.Sc., Abdullah Gül University, Kayseri, TURKEY
2019 – Present	Present Reasearch Assistant, Industrial Engineering, Abdullah Gül University, Kayseri, TURKEY

SELECTED PUBLICATIONS AND PRESENTATIONS

C1) S. Gören, İ. Gülbahar, M. Pinar, Statistical approach for table tennis athletes' success (Jul. 2018).