

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

PERFORMANCE COMPARISON OF DATA LEVEL
IMBALANCED DATA HANDLING TECHNIQUES
ON A BINARY CLASSIFICATION PROBLEM

DERYA NUR COŞAR

A THESIS OF MASTER OF SCIENCE
DEPARTMENT OF MATHEMATICS

ADVISOR: PROF. DR. NURİ ÇELİK

SEPTEMBER 2024

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

**PERFORMANCE COMPARISON OF DATA LEVEL
IMBALANCED DATA HANDLING TECHNIQUES
ON A BINARY CLASSIFICATION PROBLEM**

DERYA NUR COŞAR

A THESIS OF MASTER OF SCIENCE
DEPARTMENT OF MATHEMATICS

ADVISOR: PROF. DR. NURİ ÇELİK

SEPTEMBER 2024

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

İKİLİ SINIFLANDIRMA PROBLEMİ ÜZERİNDE VERİ
DÜZEYİNDE DENGESİZ VERİ İŞLEME TEKNİKLERİNİN
PERFORMANS KARŞILAŞTIRMASI

DERYA NUR COŞAR

YÜKSEK LİSANS TEZİ
MATEMATİK ANABİLİM DALI

DANIŞMAN: PROF. DR. NURİ ÇELİK

EYLÜL 2024



MASTER of SCIENCE JURY APPROVAL FORM

A thesis submitted by Derya Nur, Coşar, defended on 16/09/2024 before the jury formed with the 04/07/2024 date and 2024/34 numbered decision of the GTU Graduate Administration Board, has been accepted as a MASTER of SCIENCE thesis in the Department of Mathematics.

JURY

MEMBER

(THESIS ADVISOR) : Prof. Dr. Nuri ÇELİK

MEMBER : Assoc. Prof. Feray HACİVELİOĞLU

MEMBER : Assoc. Prof. Nevin GÜLER DİNÇER

APPROVAL

Gebze Technical University Graduate Administration Board
...../...../..... date and/..... numbered decision.

SIGNATURE/SEAL

Dedicated to my loving family, supportive friends, and invaluable advisor, whose guidance and encouragement made this work possible.

ABSTRACT

In machine learning studies imbalanced data is a commonly encountered problem. Especially in classification problems, imbalanced data negatively affect the generalization power of the model. Problems with this type of data should be studied with extra caution since imbalanced data mostly comes from a critical domain such as banking and healthcare. The research focuses on a specific binary classification problem with a significant imbalance between the positive and negative classes. A random forest algorithm is chosen as the base model for its robustness to imbalanced data. To address the imbalance and enhance the model's ability to identify the minority class, five data-level resampling techniques are employed: Random Oversampling, Random Undersampling, SMOTE (Synthetic Minority Oversampling Technique), Tomek Links Undersampling and NearMiss Algorithm. The thesis evaluates the performance of the base model with original dataset and the models trained with each resampled dataset. The evaluation metrics used are precision, recall, and F1-score, which provide a comprehensive assessment of model performance, particularly in imbalanced settings. Each resampling method were evaluated with these three metrics and the comparison is given in the “Results” section. The dataset subject to this study is generated from a tele-marketing campaign of a Portuguese bank. Target variable holds the information of whether the related customer subscribed a term-deposit. This thesis contributes valuable insights into the application of data resampling techniques for binary classification with imbalanced data. The comparative analysis of the techniques with a random forest model provides practical guidance for researchers and machine learning developers facing similar challenges. By understanding the strengths and weaknesses of each approach, more informed decisions is possible to make to improve the performance of their binary classification models in real-world scenarios with imbalanced data.

Keywords: Machine Learning, Imbalanced Data, Resampling, Classification Problems, Random Forest Algorithm.

ÖZET

Makine öğrenimi çalışmalarında, dengesiz veri yaygın olarak karşılaşılan bir sorundur. Özellikle sınıflandırma problemlerinde, dengesiz veriler modelin genelleme gücünü olumsuz etkiler. Bu tür veriler genellikle bankacılık ve sağlık gibi kritik alanlardan geldiği için, dengesiz veri içeren problemler üzerinde ekstra özenle çalışılmalıdır. Araştırma, pozitif ve negatif sınıflar arasında önemli bir dengesizlik bulunan belirli bir ikili sınıflandırma problemi üzerine odaklanmaktadır. Dengesiz verilere karşı dayanıklılığı nedeniyle temel model olarak rastgele orman algoritması seçilmiştir. Dengesizliği gidermek ve modelin azınlık sınıfını tanıma yeteneğini artırmak amacıyla beş veri düzeyinde yeniden örnekleme tekniği kullanılmıştır: Rastgele Aşırı Örnekleme, Rastgele Eksik Örnekleme, SMOTE (Sentetik Azınlık Aşırı Örnekleme Tekniği), Tomek Bağlantıları Eksik Örnekleme ve NearMiss Algoritması. Tezde, temel modelin orijinal veri setiyle ve yeniden örnekleme her veri setiyle eğitilen modellerin performansı değerlendirilmiştir. Değerlendirme ölçütleri olarak hassasiyet, duyarlılık ve F1 skoru kullanılmıştır; bu ölçütler, özellikle dengesiz ayarlarda model performansının kapsamlı bir değerlendirmesini sağlar. Her yeniden örnekleme yöntemi bu üç ölçütle değerlendirilmiş ve karşılaştırma “Sonuçlar” bölümünde verilmiştir. Bu çalışmaya konu olan veri seti, bir Portekiz bankasının tele-pazarlama kampanyasından elde edilmiştir. Hedef değişken, ilgili müşterinin vadeli mevduata abone olup olmadığını göstermektedir. Bu tez, dengesiz verilerle ikili sınıflandırma için veri yeniden örnekleme tekniklerinin uygulanmasına ilişkin değerli bilgiler sunmaktadır. Rastgele orman modeliyle tekniklerin karşılaştırmalı analizi, benzer zorluklarla karşılaşan araştırmacılar ve makine öğrenimi geliştiricileri için pratik rehberlik sağlar. Her yaklaşımın güçlü ve zayıf yönlerini anlayarak, dengesiz veriye sahip gerçek dünya senaryolarında ikili sınıflandırma modellerinin performansını iyileştirmek için daha bilinçli kararlar vermek mümkündür.

Anahtar Kelimeler: Makine Öğrenmesi, Dengesi Veri Seti, Yeniden Örnekleme, Sınıflandırma Problemleri, Rastgele Orman Algoritması.

TABLE OF CONTENTS

	<u>Sayfa</u>
ABSTRACT	vi
ÖZET	vii
TABLE OF CONTENTS	viii
LIST OF SYMBOLS AND ABBREVIATIONS	x
LIST OF FIGURES	xi
LIST OF TABLES	xii
1.INTRODUCTION	1
1.1 Literature Review	4
2.PURPOSE OF THE STUDY	10
3.MACHINE LEARNING	11
3.1 Types of Machine Learning Algorithms	11
3.2 Mathmematical Problem Definitions of Machine Learning Algorithms	15
4.IMBALANCED CLASSIFICATION	17
4.1 Random Over Sampling	19
4.2 Random Under Sampling	20
4.3 Synthetic Minority Over Sampling Technique (SMOTE)	21
4.4 Under Sampling with Tomek Links	24
4.5 Near Miss Algorithm	25
5.APPLICATION	27
5.1 Explanatory Data Analysis (EDA) and Data Preprocess	27
5.2 Base Model	36
5.3 Random Over Sampling	41
5.4 Random Under Sampling	42
5.5 Under Sampling with Tomek Links	43
5.6 Over Sampling with SMOTE	44
5.7 Under Sampling with Near Miss Algorithm	45
6.RESULTS	48

7.CONCLUSION	50
REFERENCES	51
BIOGRAPGY	54
APPENDICES	55

LIST OF SYMBOLS AND ABBREVIATIONS

ML	: Machine Learning
AI	: Artificial Intelligence
k-NN	: k-Nearest Neighbour
SMOTE	: Synthetic Minority Over Sampling Technique
LMS	: Least Mean Squares
NLP	: Natural Language Processing
BERT	: Bi-Directional Transformers for Language Teaching
LLM	: Large Language Models
LDA	: Latent Dirichlet Allocation
AUC	: Area Under Curve
ROC	: Receiving Operating Curve
ADASYN	: Adaptive Synthetic Sampling
CNN	: Condensed Nearest Neighbours

LIST OF FIGURES

	<u>Page</u>
Figure 4.1: Random over sampling process	19
Figure 4.2: Random under sampling process	20
Figure 4.3: Process of Synthetic Minority Oversampling Technique	23
Figure 4.4: Illustration of a Tomek Links	24
Figure 4.5: Illustration of Near Miss Algorithm	26
Figure 5.1: Histogram of variable “age”	27
Figure 5.2: Histogram of variable “campaign”	36
Figure 5.3: Histogram of variable “previous”	36
Figure 5.4: Box-plot graphs of index variables	37
Figure 5.5: Box-plot of variable “campaign”	39
Figure 5.6: Confusion matrix of the base model	44
Figure 5.7: Confusion matrix of the model 1	45
Figure 5.8: Confusion matrix of the model 2	46
Figure 5.9: Confusion matrix of the model 3	47
Figure 5.10: Confusion matrix of the model 4	48
Figure 5.11: Confusion matrix of the model 5	49
Figure 5.12: Confusion matrix of the model 6	50
Figure 5.13: Confusion matrix of the model 7	51

LIST OF TABLES

	<u>Page</u>
Table 5.1: Distribution of variable “job”	27
Table 5.2: Distribution of variable “marital status”	32
Table 5.3: Distribution of variable “education”	32
Table 5.4: Distribution of variable “default”	33
Table 5.5: Distribution of variable “housing”	33
Table 5.6: Distribution of variable “loan”	34
Table 5.7: Distribution of variable “contact”	34
Table 5.8: Distribution of variable “month”	35
Table 5.9: Distribution of variable “day_of_week”	35
Table 5.10: Distribution of variable “poutcome”	37
Table 5.11: Distribution of variable “y”	38
Table 5.12: Evaluation metrics of the base model	43
Table 5.13: Evaluation metrics of the model 1	45
Table 5.14: Evaluation metrics of the model 2	46
Table 5.15: Evaluation metrics of the model 3	47
Table 5.16: Evaluation metrics of the model 4	48
Table 5.17: Evaluation metrics of the model 5	49
Table 5.18: Evaluation metrics of the model 6	49
Table 5.19: Evaluation metrics of the model 7	50
Table 6.1: Precision scores for resampled datasets	52
Table 6.2: Recall scores for resampled datasets	52
Table 6.3: F1-Scores for resampled datasets	53

1. INTRODUCTION

Machine learning is a branch of artificial intelligence that enables computers to learn from data and make decisions. Classification problems aim to categorize data into specific classes. However, in some datasets, there may be a significant imbalance between classes. This can negatively impact the performance of machine learning models. In our master's thesis, we conducted various experiments using the Random Forest algorithm and different techniques to address this imbalance problem. Our results demonstrated that methods such as NearMiss version 2 and SMOTE are effective in mitigating the classification issues caused by imbalanced data.

Machine learning (ML) is an ensemble of specialized algorithms designed by mathematicians and computer scientists to execute specific tasks. What makes these algorithms special is their inherent ability to process large amount of input data and yield substantial and actionable outputs with minimal human intervention. These outputs are included to decision making processes across diverse sectors to gain insight and to create value. In today's world it is common to see ML solutions in many different fields and with emerging technology more and more problems are being solved by ML. For instance in finance, banks benefit predictive power of machine learning algorithms to forecast pivotal market indexes with high accuracy. Moreover, fraud attempts can be prevented by the models desinged to detect abnormal card transactions. Likewise, in the domain of healthcare ML-driven diagnostic tools such as models trained with x-ray images or blood test results of thousands of patients empowers doctors to diagnose rare diseases. There are many more examples to solutions developed by ML algorithms, they are studied in two main categories as regression and classification.

Regression is a statistical technique to analyze the relationship between independent variable and dependent variable(s). The dependent variable is explained by the independent variable(s) and forecasted for future values therefore it is a very useful tool. Regression is used in a variety of fields and lays the foundation of many modern algorithms. A financial analyst can use regression to explain the factors that is effective

on customers' spending behavior. In this case, age of a customer, income level and current economical state etc. are independent variables all impacting the spending behavior, which is the dependent variable of the study. The most common type of regression is "simple linear regression" where there is one dependent variable and independent variable and the relationship between them is linear. In many real world applications of regression the relationship between the dependent variable and the independent variable is not linear and there are multiple independent variables. The mathematical formula of simple linear regression is given by

$$Y = B_0 + B_1X \quad (1.1)$$

where Y_i is dependent variable, B_1 is slope/coefficient and X_i is the independent variable. In classification problems, it is aimed to correctly classify a group of data points into predefined categories or classes. If there are two categories then the problem is called "binary classification problem" and if there are more than two categories then it is called "multinomial classification problem." There is also another type of classification problem called "multi-label classification problem", in which the data points to be classified may belong to more than one class. For example, when categorizing movies, it is possible to put a movie into more than one category, for example, both romantic and comedy. All types of classification problems have quite a lot of applications in the literature. With sufficient amount of input data it is possible to build effective classification models yet the distribution of this data influences the model's performance. The optimal training dataset for a classification model contains a balanced representation of instances from each class enhancing model's ability to generalize across different classes. It is important to consider the distribution of data samples among the classes. The assumption is to have a balanced distribution since it is the case where the model is able learn from all the possible classes. If the assumption is not met then it is needed to study the algorithm a little differently or modify the data with suitable techniques. The least represented class in dataset is called "minority class" and similarly the highly represented class is called "majority class". If there is a serious imbalance between the minority class and the majority class, it is a common practice to use a robust classification algorithm or perform resampling on the data to

correct the imbalance. Depending on the nature of the data, sometimes a combination of imbalanced data solutions is applied. In addition, the settings and the way of application may also differ. In general, these type of problems is known as imbalanced classification problem in machine learning literature. Solutions developed to overcome the imbalanced classification problem is mainly grouped under two categories: algorithm-level techniques and data-level techniques.

At the algorithm level, techniques modify known classification algorithms to perceive the minority class as more emphasized or lower the error rate caused by incorrect predictions. On the other hand, data-level techniques are easier to understand because they work with a simpler strategy. Basically, they resample the data to increase the number of minority class samples, or remove some samples from the majority class, these operations are called “oversampling” and “under-sampling”, respectively. It is expected that changing the occurrence of data points in the dataset will affect the operation of the algorithm, as the existing information is duplicated or removed. For instance, under-sampling may cause loss of critical information to the model. Therefore, it is recommended to use imbalanced data solutions at the data level, taking into account their disadvantages.

There are other more complex undersampling and oversampling techniques, namely, Under-sampling with Tomek Links, Synthetic Minority Oversampling Technique (SMOTE) and NearMiss Algorithm. Under-sampling with Tomek Links makes use of Tomek Links to select data points to remove from dataset. Tomek Links are the elements that is helpful to define the boundary between classes and is defined as the points that are not closest to each other belonging to the same class. SMOTE is another powerful technique for increasing the size of the minority class in an imbalanced dataset by synthetically creating samples. The difference between random oversampling and SMOTE is that smote does not perform random resampling, but instead uses the k-NN (k-Nearest Neighbor) algorithm to select the resampling set. The last technique to be examined in this thesis is the NearMiss Algorithm, which is an oversampling technique. It selects the data points closest to the majority class from the minority class and overexamples them.

In general, five main data-level imbalanced data solution techniques have been studied. Two of them are based on under-sampling of the majority class, and the rest are based on over-sampling of the minority class. Detailed evaluation of the techniques is carried out on a binary classification problem with a moderately imbalanced dataset.

1.1 Literature Review

Over the years, machine learning has been an interesting topic to work on for mathematicians and computer scientists alike. In the literature, there are many successful studies that form the basis of today's latest development in artificial intelligence. The history of machine learning studies goes back to 1940-1950s.

One of them is called “Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain” (Rosenblatt, 1958). Rosenblatt proposed the basic concept of neural networks in his work: the perceptron. Neural networks are a different branch of artificial intelligence and are designed to mimic the functioning of neurons in the human brain. Neurons act together as a chain, where the output of one is the input of the other. Initially, sensors are created to solve binary classification problems. Today, neural networks are used to complete tasks where standard machine learning methods fail. Rosenblatt also pointed out some of the disadvantages of sensors such as incompatible with nonlinear functions, but it was a very innovative article for the day.

In the 1960, Bernard Widrow and Ted Hoff published their work “Adaptive Linear Element” which introduced a very important algorithm for supervised learning and signal processing called the Least Mean Squares (LMS) algorithm. The LMS algorithm updates the coefficients or weights of a linear filter and aims to minimize the average error between the filter's output and the expected output. A linear filter is a mathematical function used in signal processing that performs a linear transformation on the input and output signals. (Widrow & Hoff, Adaptive Linear Element, 1960)

Also that year, another important paper, "The Delta Rule for Learning in Neural Networks," was published by Widrow and Hoff. In this study, they also examined neural networks, but this time they developed a learning rule called the Delta Rule. It works similarly to LMS as both are used to update the weights of neural networks.

(Widrow & Hoff, The Delta Rule for Learning in Neural Networks, 1960) To date, both the LMS algorithm and the Delta Rule are actively used in neural network studies and are considered as the building blocks of the field.

Over the years, the machine learning literature continues to expand. In 1995, Vladimir Vapnik and Corinna Cortes wrote the paper "Support-Vector Networks" in which they developed a new classification approach. Vapnik and Cortes worked on binary classification problems in high-dimensional spaces using hyperplanes, a geometric object used to separate data points. The goal of the Support-Vector algorithm is to find the hyperplane that creates the largest margin between the dataset classes. They also used kernel functions to analyze complex datasets in higher dimensions where they are linearly separable. (Vapnik & Cortes, 1995) This article is important because the classification problems are studied geometrically and also some new concepts such as hyperplanes and kernel functions are proposed for classification settings. Support-Vector Networks, often called Support-Vector Machines today, are the common solution to many classification problems and are known by heart to almost every data scientist.

In 2012, Pedro Domingos published the article "A Few Useful Things You Should Know About Machine Learning". In his article, Domingos wrote a comprehensive overview of the key concepts and important definitions of machine learning. It is also possible to gain knowledge from some of the techniques applied while developing a machine learning model. (Domingos , 2012) Domingos' article stands as a must-read for anyone working on machine learning.

In recent years, the world has become more and more familiar with the creative side of artificial intelligence. When we surf the Internet, we often see some kind of media created by programs powered by artificial intelligence. The next article is about the background of productive artificial intelligence.

Goodfellow et.al. published their paper "Generative Adversarial Networks" in 2014 and introduced a system in which a pair of neural networks, called generative neural network and discriminative neural network, perform interactively to generate new data samples. (Goodfellow et al. 2014)

Another interesting subfield of artificial intelligence is Natural Language Processing (NLP), which is intended to analyze how the languages spoken by humans are interpreted by computers. NLP makes it possible to create programs that can produce human-language outputs.

An important article about NLP called “BERT: Pre-training of Deep Bi Directional Transformers for Language Teaching” was published in 2018 by Devlin et al. They suggested that adding one more step to the training phase of traditional neural network pipelines improves overall efficiency when working with large amounts of data. (Devlin, Chang, Lee, & Toutanova, 2019)

Artificial intelligence is an emerging field and is being developed more and more every day. Today, we use artificial intelligence in almost every field. As the needs of the modern world increase and change, artificial intelligence will help us more than ever before. Some common artificial intelligence applications are facial recognition, smart recommender systems, self-driving cars and financial analysis bots.

Recently, the world has been introduced to the latest application of artificial intelligence, large language models (LLM). LLMs are the most advanced delivery of the field, they are trained with an extremely large amount of data and can learn billions of different parameters simultaneously. LLMs have a large number of use cases and can be applied to almost any business area. The most popular use cases of LLMs are real-time language translations, chatbots and automated content creation. The following article is taken from the early stages of the history of the LLM. Latent Dirichlet Allocation is an article published by Blei, Ng and Jordan in 2003. The following article is taken from the early stages of the history of the LLM. “Latent Dirichlet Allocation” is a paper published by Blei, Ng and Jordan in 2003. The article describes the probabilistic model Latent Dirichlet Allocation (LDA) developed to analyze the contextual structure of different document types while maintaining the basic statistical relationships between them. (Blei, Ng, & Jordan, 2003) It provides solutions for text mining, document classification and information retrieval tasks. This article is one of the main researches in the field of LLM and is a guide for many related studies.

The articles mentioned up to this point roughly describe the historical development of artificial intelligence. The rest of the literature review is focused on machine learning approaches to classification problems with imbalanced data. The literature on this subject is quite extensive and includes many important studies. Machine Learning for the Detection of Oil Spills in Satellite Radar Images by Kubat, Holte and Matwin is an early example on this matter. The motivation of their work was to develop automated systems that can detect oil spills in radar images and reduce the harmful impact of such events on the environment and the economy. (Kubat, C. Holte, & Matwin, 1998) Their article also highlights how machine learning solutions can be applied flexibly to many different fields. The research is focused on feature extraction from radar images and classification algorithms. In this context, techniques originated from Natural Language Processing (NLP) is applied, such as feature vectorization. Also, the classic machine learning topics such as decision trees and support vector machines (SVM) are used and examined in detail, too. The dataset generated from radar images is an imbalanced dataset containing many more negative examples than positive examples, positive examples indicating the oil spills. The ratio of majority classes to the whole set was given as 96%, which sets an example of a severely imbalanced dataset. k-NN with $k = 1$, the nearest neighbour, is one of the learning algorithms used in this article. The imbalance of the dataset was mentioned as one of the main problems encountered in the study. The need for a large-scale comparative study of methods of dealing with imbalanced classes is one of the conclusions of the article.

Another important paper on imbalanced classes was published in 2002 by Chawla, Bowyer, Hall and Kegelmeyer and was titled “SMOTE: The Synthetic Minority Oversampling Technique”. SMOTE is one of the selected techniques studied in this thesis. SMOTE increases the size of the minority class by generating new samples around existing ones synthetically, preserving the characteristics of the samples. It uses one of the basic classification algorithms, the k-Nearest Neighbor Algorithm (k-NN), and a distance metric to place newly created samples. Here k is a hyperparameter, that is, it is defined by the user and is usually given as 5. After the nearest neighbors of the minority class are identified, new samples are placed between the nearest neighbors

and the existing samples. This process is an example of interpolation around minority class examples. The paper shows that the combination of SMOTE and under-sampling the minority class can achieve better results in modelling performance, comparing the results obtained when applied under-sampling only. The experiments in the study is performed using C4.5, Ripper and Naive Bayes algorithms. The method is evaluated using the area under the Receiver Operating Characteristic curve (AUC) and the ROC convex hull strategy. (Chawla, Bowyer, Hall, & Kegelmeyer, 2002) SMOTE helps to smooth the imbalance between classes by creating examples to add the minority class, but it should be used with the point in mind that oversampling a minority class can negatively affect the generalization power of the model.

ADASYN (Adaptive Synthetic Sampling Approach) is an over-sampling technique developed based on SMOTE. It is proposed in the article “Adasyn: Adaptive Synthetic Sampling Approach for Imbalanced Learning” by He, Wang and Li in 2008. ADASYN also uses k-NN algorithm to place generated samples in the dataset. It performs weighted resampling, that is, ADASYN weighs samples from the minority class according to how heavily they are represented. (He, Bai, Garcia , & Li, 2008) Then, it places the generated samples around the least represented samples within the minority class. The differences between SMOTE and ADAYSN are the weighing strategy of the samples and the use of the k-NN algorithm. The ADASYN approach has been shown to improve the learning capacity of the model by reducing the bias coming from class imbalance and shifting the classification decision boundary to examples that are harder to learn. The method was tested on different datasets using machine learning algorithms and evaluated with five different metrics.

Batista, Prati and Monard give a comprehensive summary of the techniques developed to deal with the imbalanced class problem in their 2004 study "A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data". (Batista, Prati, & Monard, 2004) Weighted sampling, under-sampling and synthetic sampling techniques have been studied in detail and their advantages and disadvantages have been stated. It is concluded that the weighted sampling strategy increases the effect of minority samples on the model, but does not solve the problem of poor representation, since their numbers remain the same. Under-sampling is found to be effective in

reducing class imbalance, but since sampling is performed randomly, it can lead to a loss of critical information. Synthetic sampling brings class sizes closer by increasing the size of the minority class, and it is concluded that it is a good approach overall. The study of Batista et al. is a helpful reference for researches studying on the topic.

The articles that have been summarized so far are related to data-level solutions against the imbalance class problem, which is the focus of this, but it is useful to have information about the ongoing studies on algorithm level and hybrid solutions. "A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches" is an article written by Galar et al. in 2018. The research focuses on ensemble machine learning methods developed to overcome imbalanced data problem. The methods includes bagging (bootstrap aggregating), boosting and hybrid methods. Bagging is an ensemble learner and which uses decision trees as base learners. (Galar, Fernandez, Barrenechea, Bustince, & Herrera, 2011) These base learners, also called weak learners, are trained on the samples of the dataset. The predictions are made by the averaging or the voting of the base learner's output, depending on the problem type. This way, the different outputs are gathered and a more balanced model is obtained. Boosting is another ensemble model and aims to increase the performance of weak learners. In boosting, weak learners are trained consequently to correct the mistakes made by previous ones. One of the motivation for this study is the lack of taxonomy of the ensemble methods developed for the issue. Once the taxonomy is defined, the methods has been tested on different real-world examples with imbalanced data and their performances are compared to ensemble methods and to non-ensemble methods. Ensemble methods has been seen to perform better than the other methods. Also, it has been emphasized the good collaboration between re-sampling techniques and bagging type learning algorithms.

In this literature review, it is aimed to make a compact review of outstanding articles written mostly on important aspects of artificial intelligence and machine learning and the solutions developed so far for the imbalanced data problem on classification problems.

2. PURPOSE OF THE STUDY

In this thesis, it is aimed to study and compare 5 different types of data-level imbalanced data handling techniques on a binary classification problem with a highly imbalanced dataset. The techniques applied have different level of complexities, with their own advantages and disadvantages. Two of them are the most basic re-sampling techniques and the rest is more developed than the others. The reason data-level techniques is chosen is that they offer a simpler solution than the other techniques in general and it is reasonable to analyze this type of techniques first before studying the imbalance learning problem deeper.

The classification problem is solved using the appropriate machine learning algorithm and the model performance is evaluated with metrics sensitive to class imbalance. Then the dataset is pre-processed individually with the techniques chosen and model is re-runned with the modified datasets. The model performances on the original dataset and on the modified datasets is compared with each other. Each imbalanced data handling technique is assessed taking into account their impact on the predictive power of the model.

In the dataset, only 13% of the samples belong to the class the minority class, that is more critical to be classified correctly. According to literature, the datasets that has 13% - 20% class ratio are mildly imbalanced datasets. The datasets that has lower ratio of positive classes (the class of interest) may require more complex solutions than data level imbalanced data solutions but for mildly imbalanced datasets the techniques studied in this thesis is a good starting point.

3. MACHINE LEARNING

3.1. Types of Machine Learning Algorithms

“Machine learning is a subfield of artificial intelligence that deals with the development of algorithms and models that enable a computer system to automatically learn and improve from experience, without explicitly programmed.” (Alpaydin, 2010). This definition of machine learning has been given in the book “Introduction to Machine Learning” written by Ethem Alpaydm in 2010. In 2016, Goodfellow et al. has given another definition for machine learning in their book about deep learning: “Machine learning is the process of desinging algorithms that can learn patterns and relationships from data, and use that knowledge to make accurate predicitons or decisions on new, unseen data.” (Goodfellow, 2016). One more definition for machine learning is from the book “Data Science: An Introduction” by Kelleher and Tierney which is stated as: “Machine learning is a method of data analysis that automates analytical model building. It is a branch of artifical intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.” (Kelleher & Tierney, 2018). These are some definitions of machine learning coming from the literature. They all emphasize these characteristics of machine learning, its ability to analyze data and make inferences from it automatically to be used in decision making processes. Machine learning is trying to generate approximate solutions to difficult real world problems by applying statistical reasoning. Machine learning does more than memorizing facts and rules, instead it learns the structure of the problem by understanding nature of the data behind it. That is why a good explanatory data analysis (EDA) is crucial for a succesful ML workflow. Briefly, EDA is the collection of statistical operations performed to understand and gain insight about the dataset. EDA asks questions like “What type of data does the dataset contains?”, “What is the statistical distribution of the variables?” , “Are there any missing values? If so, how to approach this issue?” etc. Conducting a comprehensive EDA session enables the researcher to select and implement the most suitable ML approach to the problem at hand.

Most of the machine learning studies follows the same steps: problem definition and data collection, data preprocessing, algorithm selection and training, model evaluation, model optimization, model distribution and production. Like any other practical project, ML projects start with the problem definition. Once the problem is identified, data is collected from relevant sources. The data to be used may come from different sources and therefore may be in different formats or noisy, cleaning and formatting the data is a step in the data preprocessing stage. Data may contain variables that are widely distributed across different ranges, making it difficult for the algorithm to generalize. Scaling and standardization operations can be performed in data preprocessing to project the desired portion of the data into a specific range without changing any of the statistical properties. Also, different types of data may appear in the dataset, as numerical and categorical data. Most of the machine learning algorithms assume that the data they are working with is numerical. If there are any categorical variables in the dataset they must be transformed to proper format, one-hot encoding is a good practice at this point. One-hot encoding creates additional binary variables for each category a categorical variable contains. Another point to consider about categorical variables is that sometimes they may be mistaken as ordinal instead of nominal by the algorithm. Nominal categorical data shows no hierarchical dynamic but ordinal categorical data, as the name suggests, indicate an order. In general, the data preprocessing stage prepares the data for the model to work on through operations similar to those mentioned above.

Once the data is ready, the model can start training on the training set. To effectively evaluate machine learning models, the dataset is split into two, or three if validation is applied, training and test sets. The training set is where the model completes its training phase. Once the training phase is complete and predictions are made, the model's performance is evaluated on a dataset that has never been seen by the model before. The logic of training and test sets can be likened to giving the model an exam with questions that have never been studied before. The sizes of the training and test sets are typically 75%-25% or 80%-20% of the original dataset, respectively. There are additional steps to be taken to improve the performance of the model, hyper-parameter optimization is one of them. Each algorithm has its own set of hyper-parameters

depending on the type of model. In hyper-parameter optimization, the aim is to find the best combination of this set of parameters in which the model performs best. Here, the goal of the optimization process can be changed according to the task.

Once the model's performance reaches a satisfactory level, it can be integrated into production environments. Depending on the area of use, the requirements of the model may change over time. For example, if the model provides solutions in the financial sector, current economic conditions may require some changes to the model's settings. In addition, the model's dataset should be kept up-to-date to capture the latest relationships from the real world.

Mainly, there are three types of machine learning algorithms: supervised learning, unsupervised learning and reinforcement learning. Some sources divide the machine learning algorithms into four by adding the category semi-supervised learning. Each type of machine learning algorithm has different use cases according to their properties. For example, supervised learning algorithms are used for prediction of future cases while unsupervised learning algorithms are suitable for clustering purposes, like customer segmentation and image compression. Reinforced learners are widely used in recommender systems and in game playing.

In detail, supervised learning happens when the model is trained with labeled data. This means that the dataset contains information about input variables, as the name suggests. This information can be qualitative or quantitative, depending on the type of dataset. The learner is expected to learn a rule that maps the input data to the correct output data, based on the given information. Performance evaluation of supervised learning models is done on the test set. This way, generalization power of the algorithm to predict labels correctly is tested on a dataset that is never seen before by the model. The two main types of supervised learning are regression and classification. In regression problems a continuous output is generated by the model and in classification problems the model generates a discrete output. Linear regression, logistic regression, decision trees, random forests, support vector machines and neural networks are some well-known and widely used machine learning algorithms from the supervised learning family.

Unsupervised learning algorithms work with unlabeled datasets. Unlike the datasets used in supervised learning, the training dataset has no target or output variables representing the desired output, hence the name “unsupervised”. So, they can not produce outputs like supervised learners do. Instead, they discover the underlying structure of the data and understand the relationships within. Unsupervised learning algorithms are suitable for problems where there is need to gain deeper insight about the nature of the data. Also it is possible to make use of unsupervised learning algorithms in different stages of machine learning pipelines, such as in missing value treatment and dimensionality reduction in data preprocessing step.

As a combination of supervised learning and unsupervised learning, there are semi-supervised learning algorithms where both labeled and unlabeled data are present in the training dataset. This type of algorithms try to improve the model performance on labeled data with the help of unlabeled data. Semi-supervised learning algorithms are useful when there are sufficient amount of unlabeled data but the access to labeled data is limited. In an semi-supervised setting, the unsupervised learner with unlabeled data is used initially to provide understanding about the data and then labeled data is used to fine-tune the model. An alternative approach is to make predictions on the labeled data with an supervised learning algorithm first and then predict the labels of unlabeled data, using the outputs from first step. The ratio of labeled and unlabeled data present in the training set is important and should be determined carefully since it affects the model performance directly. Image classification, natural language processing (NLP) and speech recognition are among the use cases of semi-supervised learning.

Reinforcement learning algorithms work with reward system. In this type of learning, the decisions are made in an interactive enviroment. The agent, learner in reinforcement learning, receives positive or negative feedbacks in form of rewards or penalties. This way, reinforcement learning agents come up with a policy as the output, a set of actions, to have the maximum reward over time. It would not be wrong to see reinforced learning processes as trial and error methods.

3.2. Mathematical Problem Definitions of ML Algorithms

Machine learning and mathematics can not be thought of separately. Machine learning developers use both computer science and mathematics heavily. For the mathematics part, a few different branches of mathematics are used such as linear algebra, calculus and probability & statistics. Data subject to machine learning studies are generally represented using vectors and matrices. Other techniques applied in later stages of machine learning pipelines, like feature extraction and dimensionality reduction, are based on linear algebra, too. Optimization is an essential concept for machine learning studies and is performed by the methods like gradient descent which is heavily reliant on calculus. In addition to this, probability and statistics are both essential components of machine learning in general, since there are uncertainty and randomness involved.

Considering a set of two-tuples:

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \quad (3.1)$$

where x_i 's are the input features and y_i 's are the output features. The terms “input features”, “feature” and “predictor variable” are interchangeable, same holds for “output feature”, “response variable” and “target variable”. It is important to add that, the terminology for input and output data may vary depending on the type of machine learning task. For example, in the image recognition tasks it is more suitable to refer input data as “pixels” since it is a visual format.

Machine learning algorithms aim to find a function that maps the feature variables to target variables. This function, together with its set of parameters, is referred to as model. In general, the goal is to find the model parameters that minimize the loss function, which looks at the difference between the model's output and the actual value. From this perspective, machine learning problems are also optimization problems. General form of a supervised machine learning problem is:

$$\hat{\theta} = \arg \max_{f \in F} \sum_{i=1}^n \text{goodness}(y_i - f(x_i)) \quad (3.2)$$

Here F is hypothesis/program space, n is the number of samples in the input set, *goodness* is a performance measure of choice. Alternatively, the problem can be stated with cost function C instead of *goodness* criteria, and if so the goal would be to find the value of parameters that minimizes the $C(y_i - f(x_i))$. Hence the resulting problem would be as given below:

$$\hat{\theta} = \arg \min_{f \in F} \sum_{i=1}^n C(y_i - f(x_i)) \quad (3.3)$$

In both representations, y_i is the observed outcome, f is the selected hypothesis and $f(x_i)$ is the observed outcome. Selection of function f together with corresponding cost function C determines the type of supervised machine problem. For instance, if f is chosen as linear function with C to be the sum of squares then the resulting problem would be linear regression. f and C are chosen according to the task at hand.

Consider the sets of training samples and their labels: $\{X, Y\}$. Let θ be the set of model parameters. Decision function $f(X; \theta)$ together with cost function

$$C(f(.; \theta), X, Y) \quad (3.4)$$

represents a classification algorithm. The hypothesis selected in classification problems is sometimes referred as “decision function” because they act as rule or set of rules that decides the labels (or classes) of input instances. Specifying forms of classifier f and cost function C specifies the classification algorithm.

4. IMBALANCED CLASSIFICATION

Classification problems are one of the main problems studied in supervised machine learning. In classification problems, the goal is to correctly classify a group of data points into two or more classes. In order to model to perform well, every class in the dataset must be almost equally represented. That way, there are enough examples for the algorithm to learn from, but this is almost never the case with real-world data. Datasets created from data collected from patients with rare diseases are an example of such datasets. Detection of fraudul bank account movements is an important problem of the banking sector, machine learning models developed to solve this problem are trained with such data. Therefore, the quality of datasets is crucial for machine learning models designed to address imbalanced classification problems.

A lot of work has been done to overcome the imbalanced classification problem. It is possible to find approaches that modify the dataset prior to model training or update the existing model to make it more robust to class imbalance. In this study some imbalanced data handling techniques that operates on the dataset by resampling the instances has been applied. In order to be able to compare these techniques' overall performance, first the model is trained and prediction has been made by using logistic regression. The reason to choose logistic regression as a base model is that it is a simple model suitable for binary classification tasks. Also logistic regression is not robust to class imbalance originally so it is easy to see the effect of imbalanced data handling techniques directly.

Logistic regression is a type of regression model that returns the probability value of a binary dependent variable based on one or more independent variables. It is a highly used statistical technique especially on binary classification tasks. To model the relationship between dependent and independent variables logistic regression uses the logistic function (sigmoid function). Logistic function maps any real number to a probability value. Logistic regression equation is defined as:

$$P(Y = 1 | X) = \frac{1}{1 + e^{-z}} \quad (4.1)$$

where $P(Y = 1|X)$ (4.2) is the probability of event Y happening given the values of independent variable. Here Y is the dependent variable and X 's are the independent variables. z is the linear combination of independent variables is also written as:

$$z = \beta_0 + \beta_1x_1 + \beta_2x_2 + \cdots + \beta_nx_n \quad (4.2)$$

The coefficients in the equation (4.3) are determined using the process maximum likelihood estimation. Logistic regression model aims to find the coefficients that maximizes the likelihood of the observed data. Predictions with logistic regression model is made by setting a threshold value on calculated probability values. Using that threshold value samples can be assigned to classes.

Logistic regression has some assumptions that should be checked before building the model. Violation of the assumptions does not necessarily mean the results of the model will be invalid but the satisfaction the assumptions ensures the accuracy and reliability of the results. There are mainly 6 assumptions of logistic regression. One of them is obviously met by this study which is the outcome being binary. Logistic regression assumes the existence of a binary dependent variable which has only two categories or levels. Second is linearity of predictors, logistic regression assumes the linearity in the relationship between independent variables and the log-odds of the dependent variable. Also, observations in the dataset subject to a logistic regression study must be independent of each other. Dependent observations may yield to biased or inefficient parameter estimations. In addition to these, independent variables should not be correlated with each other, i.e there should not be any multicollinearity among independent variables since it may cause unstable parameter estimations. Logistic regression models works best with large number of data samples. There is restriction on the size of training set but it should be proportionly big to set of independent variables. Lastly, it should be ensured that the dataset is free of influential outliers since they might in affect the parameter estimation process in a negative way. The

following 5 sub-sections explain imbalanced data handling techniques chosen in detail.

4.1. Random Over-Sampling

Random over-sampling is one of the most common data-level techniques to address imbalanced data problem in machine learning. In imbalanced datasets the class with the least number of samples is referred to as minor (or minority) class, similarly the class with the most number of samples is referred to as major (or majority) class. The aim of random over-sampling technique is to increase the size of minority class by duplicating samples randomly chosen from the minority class. This process can be continued until the desired balance is achieved between the classes.

In some cases random over-sampling may lead models to overfit since it duplicates the existing samples, not adding new information. Random over-sampling has potential risk of overfitting due to duplicating minority samples. As an advantage, it results in robust accuracy and computational efficiency. (Kamalov, Leung, & Cherukuri) Also, duplication of randomly chosen examples may increase the number of outliers and noise. Random over-sampled data may deviate from original distribution and it potentially produce noisy data. (Han, Qiao, & Zhan) Even though the random over-sampling process is simple and effective on neat datasets it should be applied with care to diversified datasets. Combination of random over-sampling with other similar techniques may perform well in more complex and large datasets.

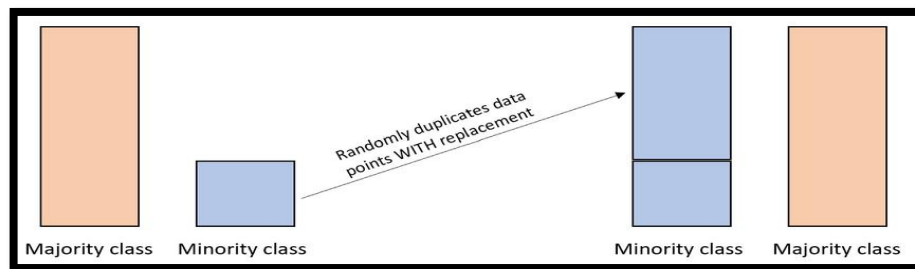


Figure 4.1 : Random over-sampling process. (Wongvorachan, He, & Bulut, A Comparison of Undersampling, Oversampling and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining, 2023)

4.2. Random Under-Sampling

Random under-sampling is another commonly used imbalanced data-handling technique that works similar to random over-sampling. The only difference between random under-sampling and random over-sampling is the resampling type, as the names suggest. Random under-sampling aims to decrease the representation of majority class in the dataset and create a more balanced setting. It chooses random samples from the majority class and removes them from the dataset.

One thing to note when working with random under-sampling is that it may cause underfitting by removing samples from the dataset that holds critical information to the model. Random under-sampling may discard important information and has risk of potential for sampling bias. (Okanovic et al, 2023) To address this potential drawback, just like with random over-sampling, it should be applied with care to the large datasets and evaluate the performance of the model with suitable metrics.

On the positive aspects, it helps balance imbalanced datasets by removing majority class samples randomly, reduces training time and achieves competitive accuracy levels. (Okanovic et al, 2023) (Hengyu, 2021) Again, combining random-under sampling with another technique may increase the overall performance in such cases

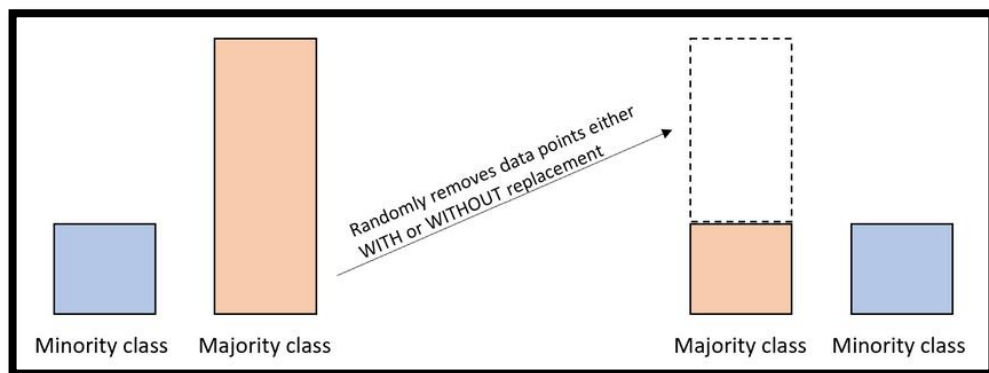


Figure 4.2: Random under-sampling process. (Wongvorachan, He, & Bulut, A Comparison of Undersampling, Oversampling and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining, 2023)

4.3. SMOTE (Synthetic Minority Over-Sampling Technique)

SMOTE is one of the most widely used and effective techniques developed to overcome the imbalanced data issue. It was first proposed in the paper "SMOTE: Synthetic Minority Oversampling Technique" by Chawla, Bowyer, Hall, and Kegelmeyer, published in the Journal of Artificial Intelligence in 2002. SMOTE can be seen as an improved version of oversampling and is described by the creators of SMOTE as "an oversampling approach in which the minority class is oversampled by creating 'synthetic' samples rather than oversampling with replacement." (Chawla, 2002) Their work is inspired by the study of Ha et al. made with handwritten character recognition data in 1997. SMOTE synthesies new minority samples between existing ones by using linear interpolation technique. It also adds some noise to the generated data samples, ensuring that they are still from the same distribution of the minority class. This technique is called "data augmentation" and helps to prevent over-fitting which is one of the main issues encountered when resampling data.

For every example in the minority class, one or more of the k -nearest neighbors are arbitrarily chosen to create these synthetic samples. The dataset is reconstructed following the oversampling procedure, and is ready for the classification algorithms to work on. Let A represent the minority class, for all $x \in A$, the k -nearest neighbors of x is identified using Euclidean distance. "Depending upon the amount of over-sampling required, neighbors from the k -nearest neighbors are randomly chosen." (Chawla, 2002) . The value of k is taken as five in the original work. N , the sampling rate, is also set proportional to the imbalance level of the dataset. Then the set A_1 is constructed by randomly selecting $N - many$ elements $(x_1, x_2, \dots, x_n)$ for each $x \in A$. Lastly, the given formula is used to generate a new example for each $x_k \in A_1$.

$$x' = x + rand(0,1) * |x - x_k| \quad (4.3)$$

$rand(0,1)$ represents the set of randomly chosen numbers between 0 and 1 and adds noise to the dataset when multiplied by the distance between x and x_k . The pseduo-code of the algorithm (Chawla, 2002) is given as follows:

Algorithm: $SMOTE(T, N, k)$ (4.4)

Input: Number of minority class samples T ; Amount of SMOTE $N\%$; Number of nearest neighbors k

Output: $\left(\frac{N}{100}\right) * T$ synthetic minority class samples

1. (* If N is less than 100%, randomize the minority class samples as only a random percent of them will be SMOTEd *)
2. If $N < 100$
3. then randomize the T minority class samples
4. $T = \left(\frac{N}{100}\right) * T$
5. $N = 100$
6. Endif
7. $N = (int)\left(\frac{N}{100}\right)$ The amount of SMOTE is assumed to be in integral multiples of 100. *)
8. k = Number of nearest neighbors
9. $numattrs$ = Number of attributes
10. $Sample[][]$: array for original minority class samples
11. $newindex$: keeps a count of number of synthetic samples generated, initialized to 0.
12. $Synthetic[][]$: array for synthetic samples
(* Compute k nearest neighbors for each minority class samples only. *)
13. for $i \leftarrow 1$ to T
14. Compute k nearest neighbors for i , and save the indices in the $nnarray$
15. Populate $(N, i, nnarray)$
16. endfor
- Populate** $(N, i, nnarray)$ (* Function to generate the synthetic samples. *)
17. while $N \neq 0$

18. Choose a random number between 1 and k , call it nn . This step chooses one of the k nearest neighbors of i .
19. for $attr \leftarrow 1$ to $numattrs$
20. Compute: $dif = Sample[nnarray][nn][attr] - Sample[i][Attribute]$
21. Compute: $gap = \text{random number between } 0 \text{ and } 1$
- $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$
23. **endfor**
24. $newindex++$
25. $N = N - 1$
26. **endwhile**
27. return (* End of Populate. *)

End of Pseudo-Code.

SMOTE is a successful technique in general, but its use has some advantages as well as disadvantages. It is a popular oversampling technique with ability to re-balance imbalanced datasets and improve overall classification performance of the models. On the other hand, SMOTE may produce instances in noisy and overlapping regions. (Grina, Elouedi, & Lefevre, 2020) Improving inference and classification quality in imbalanced data, is one benefit of applying SMOTE. Cons could include the possibility of noise introduction and higher computational effort complexity. (Pratama & Oktora, 2022)

When the literature is reviewed in general, the positive aspects of SMOTE are shown as successfully balancing the class imbalance and thus improving the model prediction performance. The following points are frequently highlighted as negative aspects: duplicating unnecessary data and hence increasing the risk of overfitting the model, oversampling data points in noisy and hard-to-classify regions of the dataset. The image below demonstrates the strategy of SMOTE.

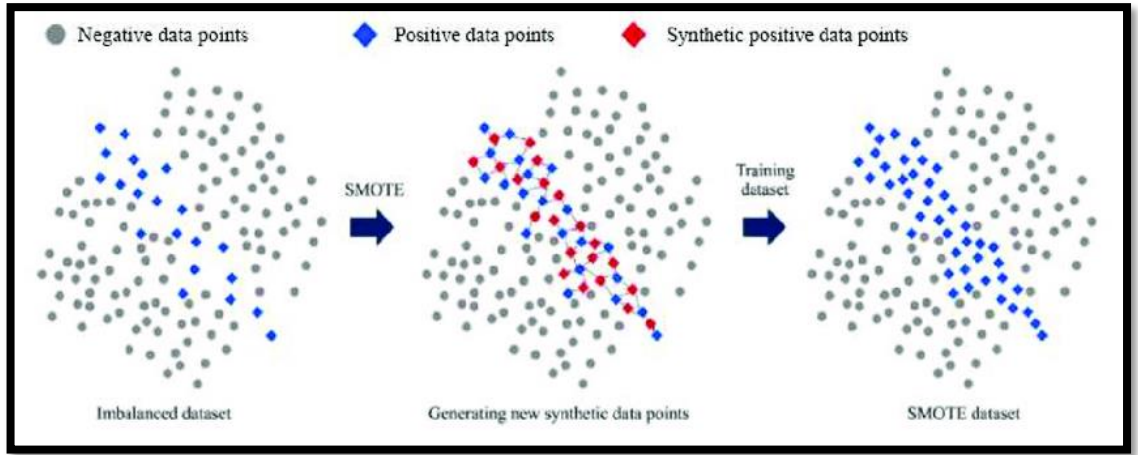


Figure 4.3: Process of Synthetic Minority Oversampling Technique Algorithm.
(Hairani, Prianto , & Anggrawan, 2023)

4.4. Tomek Links

Tomek Links is an undersampling technique from the family of Condensed Nearest Neighbours (CNN) developed by Tomek in 1976. Different from other CNN methods, Tomek Links method uses a rule to eliminate samples from the dataset. Consider that sample a and b are present in the dataset and the following conditions are satisfied:

1. The sample a 's nearest neighbour is sample b .
2. The sample b 's nearest neighbour is sample a .
3. The samples a and b belong to the different classes. In other words, they belong to minority and majority class. (or vice versa)

The aim of Tomek Link method is to identify samples from majority class that are closest to the minority class and to remove them since these kind of samples are hard to identify by the model.

To represent it mathematically, let x_i denote a sample from minority class and let x_j denote a sample from majority class. Let $d(x_i, x_j)$ denote the Euclidean distance between x_i and x_j . If there is no such sample x_k satisfying the below conditions.

$$1. \quad d(x_i, x_k) < d(x_i, x_j) \quad (4.4)$$

$$2. \quad d(x_j, x_k) < d(x_i, x_j) \quad (4.5)$$

then the pair (x_i, x_j) is said to be Tomek Link.

Like other resampling methods Tomek Link method has its own pros and cons. Considering that Tomek Link method is an under-sampling method at its core, its drawbacks are similar to other under-sampling methods. Its additional features are developed based on the weaknesses of other under-sampling methods like its fast approach based on similarity when re-sampling. (Qinghe & Ling, 2017).

It also enhances performance by removing noise from majority class since it applies under-sampling to the points that are harder to classify. (At, M., F., & M, 2016) It has been seen that Tomek Link works well with other resampling methods and classifiers, like SMOTE and Support Vector Machines (SVM). (Sun et.al 2006)

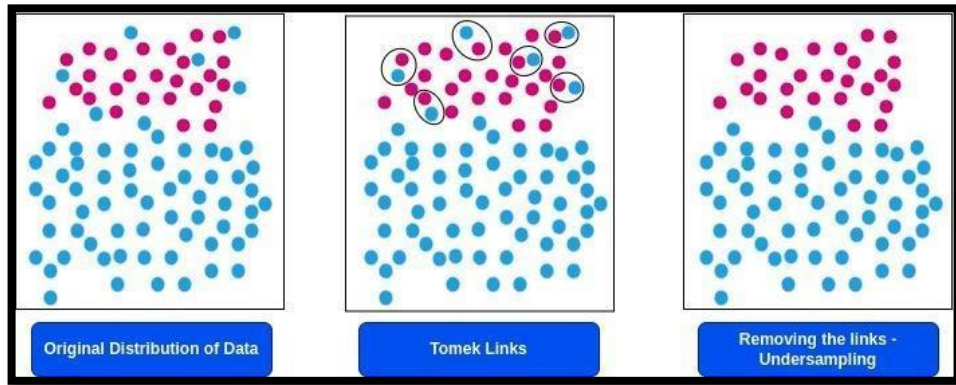


Figure 4.4: Illustration of Tomek Links. (Hussain, Oyelere, Joloudari, & Marefat, 2022)

4.5. NearMiss Algorithm

NearMiss algorithm is another under-sampling technique that aims to balance class distribution by eliminating majority class examples randomly. An example from majority class is removed from dataset when it is too close with a sample from minority class. This way, the space between classes are increased and the classification process becomes easier. Information loss is a common problem encountered in under-sampling

based techniques and nearest neighbor approaches like NearMiss algorithm are developed to overcome this issue.

In the NearMiss algorithm first the distances between all samples are calculated for both majority and minority class. Then n samples in majority class closest to minority class are selected to be under-sampled. If there are k samples in the minority class, then the algorithm will return $(k \times n)$ samples.

NearMiss algorithm offers a few different ways to find n -closest instances to the majority class. It is possible to choose majority class samples whose average distances to the k -nearest minority class cases are the least or to choose majority class samples have the least average distances to the k -farthest examples of the minority class. Last method to select majority class samples works in two steps, which is first, the M -nearest neighbors of each minority class instance will be stored. Finally, examples of the majority class with the highest average distance to the N -nearest neighbors are chosen.

NearMiss algorithm offers several advantages and disadvantages. Advantages includes improving recall and f-measure, which are metrics used to evaluate classification models. (Fahrudin, 2019) In disadvantages, similar to other under-sampling methods, there is potential information loss due to reducing dataset size. (Pias, Su, Tang, Wang, & Yao, 2023) It has also been found effective in handling class imbalance by selecting sensitive examples. (Zhang, Wang, W., Zhang, & Nugent, 2019)

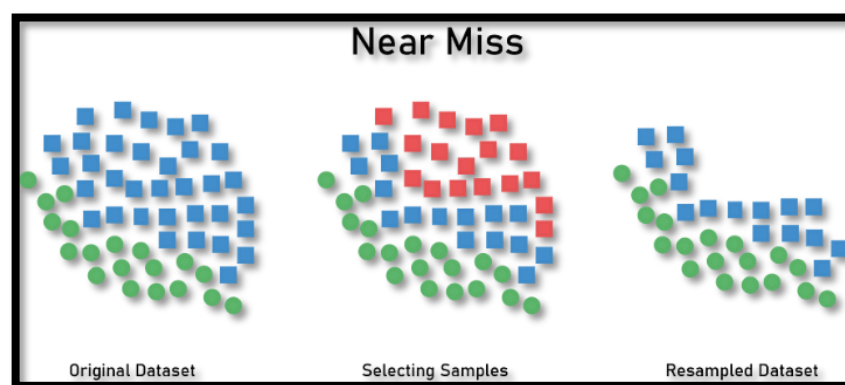


Figure 4.5: Illustration of NearMiss Algorithm. (Charfaoui, 2019)

5. APPLICATION

5.1. Explanatory Data Analysis (EDA) and Data Preprocessing

Dataset subject to this study is taken from the [UCI Machine Learning Repository](#) where it is possible to find variety of publicly available datasets. The dataset is called [“Bank Marketing Dataset”](#) and is generated by a Portuguese banking institution. Data is related with direct marketing campaigns (phone calls). Target variable y is binary and holds the information that if the product (term deposit) is subscribed by the customer or not. It is mapped to 1 if the customer accepts the campaign and mapped to 0 if doesn't.

There are 4 datasets available in the link given above. The main dataset has 41188 instances and 20 features. Other datasets are differ from each other by the number of instances and features. The dataset of choice has 10% of the instances (4119) randomly selected from the main dataset. Decreasing the number of instances makes model training faster and easier. The list of remaining features is given in Appendix A.

Distribution of variables are examined in the explanatory data analysis stage. The dataset contains no missing values.

The histogram below shows the age distribution of customers reached through this campaign. The majority of customers participating in the campaign were in their late 20s to mid-30s.

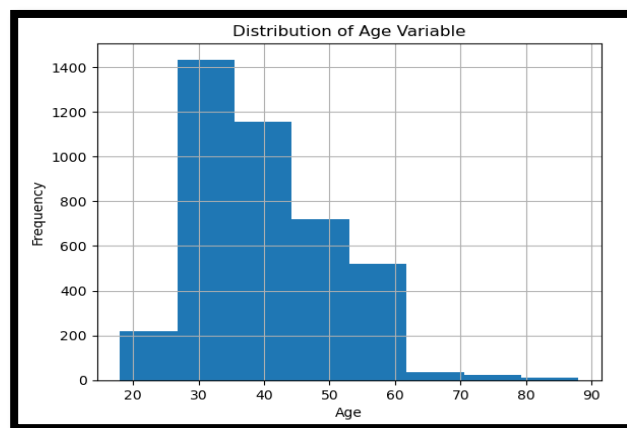


Figure 5.1: Histogram of variable “age”.

The dataset contains information of customers from different 11 (plus the “unknown” category”) type of jobs. Most of the customers have administrator, blue-collar or technician type of jobs.

Table 5.1: Distribution of variable “job”.

	count	percentage
job		
admin.	1012	0.25
blue-collar	884	0.21
technician	691	0.17
services	393	0.10
management	324	0.08
retired	166	0.04
self-employed	159	0.04
entrepreneur	148	0.04
unemployed	111	0.03
housemaid	110	0.03
student	82	0.02
unknown	39	0.01
total	4119	1.0

Customers also differ by their marital status and their education level. Distribution of these variables are given in the tables below.

Table 5.2: Distribution of variable “marital status”.

	count	percentage
marital status		
married	2509	0.60
single	1153	0.28
divorced	446	0.11
unknown	11	0.01
total	4119	1.0

Table 5.3: Distribution of variable “education”.

	count	percentage
education		
university.degree	1264	0.31
high.school	921	0.22
basic.9y	574	0.14
professional.course	535	0.13
basic.4y	429	0.10
basic.6y	228	0.05
unknown	167	0.04
illiterate	1	0.01
total	4119	1.0

The variables “default”, “housing”, “loan” are variables about whether customer has the related loan or credit. The following tables gives their distribution among the dataset.

Table 5.4: Distribution of variable “default”.

	count	percentage
default		
no	3315	0.81
unknown	803	0.19
yes	1	~0.00
total	4119	1.0

Table 5.5: Distribution of variable “housing”.

	count	percentage
housing		
yes	2175	0.53
no	1839	0.45
unknown	105	0.02
total	4119	1.0

Table 5.6: Distribution of variable “loan”.

	count	percentage
loan		
no	3349	0.81
yes	665	0.16
unknown	105	0.02
total	4119	1.0

According to the “contact” variable most of the customers are reached via cellular services by the bank employees. The day and month of contact with the customer is also available in the database.

Table 5.7: Distribution of variable “contact”.

	count	percentage
contact		
cellular	2652	0.64
telephone	1467	0.36
total	4119	1.0

Table 5.8: Distribution of variable “month”.

	count	percentage
month		
may	1378	0.33
july	711	0.17
august	636	0.15
june	530	0.13
november	446	0.11
april	215	0.05
october	69	0.02
september	64	0.02
march	48	0.01
december	22	0.01
total	4119	1.0

Table 5.9: Distribution of variable “day_of_week”.

	count	percentage
day_of_week		
thursday	860	0.208789
monday	855	0.207575
tuesday	841	0.204176
wednesday	795	0.193008
friday	768	0.186453
total	4119	1.0

“Campaign” ve “previous” variables have information of number of times the related customer is contacted during and previously of this campaign, respectively.

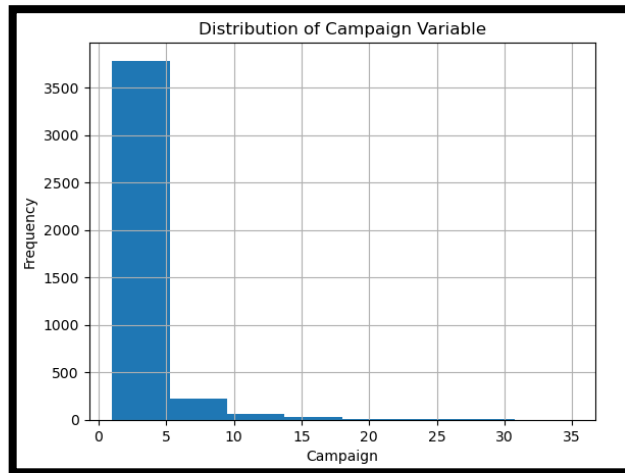


Figure 5.2: Histogram of variable “campaign”.

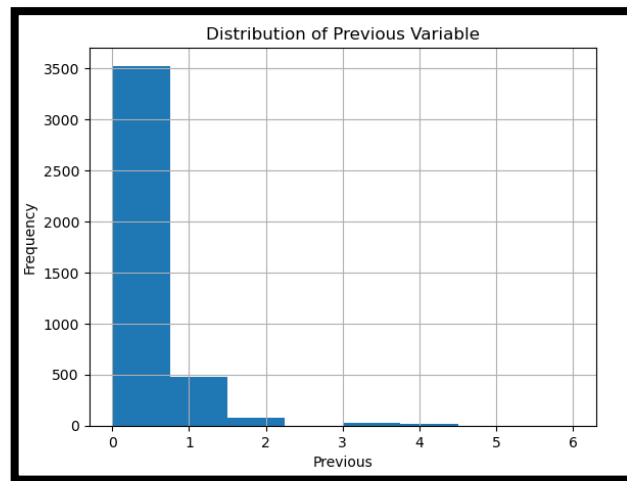


Figure 5.3: Histogram of variable “previous”.

It is also possible to find the outcome of previous campaigns in the dataset.

Table 5.10: Distribution of variable “poutcome”.

	count	percentage
poutcome		
nonexistent	3523	0.86
failure	454	0.11
success	142	0.03
total	4419	1.0

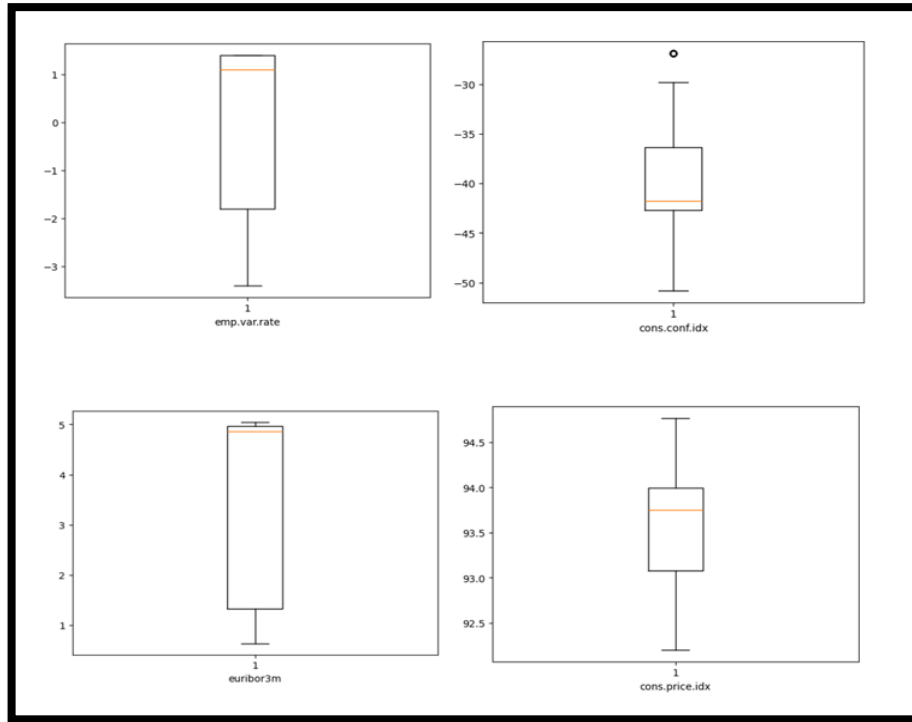


Figure 5.4: Box-plot graphs of index variables

“emp.var.rate”, *“cons.price.idx”*, *“cons.conf.idx.”*, *“euribor3m”* and *“nr.employed”* are social and economical context attributes. They are indicators of type daily, monthly and quarterly.

The target variable “y” has an imbalanced distribution among the whole dataset. Considering the ratio of y’s possible values the dataset is said to have a high-imbalance.

Table 5.11: Distribution of variable “y”.

	count	percentage
y		
no	3668	0.89
yes	451	0.11
total	4119	1.0

In machine learning workflows, the set of operations performed on the dataset before model fitting and training is called “*data preprocessing*.” Data preprocessing operations include feature engineering, outlier and missing value analysis, numerical data scaling/normalization, categorical data processing techniques, etc. These steps may vary depending on the needs of the dataset and the model.

The outlier analysis has been done for continuous numerical variables: “*campaign*”, “*previous*”, “*emp.var.rate*”, “*cons.conf.idx*”, “*euroibor3m*” and “*cons.price.idx*”. For the variable “*campaign*” the values above 5 acts as outliers. Similarly, the variable “*previous*” has some values possible note as outliers, not severe as in the latter case. “*cons.conf.idx*” has also an outlier value, shown in the box-plot in Figure 4.15. Sometimes it is a good practice to remove outliers from the dataset but since the classification algorithm of choice, the Random Forest algorithm, is robust to outliers the outlier values are kept in the dataset. (Kuswanto & Prastyo, 2023)

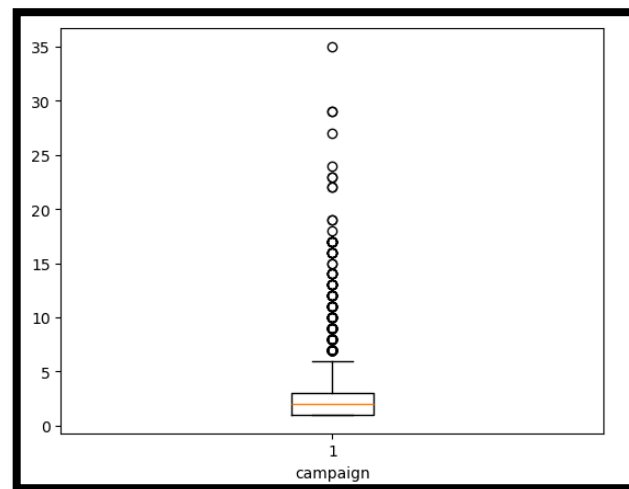


Figure 5.5: Box-plot of variable “campaign”.

It was also observed that the continuous numerical variables in the dataset have different ranges. Normally, scaling them to a fixed range improves the model performance, but since the Random Forest algorithm is robust to various ranges, no type of scaling was applied. Random Forest can work with categorical features, but they need to be encoded. Categorical variables in the data set are nominal categorical

variables, that is, their values do not contain a ranking and only express different categories. Thus, the "One-Hot Encoding" technique was applied to categorical variables and a new feature was produced for each value of a categorical feature. For example, for the variable "*contact*", which takes the possible values "*cellular*" and "*phone*", two new binary features, "*contact_1*" and "*contact_2*", were created and the original feature was removed. The target variable "*y*" take the values "*yes*" and "*no*". Its values are mapped to "*1*" and "*0*" respectively.

For the final step of data preprocessing, the dataset was split into features and target variable X and y . Separate datasets were then created for sets X and y , on which the model was trained and evaluated: X_{train} , X_{test} and y_{train} , y_{test} . This step is called "*train/test dataset splitting*".

5.2. Base Model

After the data preprocessing stage was complete the dataset is ready for the model. Before applying any of the imbalanced data techniques to the dataset a Random Forest model was trained and evaluated. Random forest is a versatile and powerful machine learning algorithm that excels in both classification and regression tasks. It is an ensemble method that combines multiple decision trees to produce more accurate and robust predictions. This algorithm is widely used in various domains, including finance, healthcare, and marketing, due to its ability to handle complex and high-dimensional data.

Random forest operates by constructing a collection of decision trees during the training phase. Each decision tree is trained on a bootstrap sample of the training data, meaning that a subset of the data is randomly selected with replacement. This process introduces randomness into the algorithm, preventing overfitting and enhancing the generalization ability of the model. Furthermore, each decision tree is constructed using a subset of features randomly selected at each split. This feature randomness further reduces the correlation between trees, making the forest less sensitive to outliers and noisy data.

Once the random forest is trained, it can be used to make predictions on new data. For classification tasks, each tree in the forest classifies the new data point, and the majority class is chosen as the final prediction. This voting mechanism ensures that the forest is not overly influenced by individual trees, leading to more reliable predictions.

While Random Forest doesn't rely on a single mathematical formula, it is built based on the of decision trees. Below there is a breakdown of a single decision tree:

- Let D be the training data with N data points and P features.
- Each data point is represented as

$$x_i = (x_{i1}, x_{i2}, \dots, x_{iP}) \quad (5.1)$$

where i denotes the data point and j denotes the feature.

- The tree recursively splits the data based on a chosen feature and a split value at each node.
- A splitting criterion, like Gini Impurity (for classification problems) or least squared error (for regression problems), is used to select the best split that maximizes the separation between classes or minimizes the error within a node.

Gini impurity is a metric used in decision trees, specifically for classification tasks, to measure the impurity or heterogeneity of a node. In simpler terms, it tells about how likely a randomly chosen data point at that node would be misclassified. Gini Impurity (G) is calculated using the probabilities p_i of each class (i) being present at a node:

$$G = 1 - \sum p_i^2 \quad (5.2)$$

This process continues until a stopping criterion (e.g., maximum depth or minimum number of data points) is met, resulting in a terminal node with a predicted class (classification) or a predicted value (regression). Lower Gini Impurity signifies a better split, as it leads to purer child nodes in the decision tree.

Application of Random Forest application was done using the open-source library of Scikit-Learn Random Forest package with Python 3. The model was fitted and trained

using test size of 0.20. Accuracy of base model's was calculated as 0.89. The formula for accuracy in classification metrics is:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (5.3) \quad \text{where}$$

- TP (True Positive): Number of correctly classified positive cases.
- TN (True Negative): Number of correctly classified negative cases.
- FP (False Positive): Number of incorrectly classified positive cases (*Type I error*).
- FN (False Negative): Number of incorrectly classified negative cases (*Type II error*).

Accuracy is a straightforward and informative metric but it can be misleading for classification problems with imbalanced data. Since it focuses on overall count, not class-specific performance: Accuracy simply calculates the total number of correct predictions divided by all predictions. In imbalanced data, the majority class often dominates, leading to high accuracy even if the model performs poorly on the minority class. Considering a model that always predicts the majority class, with imbalanced data this will lead to high accuracy, but it's not a useful model since it ignores the minority class entirely. Because of this reasons another classification metrics like *precision*, *recall* and *f1-score* were used to evaluate the model performance.

Precision, in classification tasks, tells us how many of your positive predictions were actually correct. It essentially measures the proportion of true positives among all the positive predictions made by your model. Formula of precision is stated as:

$$Precision = TP/(TP + FP) \quad (5.4)$$

where TP and FP stand for *True Positive* and *False Positive* respectively. A *precision* of 1 indicates that every single prediction your model classified as positive was actually correct.

A *precision* closer to 0 means a high number of your positive predictions were actually false positives. Precision is crucial because it is helpful to understand how reliable the

positive predictions are. In some scenarios, a high number of false positives can be very costly, such as in imbalanced data classifications.

Another important metric for imbalanced classification is “*recall*.” Recall measures how well a model can identify relevant (or positive, class of interest) data. It's also known as sensitivity or true positive rate. A high *recall* score (closer to 1) means the model is effectively capturing most of the relevant data. This is crucial when missing relevant data is costly. For example, in a medical diagnosis system, a high recall is essential to ensure you don't miss any critical cases. It is calculated as:

$$Recall = TP / (TP + FN) \quad (5.5)$$

Last metric to consider is “*F1-Score*”. The F1-score combines precision and recall into a single metric, giving a more balanced view of a model's performance in a classification task. It considers both how well the model identifies relevant data (recall) and how precise those identifications are (avoiding false positives). It is the harmonic mean of precision and recall:

$$F1 - Score = 2 \frac{Precision \cdot Recall}{Precision + Recall} \quad (5.6)$$

Models were compared by their precision, recall and F1-score values. Base model was trained on the original dataset which has 3295 rows (on training set) and 61 features. Class ratio of target variable is 89%-10%. Model has been trained with default parameters. Main parameters of Random Forest Algorithm and their default values are given below: (Pedregosa et al., 2011)

- **n_estimators:** The number of trees in the forest. *default = 100*
- **criterion:** The function to measure the quality of a split. *default = 'Gini'*
- **max_depth:** The maximum depth of the tree. *default = None*
- **min_samples_split:** The minimum number of samples required to split an internal node. *default = 2*
- **min_samples_leaf:** The minimum number of samples required to be at a leaf node. *default = 1*

Predictions were made on the test set with shape (824,61). Precision, recall and f1-score of the model's prediction for each class is given below:

Table 5.12: Evaluation metrics of the base model.

	Precision	Recall	F1-score
0	0.91	0.98	0.94
1	0.64	0.24	0.35

Below there is the confusion matrix. It plots what the model predicted against what the correct prediction was. Confusion matrix can be used to understand the tradeoff between false positives (top right) and false negatives (bottom left).

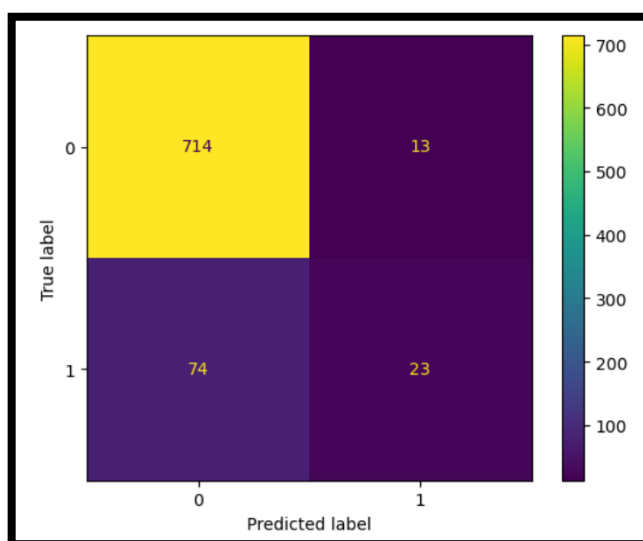


Figure 5.6: Confusion matrix of the base model.

The base model has made 13 false positive and 74 false negative prediction among 824 predictions.

In the proceeding chapter the Random Forest model trained with random over sampled data, random under sampled data, under sampled with Tomek links, over sampled with

SMOTE algorithm and under sampled with NearMiss algorithms will be referred as model 1-7, respectively.

5.3. Random Over Sampling

As the first resampling technique, Random Over Sampling technique was performed. The implementation of all of the imbalanced data techniques were implemented using Imbalanced Learning module of Scikit-Learn library. Data was resampled and then trained by the Random Forest algorithm once more, keeping the default parameters.

Random Over Sampling approach equalizes the class sizes by oversampling the minority class. The final dataset has 3668 instances of each class, making a dataset of shape (7336, 61). Model evaluation metrics and confusion matrix are given as:

Table 5.13: Evaluation metrics of the model trained on the random over sampled data.

	Precision	Recall	F1-score
0	1.00	0.94	0.97
1	0.94	1.00	0.97

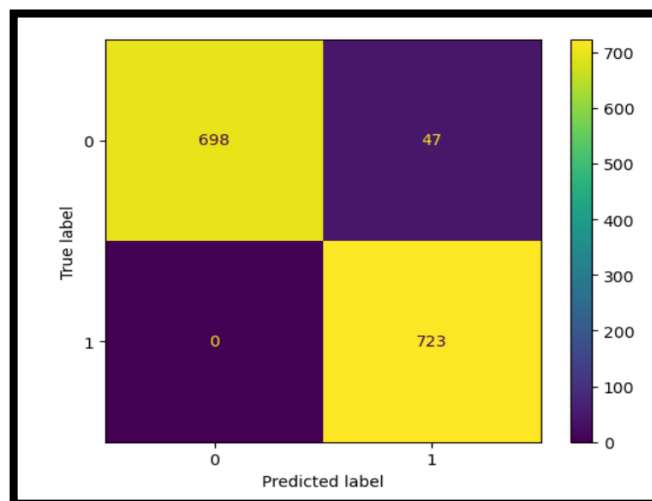


Figure 5.7: Confusion matrix of the model 1.

5.4. Random Under Sampling

A similar approach was applied to dataset with Random Under Sampling technique. In this case, class sizes were equalized by increasing the minor class by oversampling it randomly. After the resampling process was completed the dataset size became (902, 61) and the distribution of target variable became 451 samples for each class value. Below are the evaluation metrics of the model trained on resulting dataset.

Table 5.14: Evaluation metrics of the model trained on the random under sampled data.

	Precision	Recall	F1-score
0	0.72	0.76	0.74
1	0.71	0.65	0.68

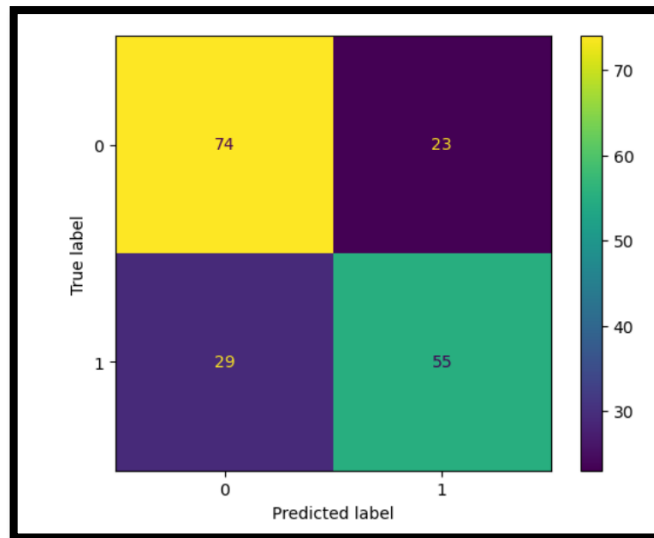


Figure 5.8: Confusion matrix of the model 3.

5.5. Under Sampling with Tomek Links

Tomek Links offers an another under sampling approach different than random sampling. This approach are based on eliminating the points satisfying the inequalities (3.4) and (3.5). Resulting dataframe has size (3987, 61) and class distribution is 3536 samples for $y = 0$ and 451 for $y = 1$. Evaluation of the proceeding model training and prediction are given as:

Table 5.15: Evaluation metrics of the model trained on the under-sampled data within Tomek Links.”

	Precision	Recall	F1-score
0	0.89	0.98	0.94
1	0.57	0.16	0.25

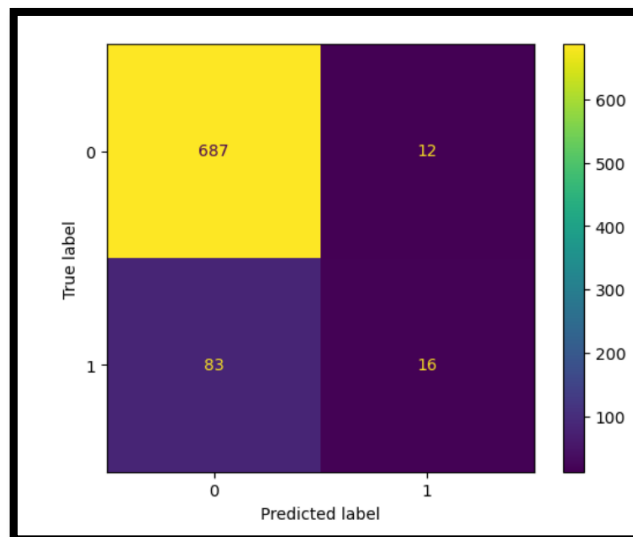


Figure 5.9: Confusion matrix of the model 3.

5.6. Over Sampling with SMOTE

Method of SMOTE synthesises samples around samples that is harder to classify, aiming improve model's generalization power. After applying SMOTE the dataset model was trained on has size (7336, 61) with equal class distribution, that is 3668 samples for each classes. Resulting model evaluation is given below.

Table 5.16: Evaluation metrics of the model trained on the over-sampled data with SMOTE.

	Precision	Recall	F1-score
0	0.93	0.95	0.94
1	0.95	0.92	0.94

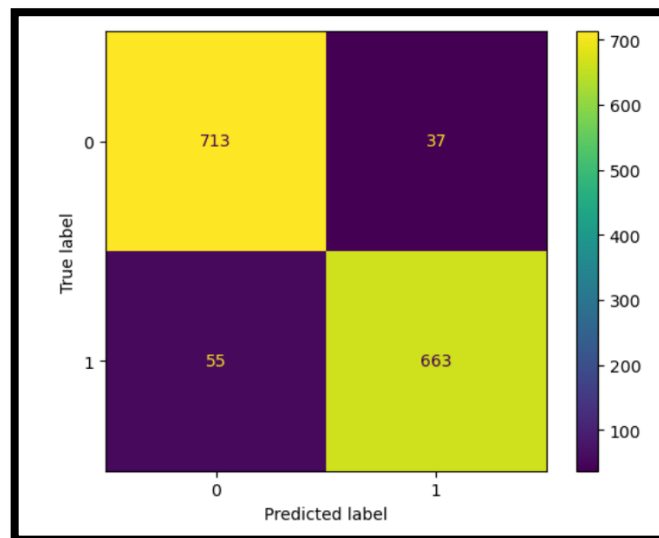


Figure 5.10: Confusion matrix of the prediction of the model trained on over sampled data with SMOTE.

5.7. Under Sampling with NearMiss Technique

NearMiss technique selects majority class examples to remove by considering their distances to minority class examples. Depending on the usage of distances between samples, NearMiss comes in three different version, as explained in detailed in Section 3. Dataset was separately resampled for all three versions of NearMiss technique. Model was trained and predictions were made for all versions, evaluation results are listed below:

For version 1:

Table 5.17: Evaluation metrics of the model trained on data under sampled by NearMiss technique version 1.

	Precision	Recall	F1-score
0	0.77	0.93	0.84
1	0.93	0.77	0.84

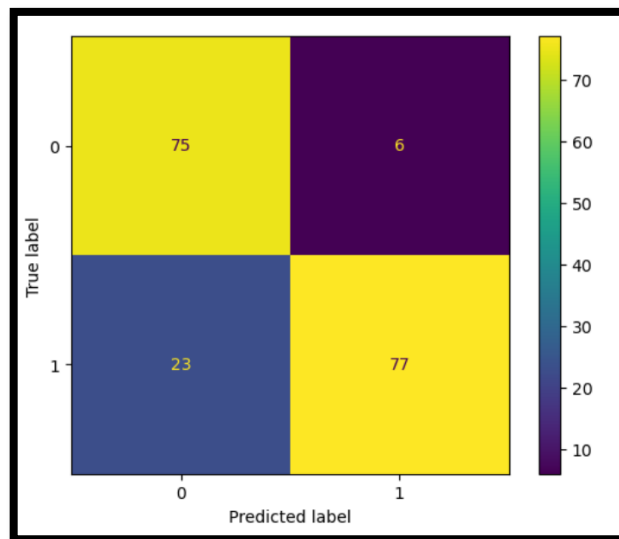


Figure 5.11: Confusion matrix of the predictions of the model trained on under sampled data with NearMiss version 1.

For version 2:

Table 5.18: Evaluation metrics of the model trained on the data resampled by NearMiss technique version 2.

	Precision	Recall	F1-score
0	0.90	0.98	0.94
1	0.97	0.88	0.93

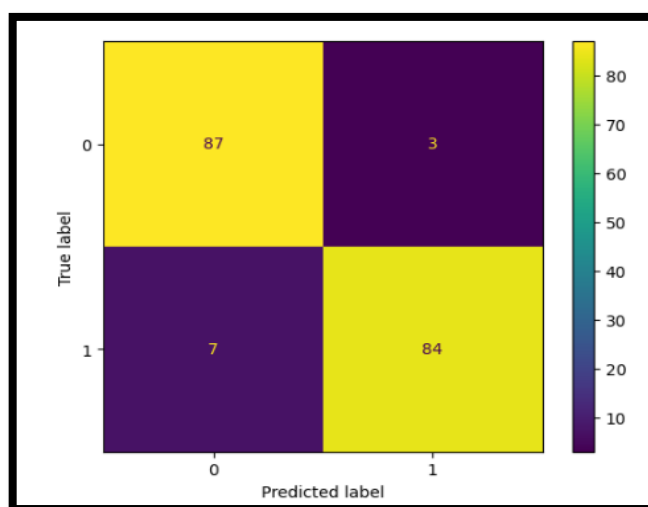


Figure 5.12: Confusion matrix of the predictions of the model trained on the data under sampled data with NearMiss version 2.

For version 3:

Table 5.19: Evaluation metrics of the model trained on the data resampled by NearMiss technique version 3.

	Precision	Recall	F1-score
0	0.56	0.62	0.59
1	0.60	0.53	0.56

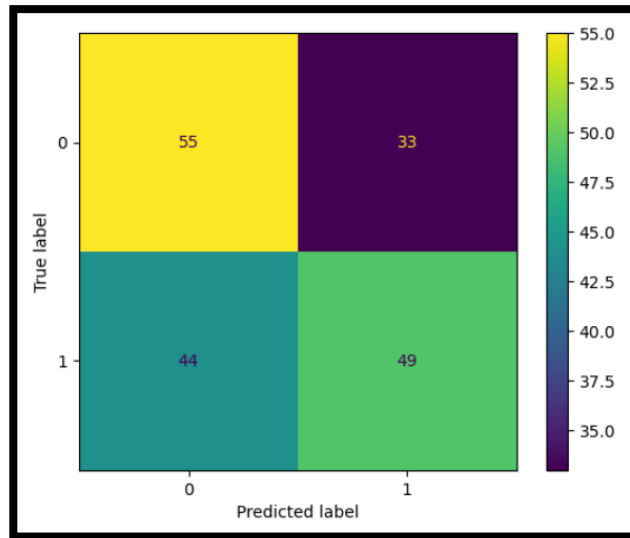


Figure 5.13: Confusion matrix of the predictions of the model trained on the data under sampled by NearMiss version 3.

6. RESULTS

Applying NearMiss Algorithm version 2 makes the most sense given the results when precision is the most crucial criterion. Similarly, if recall is the performance criterion, SMOTE is the best option out of these three unbalanced data management techniques.

The dataset were resampled 7 (4 + 3, counting the three versions of NearMiss algorithm) times separately, with 3 under sampling and with 2 over sampling methods, overall. At the end of each of resampling period, a Random Forest algorithm was trained with default parameters. Predictions were made and models were evaluated separately with precision, recall and f1-score metrics. The effects of resampling techniques will be compared over this three model evaluation metrics. Precision, recall and f1 scores of each resampling technique is given below.

Table 6.1: Precision scores for resampled datasets.

	Precision (for class of interest)
Original Data	0.64
Random Over Sampling	0.94
Random Under Sampling	0.71
SMOTE	0.95
Tomek Links	0.57
NearMiss Algorithm v1.	0.93
NearMiss Algorithm v2.	0.97
NearMiss Algorithm v3.	0.60

Table 6.2: Recall scores for resampled datasets.

	Recall score (for class of interest)
Original Data	0.24
Random Over Sampling	1
Random Under Sampling	0.65
SMOTE	0.92
Tomek Links	0.16
NearMiss Algorithm v1.	0.77
NearMiss Algorithm v2.	0.88
NearMiss Algorithm v3.	0.53

Table 6.3: F1-scores for resampled datasets.

	F1- scores (for class of interest)
Original Data	0.35
Random Over Sampling	0.97
Random Under Sampling	0.68
SMOTE	0.94
Tomek Links	0.25
NearMiss Algorithm v1.	0.84
NearMiss Algorithm v2.	0.93
NearMiss Algorithm v3.	0.56

7. CONCLUSIONS

Considering the results, it is most reasonable to apply NearMiss Algorithm version 2 when the precision is the most important metric. Similarly, SMOTE is the best choice among these three imbalanced data handling technique if recall is the performance criteria.

If both precision and recall are crucial metrics for the task at hand, then it F1-Score should be considered, which is the harmonic mean of the precision and recall. Random over sampling is the most successfull method when working with F1-Score.

REFERENCES

- At, E., M., A., F., A.-M., & M, S. (2016). Classification of Imbalance Data using Tomek Link(T-Link) Combined with Random Under-sampling (RUS) as a Data Reduction Method. *Global Journal of Technology and Optimization*.
- Batista, G., Prati, R., & Monard, M.-C. (2004). A Study of the Behavior of Several Methods for Balancing machine Learning Training Data. *ACM SIGKDD Explorations Newslette*.
- Bk, A., Djabatna , T., & Sukoco, H. (tahun yok). Streamed Sampling on Dynamic data as Support for Classification Model.
- Blei, D., Ng, Y. A., & Jordan, M. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3.
- Charfaoui, Y. (2019). *heartbeat.comet.ml*. <https://heartbeat.comet.ml/resampling-to-properly-handle-imbalanced-datasets-in-machine-learning-64d82c16ceaa> adresinden alındı
- Chawla, B. H. (2002). SMOTE (Synthetic Minority Oversampling Technique). *Journal of Artificial Intelligence*.
- Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research (JAIR)*.
- Devlin, J., Chang, M.-V., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.
- Domingos , P. (2012). A Few Useful Things to Know About Machine Learning. *Communications of the ACM*.
- Fahrudin, T. (2019). The Undersampling Effects on RANDSHUFF Oversampling Algorithms. *International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*. IEEE.
- Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). 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)*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative Adversarial Networks. *Advances in Neural Information Processing Systems*.

Grina, F., Elouedi, Z., & Lefevre, E. (2020). A Preprocessing Approach for Class-Imbalanced Data Using SMOTE and Belief Function Theory.

Hairani, H., Prianto, D., & Anggrawan, A. (2023). Improvement Performance of the Random Forest method on Unbalanced Diabetes Data Classification Using Smote-Tomek Link. *JOIV International Journal on Informatics Visualization*.

Han, Z., Qiao, X., & Zhan, S. (tarih yok). ROGA: Random Over-sampling Based on Genetic Algorithm.

He, H., Bai, Y., Garcia, E., & Li, S. (2008). ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence)*.

Hengyu, Z. (2021). An Under-Sampling Algorithm Based on SVM.

Hussain, S., Oyelere, S., Joloudari, J., & Marefat, A. (2022). Effective Class-Imbalance learning based on SMOTE and Convolutional Neural Networks.

imbalanced learn, scikit. (tarih yok). https://imbalanced-learn.org/stable/auto_examples/under-sampling/plot_illustration_tomek_links.html

imbalanced-learn.org. (tarih yok). https://imbalanced-learn.org/stable/auto_examples/under-sampling/plot_illustration_tomek_links.html

Kamalov, F., Leung, H.-H., & Cherukuri, A. (tarih yok). Keep It Simple: Random Oversampling for Imbalanced Data.

Kubat, M., C. Holte, R., & Matwin, S. (1998). Machine Learning for the Detection of Oil Spills in Satellite Radar Images. *Machine Learning*.

Kuswanto, H., & Prastyo, D. (2023). Robustness of Support Vector Regression and Random Forest Models: A Simulation Study.

Okanovic, P., Waleffe, R., Mageirakos, V., Nikolakakis, K., Karbasi, A., Guerel, N., & Rekatsinas, T. (2023). Repeated Random Sampling for Minimizing the Time-to-Accuracy of Learning.

Pedregosa et al.. (2011). Scikit-learn: Machine Learning in Python. *JMLR 12*.

Pias, T., Su, Y., Tang, X., Wang, H., & Yao, D. (2023). Undersampling for Fairness: Achieving More Equitable Predictions in Diabetes and Prediabetes.

Pratama, A. F., & Oktora, S. I. (2022). Synthetic Minority Over-sampling Technique (SMOTE) for handling imbalanced data in poverty classification. *Statistical Journal of the IAOS*.

Qinghe , L., & Ling, C. (2017). Sampling Based One-Source Link Prediction In Directed Network. *International Conference on Robotics and Automation Sciences (ICRAS)*. IEEE.

Rich Data.. https://rikunert.com/smote_explained

Salary prediction using random forest with fundamental features. (2022). *Third International Conference on Electronics and Communication; Network and Computer Technology (ECNCT 2021)*;

Sun, Y., Robinson, M., Adams, R., Boekhorst , R., Rust, A., & Davey, N. (2006). Using sampling methods to improve binding site predictions. *ESANN 2006, 14th European Symposium on Artificial Neural Networks*.

Vapnik, V., & Cortes, C. (1995). Support-Vector Networks. *Machine Learning*.

Widrow, B., & Hoff, T. (1960). Adaptive Linear Element.

Widrow, B., & Hoff, T. (1960). The Delta Rule for Learning in Neural Networks.

Wongvorachan, T.

Wongvorachan, T., He, S., & Bulut, O. (2023). A Comparison of Undersampling, Oversampling and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining. *MDPI (Multidisciplinary Digital Publishing Institute)*.

Zhang, J., Wang, T., W., W., Zhang, S., & Nugent , C. (2019). Undersampling Near Decision Boundary for Imbalance Problems. *International Conference on Machine Learning and Cybernetics (ICMLC)*. IEEE.

BIOGRAPHY

The author's name is Derya Nur COŞAR. She was graduated from Gebze Technical University, Department of Mathematics in 2021 and from Gebze Technical University Graduate School, Department of Mathematics in 2024. Her fields of study are statistical data analyse and machine learning. She has started her professional life in a financial software company in 2021 as a business analyst/software test expert. She has been working as data scientist in artificial intelligence department since 2022.

APPENDICES

Appendix-A: Attributes in the dataset

Bank Client Data			
Feature	Information	Values	Type
age	Demographic	-	numerical
job	Type of job	• admin. • blue-collar • entrepreneur • housemaid • management • retired • self employed • services • student • technician • unemployed • unknown	categorical
marital	marital status	• divorced • married • single • unknown	categorical
education	Education level of the corresponding client	• basic.4y • basic.6y • basic.9y • high.school • illiterate • professional.course	categorical

		• university.degree • unknown	
default	Does customer have credit in default?	• yes • no • unknown	categorical
housing	Does customer have housing loan?	• yes • no • unknown	categorical
loan	Does customer have personal loan?	• yes • no • unknown	categorical
Related with the Last Contact of the Current Campaign			
contact	contact communication type with the client	• cellular • telephone	categorical
month	last contact month of year with the client	• jan • ... • dec	categorical
day_of_week	last contact day of the week with the client	• mon • tue • wed • thu • fri	categorical
Other Features			

campaign	number of contacts performed during this campaign and for this client	-	numeric (includes last contact)
previous	number of contacts performed before this campaign and for this client	-	numeric
poutcome	outcome of the previous marketing campaign	• failure • nonexistent • success	categorical
Social and Economic Context Attributes			
emp.var.rate	employment variation rate (quarterly indicator)	-	numeric
cons.price.idx	consumer price index	-	numeric
cons.conf.idx	consumer confidence index (monthly indicator)	-	numeric

euribor3m	euribor 3 month rate (daily indicator)	-	numeric
Output Variable			
y	has the client subscribed a term deposit?	• yes • no	binary