ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis Econometrics and Management Science

Business Analytics and Quantitative Marketing

Combining Sampling Techniques Countering Extreme Absolute Imbalanced Data

---

Name student: Daniël Buijs
Student ID number: 483065

Supervisor: dr. MH Akyuz
Second assessor: dr. NW Koning

Date final version: March 15, 2022

Abstract

Classifying extreme absolute imbalanced datasets well is difficult due to the bias towards the majority class in the data. The size of the majority class is way larger compared to the size of the minority class in such datasets, which results into an underperforming binary classifier. To handle such datasets, and classify the minority class well, sampling techniques are used to transform the data to create a less bias towards the majority class. Sampling extreme absolute imbalanced datasets is even more challenging due to the lack of observations in the minority class, which may negatively impact the learnability by generating borderline or overlapping instances. In this research we created new sampling techniques which are able to handle extreme absolute imbalanced datasets. These new sampling techniques are combinations of known sampling methods which are not combined before. We constructed 12 new sampling techniques which are all combinations of the following sampling methods: SMOTE, SWIM, polynom-fit-SMOTE and Tomek Links. We evaluate ten different datasets, of which one contains information about real life applications collected from an International Financial Institution. We discovered that our created combination sampling method SWIM-POLY, which consists of the sampling methods SWIM and polynom-fit-SMOTE, performs better for the majority of the test datasets than other sampling methods used in the research. Further, we notice that our suggested sampling method SWIM-POLY works best when more than half of the synthetic instances are generated with the SWIM method. Therefore, this research recommends to use the oversampling combination SWIM-POLY for datasets which can be classified as absolute and extreme imbalanced.

Keywords: Combination Sampling Methods, Extreme Absolute Imbalanced, XGBoost, MCC

Contents

1 Introduction

Creating a well performing binary classification model involving extreme absolute imbalanced data is difficult. Due to the unbalancedness in the data, classifiers are likely to become biased towards the class with the most instances, the majority class. However, often, the interest is higher in properly classifying the minority instances over the majority instances. Instances in the minority class rarely occur in the data, in fact the Imbalanced Ratio (IR), which is the fraction of the number of instances from the majority class over the number of instances in the minority class, may become large for extreme imbalanced datasets. Moreover, imbalanced data is classified as absolute imbalanced if the number of instances in the minority class is very limited, about 1-20 observations.

In the past decades research provided different (sampling) techniques to handle imbalanced datasets. The IR of datasets of these studies are generally around 1.5 to 50, without identifying if the data is absolute imbalanced or not. The difference between this research and the literature towards the level of unbalancedness, is the difference in the level of the IR and the number of minority instances in the datasets. In this research, we face imbalanced datasets with higher IRs, with less instances belonging to the minority class than generally in the literature, which makes the datasets extreme absolute imbalanced.

Regrettably, extreme absolute imbalanced datasets often occur in real-world data, for example in cancer diagnostics, failure detection or credit requests at a financial institution. The chance an event occurs in such domains is very rare, but prediction of these events is still an important task to do. The performance of the classifier for the minority class is commonly bad as the learning method cannot learn decision boundaries well between the majority and minority class, and consequently becomes biased towards the majority class. However, despite the challenge of countering imbalanced data, it is necessary and important to classify the instances of the minority class correctly. To classify the minority class correctly, the imbalanced data needs to be treated or the methods which are used needs to be adapted, in a way that the classifier is able to recognize the data into two classes. There are three different types of procedures to handle the imbalanced data, to increase the prediction performance of the minority class.

First, algorithm-level methods create or adapt existing algorithms to increase the focus on the minority class and decrease the bias to the majority class. An example of an algorithm-level method is cost-sensitivity learning, where the penalty for misclassifications of instances from the minority class is higher than that of the majority class. Another example is majority voting, where the majority class is split in an arbitrary number of sub-samples and all combined with the whole minority class. The sub-sample which shows the highest accuracy scores and the lowest misclassifications of the minority class will be used to train the model.

Second, data-levels methods are used to rebalance the data, for example under- or oversampling techniques. Undersampling techniques remove instances from the majority class which are close to instances from the minority class in the feature space, to create a more clear boundary between both classes. An example of such an undersampling technique is Tomek Links, later referred as TOMEK. A disadvantage of undersampling is losing important information by removing valuable instances from the data. Another sampling method is the oversampling technique, where instances are generated for in the minority class to make both classes of equal size. A well-known oversampling technique is Synthetic Minority Oversampling Technique (SMOTE), which generates new synthetic minority instances between original minority instances, introduced by Chawla et al. (2002a). SMOTE was the first oversampling technique that generates new synthetic minority instances instead of weighting the original minority instances more heavily, such as the Random Oversampling method does (Solberg and Solberg (1996)). Generating new synthetic instances in the range of the original minority instances causes into a larger and less specific feature space of the minority class, which may improve the prediction performance of a classifier. The main disadvantage of SMOTE is that it may generate minority instances in the feature space of samples of the majority class, as SMOTE does not consider about the attribute values of instances belonging to the majority class, only on instances of the minority class. In total, there are at least 85 different SMOTE variants created, all compared in Kovács (2019). One of them is the polynom-fit-SMOTE variant, also referred as POLYSMOTE, which is a derivation of the original SMOTE sampling method. The new added synthetic instances are generated using Curve Fitting Methods to find the coefficients of a polynomial $p(x)$ that fits the instances from the minority class, as explained in Gazzah and Amara (2008).

Third, ensemble learning methods combine learning technologies with algorithm- or data-level methods to improve classification performances. An example is the combination of the majority vote algorithm with Adaboost, described in Randhawa et al. (2018).

Contemporary studies use imbalanced datasets and show that the aforementioned procedures may improve the classification performances. However, there is a lack of studies handling extreme absolute imbalanced datasets, compared to normally imbalanced datasets. To handle absolute imbalanced datasets, Sharma et al. (2018) proposed an algorithm: Sampling WIth the Majority (SWIM) which shows promising results compared to other sampling techniques, especially for extreme absolute imbalanced datasets. SWIM is an oversampling method and differs from other oversampling methods, such as SMOTE and ADASYN, as SWIM uses the information which is captured in the majority class. SMOTE only uses the feature space of the observations belonging to the minority class to oversample the data, without taking the information of the majority class into account. By changing the point of view to use the knowledge of the majority class over the minority class, the method is less susceptible for aberrant minority instances. The distribution of the majority class is used to generate new synthetic minority instances in the sub-spaces along the density contours of the original minority instances. SWIM is a Mahalanobis distance (MD)-based approach. The MD of each scarce instance of the minority class has a hyperelliptical density contour around the majority class. New synthetic minority instances are generated along the contours for each minority instance to rebalance the data. A clear advantage of the SWIM sampling method is that it is resistant against a division of the minority space. If the minority class is divided into multiple sub-samples, the SWIM method can treat the minority class as it is split into multiple spaces. Therefore, the risk of misclassifications is reduced, as the risk of overlapping feature spaces of both the majority and minority class becomes smaller. Meanwhile, a disadvantage of the SWIM method is that it only generates new minority instances on the contours of the hyperelliptical density contours of the majority class. This is contradictory to the procedure of SMOTE that generates a cloud of synthetic minority instances, which is an advantage of SMOTE compared to the SWIM sampling method. A disadvantage of SMOTE is that it can harm prediction performances of the classifiers with extreme absolute imbalanced datasets, as SMOTE is not able to handle multiple separated sub-samples of the minority class. Therefore, the risk of double classifications increases in the data, as SMOTE may generate new minority instances in the feature space of the majority class, by connecting the minority sub-samples.

In this research we create three new sampling techniques which are combinations of known sampling methods to explore the opportunities to improve the prediction performances of classifiers in extreme absolute imbalanced datasets. By combining the sampling methods, SMOTE, SWIM, POLYSMOTE and TOMEK, we combine the strengths of the single sampling methods whereby the prediction performance may improve. The first created sampling method is the SWIM-SMOTE oversampling method, which combines the relative strengths and advantages of both sampling techniques SWIM and SMOTE. Because of the framework, SWIM can handle the sub-samples in the minority class, and can be strengthened by the procedure of SMOTE, which creates multiple clouds of synthetic minority instances. The second created framework is the SWIM-SMOTE-TOMEK sampling method, where TOMEK removes instances from the majority class located on the borderline of the majority class and are close to minority instances. By adding TOMEK to the framework, the method creates a more clear boundary between the majority and minority class, which may improve the prediction performance. The third created sampling method is the oversampling method SWIM-POLY, which is a combination of SWIM and POLYSMOTE. Gazzah and Amara (2008) shows that the oversampling technique POLYSMOTE gives more accurate prediction performance than SMOTE for datasets with a high IR and a small size of the minority class. In all the three frameworks we start oversampling with the SWIM method, until a fraction α. This value for α is the fraction of the total number of minority instances that needs to be generated for a completely rebalanced dataset, by SWIM. We use for all the three frameworks four different values for α, to gain insight how much instances needs to be generated with which sampling method. In total, this research evaluates 12 created combination sampling methods on extreme absolute imbalanced datasets, where we expect that they outperform the single sampling methods: SMOTE, SWIM, POLYSMOTE and TOMEK.

This research has the following research questions, subdivided under a main question and two sub-questions. The main research question for this paper is defined as followed:

What is the effect of rebalancing methods on extreme absolute imbalanced datasets?

In order to answer the main question, the following sub-questions are formulated:

1. Is it possible to outperform standalone sampling methods by creating a framework which is a combination of existing single sampling techniques for extreme absolute imbalanced datasets?

2. How can we create a well performing sampling method that is a combination of over- and/or undersampling methods to handle datasets that are qualified as extreme absolute imbalanced?

The remainder of this paper is organized as follows: Section 2 presents relevant literature to broaden the knowledge of the problem of handling extreme imbalanced datasets. Next, Section 3 describes ten different datasets that are used in this research and shows their level of unbalancedness. Then, the methodology of the sampling techniques is presented in Section 4. It gives short overviews of the frameworks of the single sampling methods SMOTE, SWIM, TOMEK, and POLYSMOTE, and gives the frameworks of the combination sampling methods SWIM-SMOTE, SWIM-SMOTE-TOMEK, and SWIM-POLY. The corresponding results are presented in Section 5. Finally, Section 6 shows the conclusion of the research and give answers to the research questions.

2 Literature Review

We discuss the existing literature about several different data-level algorithms, such as over- and undersampling methods and combinations of sampling methods. Further, an overview of different algorithmic methods in the present literature is given. Lastly, we show the literature of different ensemble learning methods, which combine data-level techniques or algorithmic methods with ensemble learning techniques.

2.1 Data-Level Methods

We can define the data-level methods into three groups. First, we define the oversampling methods, where the size of the minority class is increased by generating new synthetic instances for the minority class. Then, an overview of different undersampling methods are presented, where the size of the majority group is reduced. Lastly, we show the literature about the combination of the two sampling techniques.

2.1.1 Oversampling Methods

Research on countering imbalanced data started already in the past decades. Solberg and Solberg (1996) explored the Random Oversampling method (ROS), which randomly select existing minority instances and add them with replacement to the minority class. Due to this procedure, ROS suffers from the risk of overfitting, as there is no assumption about the distribution of the minority class. In 2002, Chawla et al. (2002a) discovered a new oversampling method, namely Synthetic Minority Oversampling Technique (SMOTE). It uses the feature space of the minority class to increase the size of the minority class and make it equal to the size of the majority class. It uses the k-nearest neighbors algorithm and the distance metrics for every observation in the minority class to generate new minority instances in the dataset. In the years after the introduction of SMOTE, several extensions and alternatives are proposed of SMOTE. In 2005, Han et al. (2005) introduced Borderline-SMOTE, which adjusted the SMOTE algorithm and only focuses on instances in the minority class located at the borderline between the minority and majority class. The borderline-SMOTE sampling method only generates new minority instances on the borderline to increase the distinction of the two classes in the dataset. Another known extension of SMOTE is the Adaptive Synthetic sampling approach (ADASYN), proposed by He et al. (2008). In ADASYN a density distribution is used to come up with the number of new synthetic minority examples for every minority instance. The density distribution depends on the complexity to learn each minority instance, which is based on the ratio of examples which belong to the majority class and to the nearest neighbors. In 2008, Gazzah and Amara (2008) proposed the polynom-fit-SMOTE variant which generates instances along line segments between relatively far samples of the minority class. In the research they described four variants with different topology parameters, namely 'bus', 'star', 'mesh', and 'polynomial-curved bus'. As the variant with the 'star' parameter shows the best performance, it is set as default.

In total, 85 different extensions of SMOTE are proposed in different researches. Kovács (2019) shows a comparison of all the 85 different SMOTE performances on 104 different imbalanced datasets. They used four different classification techniques, namely Support Vector Machines

(SVM), Decision Trees (DT), multilayer perceptron (MLP) and k-Nearest Neighbors (kNN). The best overall performing SMOTE variant on the large number of datasets, with different IRs, is the polynom-fit-SMOTE sampling method. Further, Kovács (2019) shows that polynom-fit-SMOTE performs best on datasets which is highly imbalanced and has less than 30 instances in the minority class. In addition, it performs best of all variants independent of the number of attributes in the data.

In 2018, Sharma et al. (2018) proposed a new oversampling method, namely Sampling WIth the Majority (SWIM). Generally, oversampling methods use information only derived from the minority instances and omit using majority training examples. The authors of SWIM started using information from the majority class, as they mentioned that the instances of the majority class contains valuable information to use, to create a well performing oversampling method. They use the Mahalanobis Distance (MD) metric to come up with hyperelliptical density contours around the majority class for real minority instances, to generate synthetic minority instances located on the same contours. An advantage of the SWIM method is that it may classify the minority group into different sub-samples, which results in multiple sub-samples of the minority class after using the SWIM method, instead of one large minority sample, as may arise after using SMOTE. Sharma et al. (2018) show that the SWIM oversampling method performs well especially on extreme absolute imbalanced datasets, with an IR above 100. In 2020, an adjustment of the SWIM sampling method appeared from Bellinger et al. (2020), which replaces the MD-measure with a radial basis function.

2.1.2 Undersampling Methods

The most basic undersampling technique is Random Undersampling (RUS), which removes instances randomly from the majority training set. A disadvantage of this technique is that it suffers from a loss of information, according to Sharma et al. (2018). Further, Tomek Links, introduced by Tomek et al. (1976), removes instances from the majority class which are at the borderline of the majority class and comply with three criteria which belong to Tomek Links. The method uses Euclidean distances where majority instances which are closest to an instance from the minority class and vice versa are removed from the data. Another undersampling technique DBMUTE, proposed by Bunkhumpornpat and Sinapiromsaran (2017) in 2017, uses density functions to eliminate majority examples which fade the class boundary in the overlapping regions. The undersampling method Edited Nearest Neighbors (ENN), introduced by Wilson (1972), removes observations from the majority class as minimal one of the three k-nearest neighbors of an instance from the majority class can be classified as an minority instance. Because of this, ENN removes more instances from the majority class than Tomek Links. The ENN sampling method can also be used to remove instances from the minority class.

By using an undersampling method, the dataset may not become fully rebalanced after applying the sampling method, which is contradictory to oversampling methods. As stated by Fernández et al. (2018), only using an undersampling method is not a popular way of rebalancing, as undersampling techniques increase the variance of the classifier and removes important features and information. Undersampling methods are nowadays often only used in combination with an oversampling method to increase the prediction performance of the classifier.

2.1.3 Combination of Over- and Undersampling Methods

A hybrid solution is to use both an oversampling and undersampling technique by increasing the size of the minority class and decrease the size of the majority class. An example of such a combination is introduced by Jian et al. (2016), where SMOTE is combined with Random Undersampling, which gives significant improvement in performance compared with standalone techniques as SMOTE and RUS. Another promising combinations of sampling methods is the oversampling method SMOTE and the undersampling method Tomek Links, introduced by Zeng et al. (2016). Here, the minority class is oversampled and critical majority instances are removed from the dataset.

In this research we created sampling methods which are combinations of two oversampling methods and an undersampling method. The SWIM-SMOTE sampling method combines two oversampling methods to fully rebalance the dataset by increasing the number of minority instances. The SWIM-SMOTE-TOMEK uses the same principles of SWIM-SMOTE, however, after oversampling, misleading instances from the majority class are removed from the dataset. Finally, SWIM-POLY uses the SWIM sampling method and subsequently the POLYSMOTE sampling method. We create this sampling method as POLYSMOTE may outperform the SMOTE sampling method in the combination framework.

2.2 Algorithmic Methods

On the contrary of sampling techniques, where the data is manipulated to create well performing classifiers, without bias towards the majority class, there is also a technique that is method based. For example, cost-sensitive learning, introduced by Elkan (2001), penalizes misclassifications after classifying for certain classes. Hereby, the method can emphasis more on correctly classifying instances out of the minority class than for the majority class. A disadvantage of the cost-sensitive learning that is difficult to come up with an appropriate cost function when facing with complex imbalanced data. (Cheng et al. (2016)).

Feature methods include both feature selection and feature extraction procedures. Feature selection can increase the focus on features which optimizes the contrast of the borderline between the majority and minority class. Yin et al. (2013) show a feature selection algorithm, where the data is clustered into C classes where the features correlations are measured. Then, the best N features are selected based on their evaluation values. These features uses to counter the unbalancedness in the data and come up with a well performing classifier.

Another algorithmic approach is using the probability threshold moving method, which changes the decision threshold to handle a severe class imbalance. First, the model is learned from the imbalanced train set and predicted on probabilities of the test dataset. Then, the continuous is converted into a class label by using an optimal threshold value. Collell et al. (2018) propose PT-bagging, which uses bootstrap sampling in combination with threshold moving. The results showed that the model outperforms single classifiers and is comparable with other state-of-the-art algorithms.

2.3 Ensemble Learning Methods

Ensemble learning methods combine data-level techniques or algorithmic methods with ensemble learning techniques to improve classifiers' performance. For example, SMOTE-Boost, derived in Chawla et al. (2003), uses a combination of the SMOTE algorithm and Adaboost to leverage SMOTE each step to compensate better for skewed distributions. Another ensemble learning method is a combination of data-level techniques and Bagging, explored in Sun et al. (2015). Here, the imbalanced data is split into multiple balanced subsets and uses several classifiers to create a whole classification algorithm. These multiple classifiers are combined into an ensemble classifier to classify new data.

Another way of using ensemble learning is combining it with algorithm-level methods, such as BPSO-ADABOOST-KNN, developed by Haixiang et al. (2016). They integrate feature selection and Adaboost into ensemble, where kNN is the basic classifier. The difference in performance is not significant between the ensemble model and other state-of-the-art algorithms. Yijing et al. (2016) explored an ensemble algorithm which combines data-level and algorithm techniques with ensemble learning to rebalance multiclass data. The adaptive multiple classifier system, named as AMCS, uses three components: feature selection, resampling and ensemble learning. The results of AMCS show that the ensemble method is comparable with other state-of-the-art algorithms.

The main disadvantage of ensemble learning methods is that is very time-consuming. Further, ensemble learning methods encounter problems with the rising dimensionality in the data, which often belongs to big data.

3 Data

In this research we use ten different datasets to test the combination sampling methods SWIM-SMOTE, SWIM-SMOTE-TOMEK and SWIM-POLY. The outcomes of these methods will be compared with outcomes of other single sampling methods, such as SWIM, SMOTE, TOMEK, and POLYSMOTE. All datasets differ in the number of attributes, the IR value, the majority size and the minority size. Here, the IR value is calculated by

$$IR = \frac{N_-}{N_+} \tag{1}$$

where N_- is the number of observations in the majority class (negatives) and N_+ is the number of observations in the minority class (positives). For each dataset, the minority class is artificially down-sampled to a size of 10 instances, to comply on the criteria of extreme absolute imbalanced data. These minority instances are randomly picked from the complete minority class. We do this procedure 2000 times for each dataset, to prevent for misleading outcomes, and we take the average values of the evaluation criteria used in this research, described in Section 4.4. Due to this randomly down-sampling of the minority class, the classifiers‘ outcomes for the obtained datasets can not be compared with other outcomes of classifiers in other studies. Table 3.0.1 gives an overview of all the different datasets used in this research. Here, the IR is calculated after down-sampling the minority class. Further information about the experimental set-up of this research is given in Section 4.3.

Table 3.0.1: Overview of different datasets with minority size of 10

Dataset	Name	Attributes	IR	Majority size
D1	Abalone	8	273.0	2,730
D2	Car	6	159.4	1,594
D3	Page	10	491.3	4,913
D4	Penbased	16	99.4	994
D5	Shuttle	9	182.9	1,829
D6	Spambase	57	460.1	4,601
D7	Thyroid	21	68.2	682
D8	Vehicle	18	62.8	628
D9	Yeast	8	144.0	1,440
D10	IFI	121	300.0	3,000

Five datasets are obtained from the UCI Machine Learning Repository, while four datasets are retrieved from Knowledge Extraction based on Evolutionary Learning (KEEL). The IFI dataset is obtained from a International Financial Institution (IFI), and is handled separately as this dataset contains real life applications compared to the other datasets, which are often used in many studies. All ten datasets show different levels of imbalance, and due to the limited number of instances in the minority class, namely 10, all datasets are extreme absolute imbalanced. The upcoming subsections show insights in the different datasets, which are divided among the datasets' sources.

3.1 KEEL Repository

Four different datasets are obtained from the KEEL Repository, namely *Penbased*, *Shuttle*, *Thyroid*, and *Yeast*. The dataset *Penbased* is a version of the Pen-Based Recognition of Handwritten Digits dataset, where the target variable is equal to one for the handwritten digit eight, and zero for others. The dataset contains 16 different explanatory variables. The IR is 99.4 with a majority class size of 994 instances. The dataset *Shuttle* gives insight which type of control of a spacecraft should be employed. In the dataset, we target the value *high* which is equal to the value one in the y-variable. The target values *Rad Flow* and *Bypass* are targeted with value zero into the target variable. The data contains 9 explanatory variables, and has an IR of 182.9. The number of instances in the majority class is 1829. The dataset *Thyroid* gives insight in the Thyroid disease, where the target variable is equal to one to values of *hyperfunction*. A value zero is appointed to the target variable for normal and subnormal functioning thyroids. The IR is 68.2 due to a majority class size of 682 instances. The dataset contains 21 different explanatory variables. The dataset *Yeast* shows the prediction of cellular localizations sites of proteins. Here, the target variable is equal to one if the outcome has value *ME1*, and zero others. The dataset contains 8 different explanatory variables. The IR is 144, and the data contains 1440 instances in the majority class.

3.2 UCI Machine Learning Repository

This research uses five different datasets available from the UCI Machine Learning Repository, namely: *Abalone*, *Car*, *Page*, *Spambase* and *Vehicle*. The dataset *Abalone* shows the prediction of the age of abalones. The target variable has value one if the prediction of the age is equal or higher than 20 years. The target variable has a value zero for age predictions of 10 years or less. The dataset contains 8 different explanatory variables, all physical measures, such as sex, length, diameter and different weights of the abalone. The IR is 273, and the size of the majority class is 2730. Further, the dataset *Car* is particularly useful for testing constructive induction and structure discovery methods. The target variable has value one for target outcomes *good* and *vgood*. The target variable has value zero for target outcomes *unacc* and *acc*. The dataset contains 6 different explanatory variables. Examples of variables are: safety, doors, and persons. The IR value of the dataset is 159.4 and the majority class contains 1594 instances. The dataset *Page*, also known as the Page Blocks Classification Dataset, is useful for classifying blocks of a page layout of a document that has been detected by a segmentation process. The target variable is equal to one if the outcome of the classification is one, and zero others. The data contains 10 different explanatory variables, such as: height and length of the block, area of the block, the percentage of black pixels and the number of pixels. The IR value is 491.3 and the size of the majority class is 4913 instances. The dataset *Spambase* gives insight in emails and if they can be classified as spam or not. If the data is classified as spam, the target variable has value one, and zero others. The data contains 57 different explanatory variables, which shows the number of times a specific word, where a word belongs to one variable, appears in the document. Examples of words are *pay*, *link*, *web*, words that are often used in spam emails. The IR of the *Spambase* dataset is 460.1 and the majority class contains 4601 instances. The dataset *Vehicle* got information about vehicles, if a vehicle is a bus, saab, van or opel. The target variable has value one if the target outcome is equal to 'bus', zero others. The data contains 18 different explanatory variables. The IR value of the data is 62.8 and the size of the majority class is 628.

3.3 IFI Dataset

The last dataset used in this research is obtained from an IFI. The data targets potential clients who are likely to receive a financial credit from the financial bank. The IFI has a large dataset at their disposal with already clients at the bank. The IFI is interested in which clients they can cross-sell a financial credit. If the bank knows which clients are targeted as potential clients, these clients are getting more interest and attention of the bank. The data contains 121 variables and shows information about the characterizes of the client. The number of clients who are not likely to receive a financial credit is 421,000. The dataset contains 960 positive responses for clients who are likely to receive a credit. In this research we take a subset of the data of both the majority class and the minority class, respectively of size 3000 and 10, to obtain a number of instances which is around the sizes of other datasets used in this research. We take both subsets 2000 times to avoid biased prediction performances due to the potential of large variances of the subsets caused by the small number of both sets. As this dataset contains information about real life applications, we have handle the dataset separately and have more interest in the working of the created combination sampling techniques.

4 Methodology

In this section, we describe the single sampling methods and the new created combination sampling methods which are used in this research. First, the SMOTE technique is described. Then, the we have a closer look to the SWIM method. Further, we clarify the POLYSMOTE sampling method, especially where it differs from the SMOTE sampling method. Moreover, the undersampling technique TOMEK is described. Next, we elaborate on the combination sampling methods which are the contribution of this research. The first sampling method combines the sampling methods SWIM and SMOTE and creates the combination method: SWIM-SMOTE. Moreover, we add to the framework the undersampling method TOMEK to construct a new framework: SWIM-SMOTE-TOMEK. Further, we look to the combination of SWIM and POLYSMOTE, which creates SWIM-POLY. Lastly, insights in the experimental set-up is provided, along with the evaluation criteria used in this research.

4.1 Single Sampling Methods

In this research, we make difference between single and the self-created combination sampling methods. Single sampling methods are sampling techniques which are known in the literature and for users who face imbalanced datasets. The single sampling methods used in this research are SMOTE, SWIM, POLYSMOTE, and TOMEK. Thereafter, this research created sampling methods which are combinations of single sampling methods, therefore called combination sampling methods. First, we explore the SMOTE oversampling method. Then, we clarify the SWIM oversampling method. Next, we explore the POLYSMOTE oversampling method, and lastly the undersampling method TOMEK is explained.

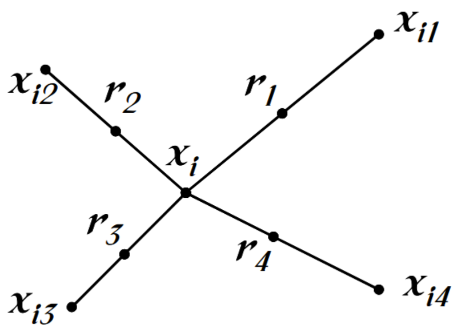
4.1.1 SMOTE

The Synthetic Minority Oversampling TEchnique (by Chawla et al. (2002b)), also known as SMOTE, is an oversampling technique which generates new synthetic minority instances in the neighborhood of the already existing minority instances in the data. The method interpolates synthetic minority instances between the k-nearest neighbors for each original minority instance. As the method only generates new minority instances in the defined neighborhood of the minority class, the arising generated minority feature space gets a higher density of real and synthetic minority instances. By generating new synthetic minority instances, the data is getting rebalanced, which creates a less biased predictor, which may reduce the number of misclassifications by classifying the data.

Figure 4.1.1 illustrates the way of working of the SMOTE sampling method. Here, the original minority examples are the x-points, and the r-points are the new synthetic minority examples. In the example, the parameter k is equal to 4, as four minority instances $(x_{i1}, x_{i2}, x_{i3}, x_{i4})$ are linked with the original instance (x_i). The synthetic minority instances (r_1, r_2, r_3, r_4) are generated on the line segments created between the k nearest neighbors of x_i.

The oversampling technique SMOTE can be divided into several steps. Algorithm 1 shows the framework of the single oversampling method SMOTE. The SMOTE method has as input values the minority sample B, the number of minority instances to generate n, and the number

Figure 4.1.1 An illustration of how to create the synthetic data points in the SMOTE algorithm



of neighbors used in the model k. In line 1, we calculate the number of minority instances in the dataset. Then, in line 3 and 4, we compute the k nearest neighbors of each instance located in B. For each original instance, we calculate the number of instances to generate to come up to the level of n. In line 7 we randomly pick a neighbors of the set of neighbors of the original instance, and in line 8 we calculate the difference in feature values of the original instance and its neighbor. In line 9, we take a random number between zero and one with an uniform distribution, and multiply it with the difference value plus the features value of the original instance in line 10. In line 11 we combine the new created minority instance to the self-made minority sample. In line 13, we combine the two samples to create the new rebalanced minority class.

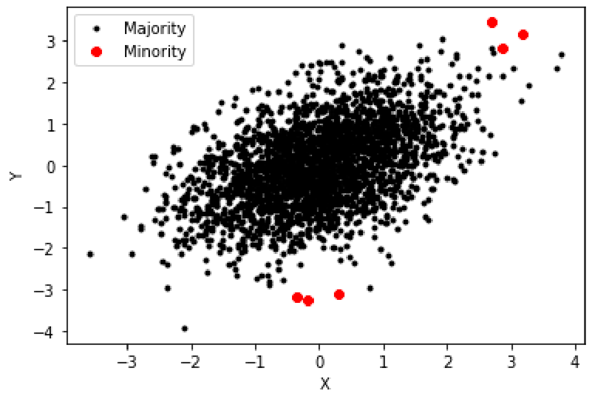
Algorithm 1 SMOTE Framework (from Fernández et al. (2018))

```
1. Function SMOTE ($B$, $n$, $k$)
Input:
$B$, minority class
$n$, number of minority instances to generate
$k$, number of neighbors in the SMOTE method
Output:
$B$; minority class with original and new synthetic minority instances
Method:
2. $d = |B|$, number of instances in the minority class
3. for $i$ = 1 to $d$ do
4.     Compute $k$ nearest neighbors for instance $i$, and save the indices
5.     $N = \text{int}\left(\frac{n}{|d|}\right)$
6.     while $N \neq 0$ do
7.         $B_{il}$; randomly pick neighbor $l$ of the neighbors of instance $B_i$
8.         $D_{il} = B_{il}$ - $B_i$; difference between instance
9.         gap; random number between $Uniform(0, 1)$
10.        $b''_{il} = B_i + \text{gap} \times D_{il}$; generate new minority instance
11.        $B' = (B' \cup b''_{il})$; append $b''_{il}$ to the synthetic minority sample
12.        N = N - 1
13. $B = (B \cup B')$; combine the original minority class with the synthetic minority sample
Return $B$
```

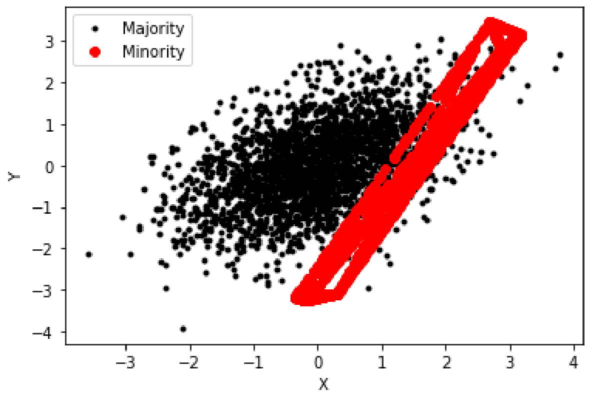
The advantage of the SMOTE sampling method is that it generates new minority instances between the original minority instances, which creates a cloud of minority points after resampling. By rebalancing the data, the number of instances in the majority class becomes equal to the number of instances in the minority class, which reduces the bias of the classifier to the majority class.

The disadvantage of this procedure comes up when the size of the minority class is very limited, with around 10 instances, and split into multiple sub-samples, which are highly independent of each other. If SMOTE is applied in these circumstances, occurring with extreme absolute imbalanced data, the minority sub-samples are becoming correlated with each other as SMOTE generates new instances between these sub-samples. This can be problematic, due to the rising change of double classification areas in the data. Namely, synthetic minority instances are generated on the line segments between the original minority instances, independent on the presence of instances from the majority class. If these original sub-samples of the minority class are less correlated, these sub-samples might be far away from each other, with a large chance of majority instances *in between*. Figure 4.1.2a shows an example of an extreme absolute imbalanced dataset without the use of any sampling method. Clearly, the minority instances are located into two different spaces. 4.1.2b shows the same imbalanced dataset, however, the minority data is oversampled with the oversampling technique SMOTE.

Figure 4.1.2 An illustration how SMOTE creates new instances in a dataset



(a) Extreme absolute imbalanced dataset without sampling technique



(b) Extreme absolute imbalanced dataset with SMOTE

As Figure 4.1.2a illustrates, the minority instances are divided into two sub-samples. If we apply the SMOTE oversampling technique to this specific dataset, synthetic minority instances will be generated in the same feature space of the instances from the majority class. In other words, after sampling the data, a large share of the feature space is doubly eligible, as minority and majority class. These doubly qualifications raises the number of misclassifications of the classifier, which reduces its predictive power. Therefore, using the sampling technique SMOTE is not recommendable for extreme imbalanced datasets due to the chance of a reduction of the classifiers' prediction performance. To counter for the shortcomings of SMOTE, SWIM is conceived which is able to handle the minority data into several sub-spaces.

4.1.2 SWIM

In this subsection we describe the structure and (dis)advantages of the Sampling WIth the Majority (SWIM) sampling method, by Sharma et al. (2018). First, we provide insights in the Mahalanobis distance method, which is used in the SWIM framework. Then, we discuss the steps of the SWIM Algorithm and the methods' advantages and disadvantages.

In the SWIM framework we use the Mahalanobis Distance (MD) metric to calculate the distance between the mean of the distribution of the majority class and a single instance from the minority class. In the dataset, the mean of the majority class is the same for every minority instance. So, by calculating the MD between these points, we create a variable to compare singular minority observations. Minority instances which have the same MD to the mean of the majority class, are on the same hyperelliptical density contour around the majority class. To calculate the MD, the values for μ and Σ of the majority class are required. Here, μ is the mean value for every feature f in the majority class. Σ is the covariance matrix of the majority class. However, calculating the values of μ and Σ for the majority class is difficult to do, due to the chance of occurrence of linear dependent columns in the majority dataset. In addition, the computing time of these parameters can become extremely long. Because of this, the MD uses an estimate of the parameters: $\bar{\mu}$ and $\bar{\Sigma}$, which are both estimated on a sub-sample from the majority class. Equation 2 shows the MD

$$MD\big(x, \bar{\mu}, \bar{\Sigma}\big) = \big(x - \bar{\mu}\big)^T \bar{\Sigma}^{-1} \big(x - \bar{\mu}\big), \tag{2}$$

where x is an instance in the minority class and $\bar{\mu}$ and $\bar{\Sigma}$ are the mean and covariance of the subsample of the majority class, respectively.

The difference between SWIM and other oversampling methods is that SWIM uses information from the majority class to generate new minority synthetic instances. As described above, the distribution measures of the majority class are used to calculate the MD of the original minority instance to the mean of the majority sample. By using the standard deviation σ_f of feature f, an interval is created wherefore new synthetic minority instances are generated on the same MD as the real minority instance f. To change the interval of minority instances which needs to be generated, a variable γ is used to manipulate the spread of the minority instances along the density contours of the majority class. In this research we set γ equal to 0.5 for every f, as the value 0.5 is the most common in the literature. Beside, if the value for γ becomes large, the spread of the synthetic minority instances along the density contours increases, which may reduce the prediction performance. If the value of γ is too low, the new minority instances are to close to each other, which also may reduce the prediction performance of a classifier.

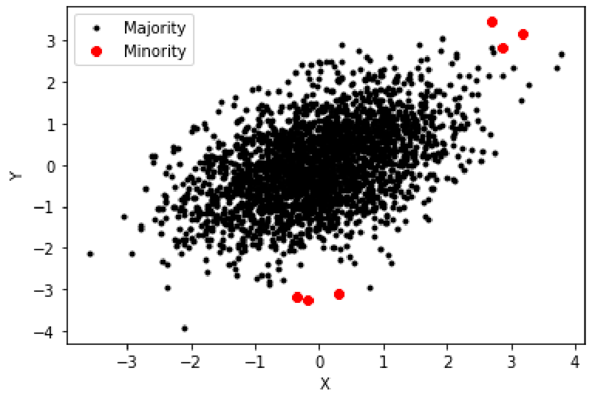
Algorithm 2 shows all the steps of the SWIM framework. In line 1, we calculate the number of minority examples which needs to be generated, to rebalance the dataset completely. In line 2-4, we centre the minority and majority classes with the mean of the majority class, so that the majority class gets a mean value of zero. In line 5 and 6, we whiten the minority class which simplifies the calculations for generating synthetic data. Namely, in a whitened space of a distribution, the MD becomes equivalent to the Euclidean distance. In line 7-11, we obtain the bounds of the new generated minority examples by creating ranges of values for every feature in the whitened minority dataset. We found for every feature f its mean and standard deviation. Here, $\gamma \in \mathbb{R}$ controls the deviation of the range of the bounds of the new minority examples. The larger the value for γ, the wider the range, the larger the level of spread along the density contours. In line 11, a while loop is started and stops until the number of new generated instances becomes zero. In line 12 and 13, for every new synthetic instance, we pick a real minority example b_i at random, and define for every feature its value between the lower- (l_f) and upper bound (u_f). Then, in line 14, the new synthetic instance is transformed to the original space instead of the whitened space. The new synthetic example $b_i^{''}$ is added to the new minority class $B^{'}$. We repeat this until a fully rebalanced dataset is created. Afterwards, in line 17, we combine the two minority samples into one minority class, which contains the original minority instances and the created minority instances.

Algorithm 2 SWIM Framework (from Sharma et al. (2018))

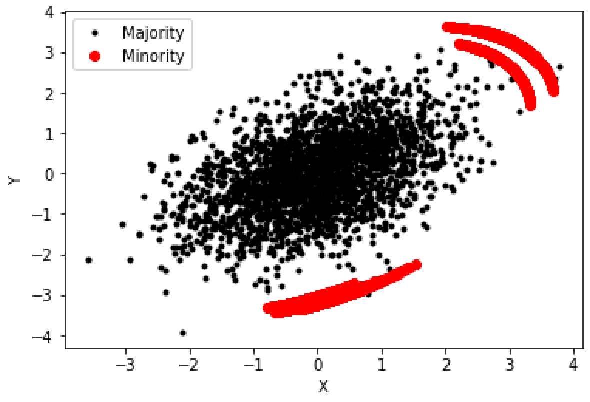
```
1. Function SWIM($A$, $B$, $\gamma$, $n$)
Input:
$A$, Majority class
$B$, Minority class
$\gamma$, controls the spread of the synthetic samples along the density contours ($\gamma \in \mathbb{R}$)
$n$, number of minority instances to generate
Output:
$B$; minority class with original and new synthetic minority instances
Method:
2. Calculate $\mu_A$; the mean vector of majority class A
3. $A_c = A - \mu_A$; centre the majority class
4. $B_c = B - \mu_A$; centre the minority class
5. Calculate $\Sigma^{-\frac{1}{2}}$; here, $\Sigma$ is the estimated covariance matrix of $A_c$
6. $B_w = B_c\Sigma^{-\frac{1}{2}}$; whiten the centered minority class
7. for feature $f$ in $B_w$ do
8.     Find the mean $\mu_f$ and standard deviation $\sigma_f$
9.     Calculate $u_f = \mu_f + \gamma\ \sigma_f$; upper bound
10.    Calculate $l_f = \mu_f - \gamma\sigma_f$; lower bound
11. while $n \neq 0$
12.    Select $b_i$ from minority class at random
13.    Randomly generate sample point $b_i^{'}$ where $l_f \leq b_{i,f}^{'} \leq u_f\ \forall\ f$
14.    Transform $b_i^{'}$ to $b_i^{''}$, where $b_i^{''} = (\Sigma^{-\frac{1}{2}})^{-1}\ b_i^{'}\frac{\|b_i\|_2}{\|b_i^{'}\|_2}$
15.    $B^{'} = (B^{'} \cup b_i^{''})$; append $b_i^{''}$ to the synthetic minority sample
16.    $n = n - 1$
17. $B = (B \cup B')$; combine the original minority class with the synthetic minority sample
 Return $B$
```

Figure 4.1.3b illustrates the way of working of the oversampling technique SWIM, compared to the original dataset in 4.1.3a, also used in 4.1.2a. As SWIM only generates new synthetic minority instances along the density contours of the majority class, the SWIM method has its pros and cons. Clearly, the minority data in 4.1.3a is split into two different sub-samples. As Figure 4.1.3b shows, SWIM is able to recognize and treat the minority data as being two sub-samples, even after rebalancing the data. Hereby, there will be minimal overlap between the two classes in the data which may reduces the number of misclassifications that increases the prediction performance of the classifier. The disadvantage of the SWIM method is that it generates only new synthetic minority instances along the lines of the density contours, which results in a sharp feature space of the minority class.

Figure 4.1.3 An illustration how SWIM creates new instances in a dataset



(a) Extreme absolute imbalanced dataset without sampling technique



(b) Extreme absolute imbalanced dataset with SWIM

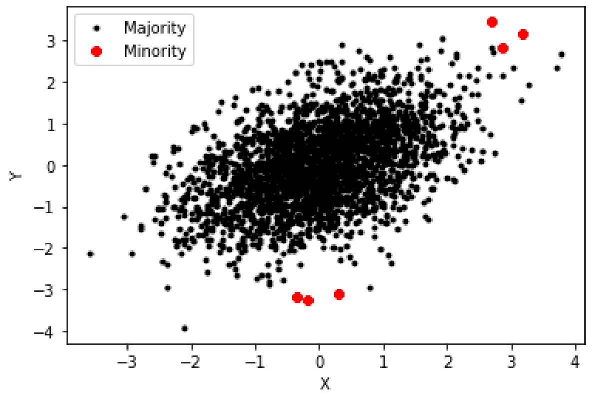
4.1.3 POLYSMOTE

Polynom-fit-SMOTE, also known as POLYSMOTE, is an adjustment to the original SMOTE sampling method. Kovács (2019) showed that the POLYSMOTE sampling method performs better for extreme imbalanced datasets, than the original SMOTE sampling method. The POLYSMOTE sampling technique uses a Curve Fitting Method to find the coefficients of a polynomial $p(x)$ of degree n that fits the minority instances. The synthetic minority instances are generated with the star topology, where the instances are arranged as a star silhouette.

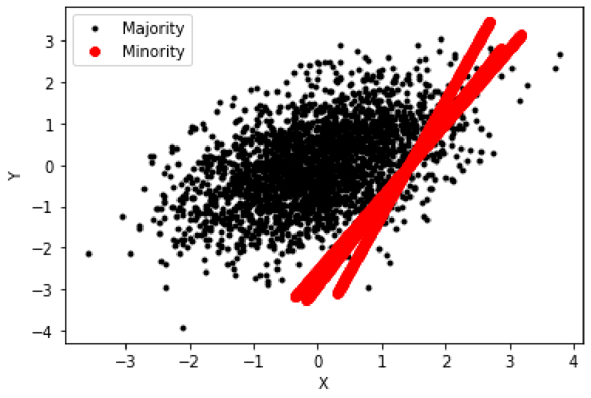
The sampling method POLYSMOTE with the star topology uses the mean value of all the features of the minority class dataset. Thereafter, the method generates new synthetic minority instances between the original minority instance and the mean value. The location of the synthetic minority instance is dependent of the oversampling rate, which is a metric to distribute the synthetic instances evenly between the mean and the minority instance. Due to this procedure, the data is generated into a star silhouette.

Figure 4.1.4a shows the original dataset, also used for the SMOTE and SWIM methods. Figure 4.1.4b shows the dataset after using the POLYSMOTE sampling technique with the topology *star*. As can be seen is that the minority dataset is arranged as a star silhouette. By using the POLYSMOTE oversampling technique, we still create double classification regions, but these regions might be smaller than for the SMOTE sampling technique, as the *star* topology causes into a smaller spread of synthetic instances. The sampling method has as option to the user if the dataset has to be become fully rebalanced or not.

Figure 4.1.4 An illustration how POLYSMOTE creates new instances in a dataset



(a) Extreme absolute imbalanced dataset without sampling technique



(b) Extreme absolute imbalanced dataset with POLYSMOTE

Algorithm 3 shows the framework of the POLYSMOTE sampling method. In line 2, we calculate the mean values of the features in the minority class B and in line 3 we calculate the number of data points located in the minority class. Then, we determine the value of k, which is the multiply of the number of instances in the minority class to rebalance the data. The value of s contains information to which extent the data must be rebalanced. Thereafter, for every minority instance in B, we calculate in line 6 the difference between the instance and the mean values. In line 8, we generate a new synthetic minority instance which is dependent of the oversampling rate and the difference between the original minority instance and the mean value. In line 9-10 we append the simulated minority instances to the original minority class.

Algorithm 3 POLYSMOTE Framework

```
1. **Function** POLYSMOTE($B$, $s$)
**Input:**
$B$, Minority class
$s$, number of minority instances to generate by the POLYSMOTE sampling method
**Output:**
$B$; minority class with original and new synthetic minority instances
**Method:**
2. $\bar{B}$ = mean of features in class $B$
3. $m = |B|$; number of data points in the minority class
4. $k = \max\left(1, \text{int}\left(\frac{s}{m}\right)\right)$
5. **for** $x$ in $B$ **do**
6.     $diff = \bar{B}$ - $x$
7.     **for** $i$ in $[1, k]$ **do**
8.         $sample = \left(x + \frac{float(i)}{(k+1)} \times diff\right)$
9.         $B' = (B' \cup sample)$; append $sample$ to the minority class
10. $B = (B \cup B')$; combine the original minority class with the synthetic minority sample
 **Return** $B$
```

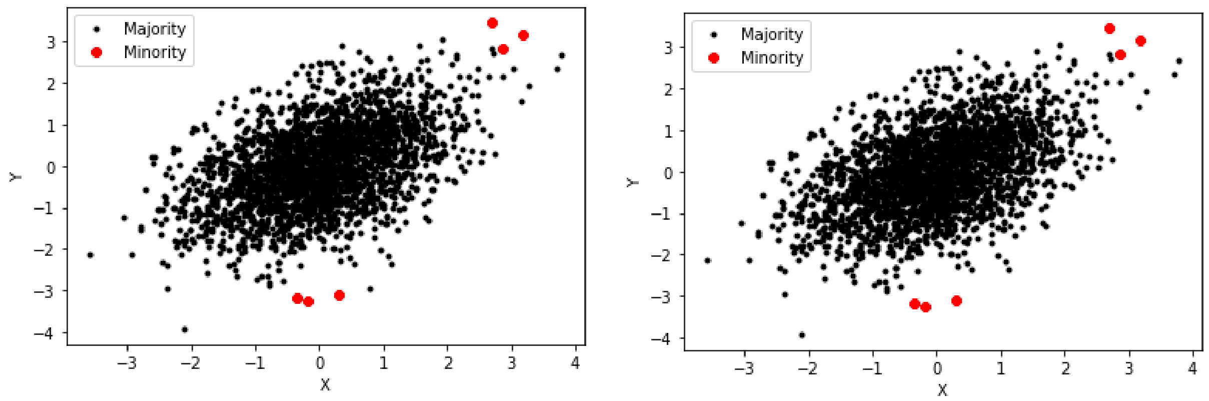
4.1.4 TOMEK

Tomek Links, introduced by Tomek et al. (1976), also represented as TOMEK, is the only undersampling technique in this research. Undersampling methods reduces the number of instances in the majority class to rebalance the data. A pair of observation (a, b) is a Tomek Links if and only if all the following three criteria are met:

- The nearest neighbor of a is b
- The nearest neighbor of b is a
- Observation a and b belong to different classes.

If (a,b) is classified as a Tomek Link, the observation in the pair belonging to the majority class is removed from the dataset. By removing these majority instances, the border between the majority and minority class becomes more clearly, which may increase the prediction performance of the classifier. The undersampling technique Tomek Links does not fully rebalance the data, as SMOTE and SWIM do, but only removes observations that met the three criteria described above.

Figure 4.1.5 An illustration how TOMEK removes instances from the majority class



(a) Extreme absolute imbalanced dataset without sampling technique

(b) Extreme absolute imbalanced dataset with TOMEK

Figure 4.1.5 shows the process of the undersampling method Tomek Links. As can be seen in 4.1.5b, the effect of the undersampling method is limited. Therefore, the undersampling technique TOMEK is often not used as a standalone method but in combination with an oversampling method. Another disadvantage of the Tomek Links method, and other undersampling methods in general, is that the method throws observations from the majority class away which may contain useful information and insights.

The algorithm of the undersampling technique Tomek Links is provided in Algorithm 4. In line 2 we combine the two classes into one dataset. In line 3 - 6, the TOMEK undersampling method is described. In line 3, we pick two instances x and y from the dataset, which is a combination of the two classes A and B. In line 4, we check if x is the nearest neighbor of y and y is the nearest neighbor of x. If this is the case, then we check in line 5 if one of the instances is in class A, and one in class B. If this criteria is met, then the instance from the majority class is removed from the dataset in line 6.

Algorithm 4 TOMEK Framework

```
1. Function TOMEK($A$, $B$)
Input:
A, Majority class instances.
B, Minority class instances.
Output:
$A$, $B$; new majority and minority sample, respectively
Method:
2. data = ($A \cup B$); combine majority and minority data
3. for instance $x$, $y \in$ data do
4.     if (nearest neighbor of $x$ is $y$) and (nearest neighbor of $y$ is $x$) then
5.         if ($x \in$ A and $y \in$ B) or ($x \in$ B and $y \in$ A) then
6.             A = A' ; Remove $x$ or $y$ from majority class A
 Return $A$, $B$
```

4.2 Combination Sampling Methods

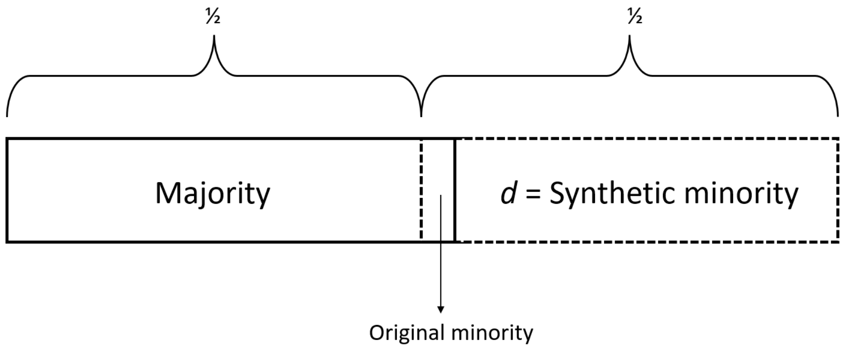
The contribution of this research is providing combination sampling methods, which are combinations of the single sampling methods SWIM, SMOTE, TOMEK and POLYSMOTE. Next, we clarify the combination sampling methods used in this research, namely the combination of SWIM and SMOTE into SWIM-SMOTE, which is expanded with TOMEK to create the sampling method SWIM-SMOTE-TOMEK. Finally, we express the sampling method SWIM-POLY, which combines the single sampling methods SWIM and POLYSMOTE.

4.2.1 SWIM-SMOTE Framework

First, we combine the aforementioned sampling methods SWIM and SMOTE. To rebalance extreme imbalanced datasets, we first use the SWIM method to increase the number of minority instances by creating new synthetic instances located on the same MD as the real minority data. We start with the SWIM method, which is able to take multiple sub-spaces of the minority data into account. After applying the SWIM method, we consider the original and generated minority instances as original data for the minority class for the SMOTE sampling method. By using SMOTE, we generate synthetic instances only in the spaces where SWIM created minority instances. We follow SWIM up with SMOTE to reduce the sharpness of the feature space of the instances in the minority class. The feature space of the minority class will increase. Figure 4.2.1 schematically shows the number of synthetic instances d to generate in the SWIM-SMOTE combination to rebalance the data completely. This illustration applies for all the oversampling methods where the dataset becomes fully rebalanced. By generating d number of minority instances, the size of the majority class becomes equal to the size of the minority class.

By applying the SWIM-SMOTE combination sampling method, we alternate the sampling method SWIM with SMOTE for different values of α. These values for α are fractions of the whole yet to be generated synthetic minority set d. Thus, the combination oversampling method SWIM-SMOTE$\left(\alpha = \frac{1}{5}\right)$ generates first $\alpha \times d$ synthetic minority instances with the SWIM method, and thereafter, SMOTE generates (1 - α) $\times$ d synthetic instances to rebalance the data completely. By doing this we create a rebalanced dataset which can be used for training the classifiers which becomes less biased towards the majority group. In this research we opt for different values of α, namely $\alpha = (\frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5})$. To investigate which value of α gives the most promising results, we investigate the framework for every dataset for all the different values of α. Further, we analyze the standalone sampling methods SMOTE and SWIM, which corresponds to the SWIM-SMOTE framework with an α value of zero and one, respectively.

Figure 4.2.1 Schematic display of the number of instances to generate in the minority class



Algorithm 5 shows the SWIM-SMOTE framework, which is a combination of the SWIM Algorithm 2 and the SMOTE Algorithm 1. The framework has as inputs variables the majority class dataset A, the minority class B, the variable γ which shows the level of spread of the synthetic minority instances along the density contours in the SWIM method. Further, α shows the fraction of the total number of instances to be generated by SWIM, and k, which is a variable from the SMOTE sampling method which are the number of neighbors used to generate new synthetic minority samples.

Algorithm 5 SWIM-SMOTE Framework

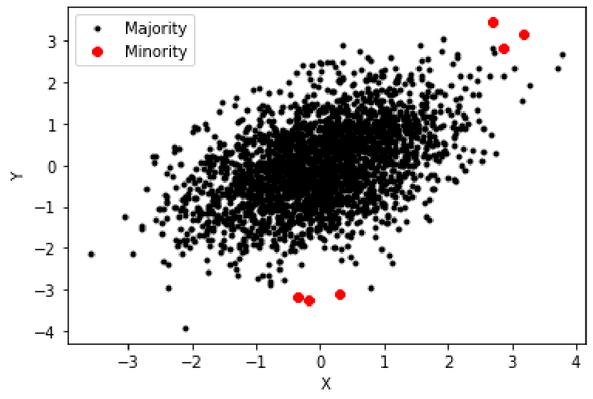
```
1. Function SWIM-SMOTE($A$, $B$, $\gamma$, $\alpha$, $k$)
Input:
A, Majority class
B, Minority class
$\gamma$, spread of the generated synthetic minority instances along the density contours ($\gamma \in \mathbb{R}$)
$\alpha$, Ratio change from SWIM to SMOTE ($\alpha = \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}$)
$k$, number of neighbors in the SMOTE method
Output:
$B$; minority class with original and new synthetic minority instances
Method:
2. $d$ = |A| - |B|, number of minority samples to generate to fully rebalance the dataset
3. $n = \lfloor \alpha \times d \rfloor$, number of instances generated by SWIM oversampling method
4. $B$ = SWIM($A$, $B$, $\gamma$, $n$), see Algorithm 2
5. $s = \lceil (1 - \alpha) \times d \rceil$, number of instances to generate by SMOTE
6. $B$ = SMOTE($B$, $s$, $k$), see Algorithm 1
 Return $B$
```

In line 2 we calculate the size of the variable d, the total number of instances to generate in the minority class to rebalance the data completely. Then, in line 3, we determine the number of instances to generate by the sampling method SWIM, where we round up the number to its greatest smallest integer. In line 4 we apply the SWIM sampling method, and in line 5 we decide the number of instances to be generated by SMOTE. Here, we round the number to its least greater integer. Thereafter, we apply the SMOTE sampling method and we return the new minority class.

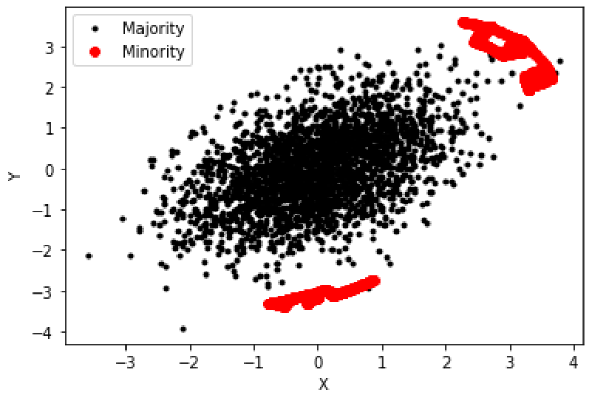
The advantage of the framework SWIM-SMOTE is that the sampling framework uses the strengths of both sampling methods. The SWIM sampling method is able to handle several smaller sub-sets in the minority class well, but does not create a cloud of points for every sub-set, but only 'lines of data' on the density contours of the majority class. By switching from SWIM to SMOTE, we create several cloud of new synthetic minority instances for every sub-sample in the heavily imbalanced dataset. Due to the combination of both strengths, the hypothesis is that the SWIM-SMOTE framework can outperform the standalone sampling methods SWIM and SMOTE.

Figure 4.2.2 shows the procedure of the SWIM-SMOTE combination sampling method. As can be seen in 4.2.2b, the two sub-samples of the minority class are treated as two separate samples of the minority class. First, the SWIM method is used, thereafter, the SMOTE sampling method is applied. As a result of this procedure, two *cloud* of data points are generated, which may increase the prediction performances of a classifier.

Figure 4.2.2 An illustration how SWIM-SMOTE creates new instances in a dataset



(a) Extreme absolute imbalanced dataset without sampling technique



(b) Extreme absolute imbalanced dataset with SWIM-SMOTE

4.2.2 SWIM-SMOTE-TOMEK Framework

The SWIM-SMOTE-TOMEK sampling framework is an extension of the SWIM-SMOTE framework. The procedure of the oversampling technique is the same as for the SWIM-SMOTE framework. After oversampling the minority instances, the TOMEK undersampling method removes all the Tomek Links from the dataset. These links are classified as Tomek links as the data satisfies the three requirements, described in 4.1.4. The advantage of SWIM-SMOTE-TOMEK framework is that the method makes the boundaries between the two classes more clear, which may increase the prediction performance of the classifier.

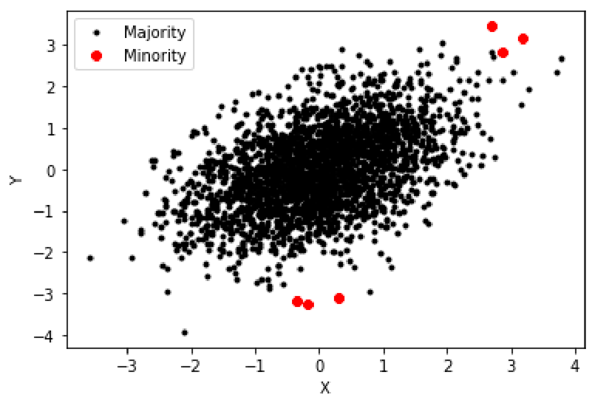
Algorithm 6 shows the framework of the SWIM-SMOTE-TOMEK sampling method. In line 1 - 6, the framework is the same as for the sampling method SWIM-SMOTE. In line 7, we add the undersampling technique TOMEK to the framework, which is defined in the Algorithm 4. The SWIM-SMOTE-TOMEK returns both the majority class as the minority class dataset, as both sets can be adjusted by the methods.

Algorithm 6 SWIM-SMOTE-TOMEK Framework

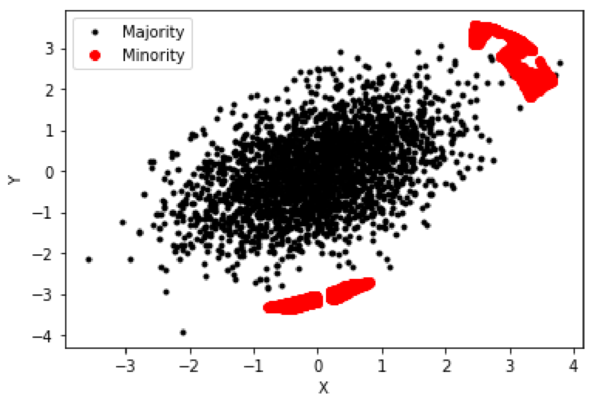
```
1. Function SWIM-SMOTE-TOMEK($A$, $B$, $\gamma$, $\alpha$, $k$)
Input:
A, Majority class instances.
B, Minority class instances.
$\gamma$, spread of the generated synthetic minority instances along the density contours ($\gamma \in \mathbb{R}$)
$\alpha$, Ratio change from SWIM to SMOTE  ($\alpha = \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}$)
$k$, number of neighbors in the SMOTE method
Output:
$A$, $B$; new majority and minority sample, respectively
Method:
2. $d$ = |A| - |B|, number of minority samples to generate to fully rebalance the dataset
3. $n = \lfloor \alpha \times d \rfloor$, number of instances generated by SWIM oversampling method
4. $B$ = SWIM($A$, $B$, $\gamma$, $n$), see Algorithm 2
5. $s = \lceil (1 - \alpha) \times d \rceil$, number of instances to generate by SMOTE.
6. $B$ = SMOTE($B$, $s$, $k$), see Algorithm 1
7. $A$, $B$ = TOMEK($A$, $B$), see Algorithm 4
 Return $A$, $B$
```

Figure 4.2.3 illustrates the process of the SWIM-SMOTE-TOMEK sampling method. The main difference is the removal of an instance from the majority class on the right of the lower sub-sample of the minority class. Due to this removal, the boundary between the two classes becomes more clear, which may increase the prediction performance. However, the number of removed instances from the majority class is limited, which indicates that the criteria, appointed in 4.1.4, are too strict. Further, the SWIM algorithm contains a random function, which explains the difference of the minority classes between Figure 4.2.3b and Figure 4.2.2b.

Figure 4.2.3 An illustration how SWIM-SMOTE-TOMEK creates new instances in a dataset



(a) Extreme absolute imbalanced dataset without sampling technique



(b) Extreme absolute imbalanced dataset with SWIM-SMOTE-TOMEK

4.2.3 SWIM-POLY Framework

The last combination sampling method is the SWIM-POLY framework, which is a combination of SWIM and POLYSMOTE. First, the data is oversampled with the SWIM oversampling method. A part, α, of the synthetic minority set d, as explained in Figure 4.2.1, is generated by the SWIM method. Afterwards, the POLYSMOTE sampling method rebalances the data completely. The hypothesis is that the combination of SWIM and POLYSMOTE may outperform the combination of SWIM-SMOTE(-TOMEK), as POLYSMOTE shows better results than SMOTE as a standalone sampling method. Using POLYSMOTE over SMOTE often results in a higher precision value, as the feature space of the minority class is smaller after sampling the data with SWIM compared to SMOTE.

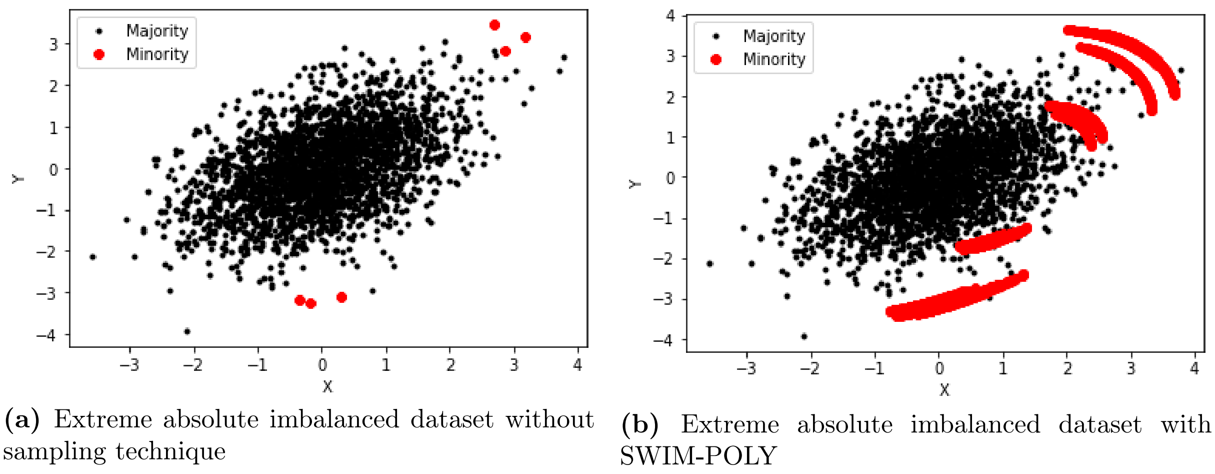
Algorithm 7 shows the framework of the SWIM-POLY sampling technique. In line 2, we calculate the number of minority instances to generate to rebalance the dataset completely. In line 3, we determine the number of instances to generate by the SWIM method, which is invoked in line 4. In line 5, we determine the number of instances to generate by the POLYSMOTE sampling technique. In line 6, we use the POLYSMOTE algorithm to generate the rebalanced majority and minority classes. The framework returns the oversampled minority class dataset.

Algorithm 7 SWIM-POLY Framework

```
1. Function SWIM-POLY($A$, $B$, $\gamma$, $\alpha$)
Input:
A, Majority class instances.
B, Minority class instances.
$\gamma$, spread of the generated synthetic minority instances along the density contours ($\gamma \in \mathbb{R}$)
$\alpha$, Ratio change from SWIM to POLYSMOTE ($\alpha = \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}$)
Output:
$B$; minority class with original and new synthetic minority instances
Method:
2. $d$ = |A| - |B|, number of minority samples to generate to rebalance the dataset completely.
3. $n = \lfloor \alpha \times d \rfloor$, number of instances generated by SWIM oversampling method
4. $B$ = SWIM($A$, $B$, $\gamma$, $n$), see Algorithm 2
5. $s = \lceil (1 - \alpha) \times d \rceil$, number of instances to generate by POLYSMOTE.
6. $B$ = POLYSMOTE($B$, $s$), see Algorithm 3
 Return $B$
```

Figure 4.2.4 displays the operations of the SWIM-POLY combination sampling method. As Figure 4.2.4b shows, new synthetic minority instances are generated on the outer density contours of the majority class due to the procedure of SWIM. These lines are almost equal to the minority class which are created by the standalone sampling method SWIM in Figure 4.1.3b. Meanwhile, SWIM-POLY generates instances for the minority class on other density contours which are closer the mean of the features of the minority dataset. This property is due to the calculation of the mean score B in the Algorithm 3. Therefore, the SWIM-POLY gets the display as in Figure 4.2.4b, which causes into a larger feature space of the minority class, which may improve the prediction performance of the minority class.

Figure 4.2.4 An illustration how SWIM-POLY creates new instances in a dataset



(a) Extreme absolute imbalanced dataset without sampling technique

(b) Extreme absolute imbalanced dataset with SWIM-POLY

4.3 Experimental Set-up

We have ten different datasets available, all with different IR values and number of observations in the majority class. For each dataset, we randomly down-sample the size of its minority class to 10 minority instances, to create the desired level of absolute imbalance in the data. For each dataset, we repeat this down-sampling for 2000 times which creates 2000 different sub-problems. We split the sub-problem into a train- and test set with equal proportions of the minority set. Thence, every train set contains five instances from the minority class, which is equal to the number of minority instances in the test set. Then, the data of the sub-problem is being standardized and missing values are removed from the dataset. Each single and combination sampling method is used to rebalance the data of the sub-problem. However, we only rebalance the data which is attributed to the train data. The test set is not adapted by any sampling method as the test set has to capture the features and properties of the reality.

In this research we use the eXtreme Gradient Boosting (XGBoost) classifier to create binary classification models for each sub-problem of each dataset, as XGBoost is very well suited to build up a strong classification model, as explained by Dhaliwal et al. (2018). For every XGBoost model we set the values of the hyperparameters to the default options to create a fair comparisons between different sampling methods. Therefore, we do not create a validation set along the train and test set. Further, we do not make use of a feature selection of the variables, also to create the comparison among the different datasets as fair as possible.

For each sub-problem, we calculate the evaluation measures. Afterwards, we take the average of these measures for the 2000 sub-problems for every dataset. By repeating down-sampling the minority class for 2000 times, we ensure the estimations of the evaluation criteria to be accurate given the potential for large variances caused by the small number of instances in the minority training sets. For all the the methods in this research, we set their parameters equal to their default options.

4.4 Evaluation Criteria

Evaluating the performances of classifiers which handle extreme imbalanced datasets is challenging. Due to the difference in group sizes of the minority and majority classes, the outcomes of some well-known evaluation criteria are often biased. For example, the *accuracy* measure, which is the fraction of well classified instances from both the majority and minority class over the number of observations in the test set. However, as the number of instances belonging to the majority class in the test set is way larger than the number of minority instances, the *accuracy* measure becomes biased towards the classifications of the instances belonging to the majority class. On top of that, the samples in the majority class are often well classified in extreme imbalanced datasets, which enhances the bias towards the majority class. This makes that the *accuracy* measure is not a robust and reliable metric for classifying extreme imbalanced datasets, which makes the measure inappropriate for this research.

In binary classification, we can define the prediction outcomes into four different groups: True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). In this research we use two classes, the majority and minority class. Therefore, we identify the instances from the majority class as *Negative* and from the minority class as *Positive*. If a predicted instance is labeled as TP, the instance is predicted well in the minority class. If it is labeled as FP, than the instance is predicted as a minority instance, but incorrectly, which means that the instance is actually a majority instance. TN means that the instance is labeled from the majority class, which is well predicted. If a test instance is labeled as FN, it is predicted as a majority instance, but is actually a minority instance. An overview of the four different labels is given in Table 4.4.1.

Table 4.4.1: Confusion matrix with the four different labels

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

To overcome the shortcomings of biased metrics like *accuracy*, we have to use an evaluation criteria which is able to handle imbalanced datasets. Baldi et al. (2000) explored a new evaluation criteria, the Matthews Correlation Coefficient (MCC), which is a robust metric able to handle imbalanced datasets, which scores between [-1, 1]. Chicco and Jurman (2020) showed the argumentation of using the MCC over other evaluation criteria, such as *accuracy* and F_1, by handling extreme imbalanced datasets. The MCC metric scores well only if the majority of the instances belonging to the minority class and the instances belonging to the majority class are well classified. As the majority of the minority instances needs to be well classified, the outcomes of the metric become unaffected by unbalanced datasets. Therefore, the MCC is a more reliable statistical rate which produces a high score only if the prediction obtained good results for the majority of both the minority and majority classes.

In this research we use the Matthews Correlation Coefficient as the main evaluation criteria to make a comparison between the outcomes of the sampling methods, and create insights which sampling method shows the best performances. Further, we use the precision and recall metric to gain insight in the number of well classified majority and minority instances, respectively. These two metrics are used to specify and support the findings of the MCC metric outcomes. The recall metric measures the fraction of well defined positive instances over the total number of positive instances. See Equation 3 for the formula of the recall measure.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

As the number of false negatives increases, the recall measure decreases. To get a high value of the recall, the false negatives should be as small as possible and the true positives as high as possible. In imbalanced datasets, the recall measure is of high interest as it is important to obtain as much as possible well classified instances from the minority class. The precision metric measures the fraction of the well classified positive instances over the total number of instances which are classified as positive. The precision equation is given in Equation 4.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

The precision measure uses the false positive rate. If the FP increases, the precision measure decreases. In a well performing classifier, both the recall and the precision measure is as high as possible. For datasets which are imbalanced, the level of the recall and precision metrics are often a trade-off. The higher the recall, the lower the precision value. Both criteria are used in this research to support the findings of the main metric, the MCC. This metric scores high only if the classifier can predict the majority of the positive data and the majority of the negative data correctly. In classification of extreme imbalanced datasets, it is important to classify the majority of both the positives and the negatives correctly, which makes the MCC metric appropriate in this research. Equation 5 shows the calculation of the MCC score.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}} \tag{5}$$

Chicco and Jurman (2020) explain that the MCC measure is a more robust, useful, reliable and truthful measure over the *accuracy* and F_1 metrics, as these can show overoptimistic inflated results especially on imbalanced datasets. The MCC metric scores between -1 and +1, where the extreme value -1 is obtained for classifiers which do not perform well and contain a lot of misclassifications, and +1 for classifiers which perform well with perfect classifications. Meanwhile, the shortcoming of the MCC metric is that there might be a change that the MCC is undefined, as a row or column in the confusion matrix is equal to zero. If that occurs, the MCC is not defined, and we substitute the zero values with an arbitrary ϵ, which has a value close to zero. In this case, the MCC value gives a value close to zero, which is equal to a MCC of a coin tossing classifier.

5 Results

In this section we provide the results and the consequential insights of this research. First, we provide the metric values for the different sampling techniques for all the different datasets. Secondly, based on these preliminary results, we give more insights which sampling method has the most promising effectiveness for classifying extreme absolute imbalanced datasets.

5.1 Metric Performance

In this subsection we show the results of the different sampling methods for the different datasets. We explore 16 different sampling methods for every different dataset plus the option without applying any sampling method, which is referred as *None*. We provide ten different tables to give the metric scores precision, recall and MCC for all the different sampling methods. Each score is an average value of the 2000 different sub-problems, except for the precision metric scores. That is because to calculate the precision metric, we divide the number of TP by TP + FP. Around 10-40% of the 2000 sub-problems, TP + FP becomes equal to zero, as the classifier predicts all the instances in the test data as negatives (majority data). As we can not divide by zero, we do not take these iterations into account to calculate the precision metric scores.

Table 5.1.1: Results for Abalone dataset

D1 *Abalone*	Precision*	Recall	MCC
Single methods			
None	0.754	0.055	0.095
SWIM	0.472	0.111	0.142
SMOTE	0.172	0.232	**0.158**
TOMEK	0.739	0.054	0.092
POLYSMOTE	0.507	0.084	0.112
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.444	0.114	0.138
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.457	0.120	0.147
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.455	0.115	0.143
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.460	0.112	0.139
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.442	0.117	0.141
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.459	0.112	0.141
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.458	0.117	0.143
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.458	0.112	0.140
SWIM-POLY $\left(\frac{1}{5}\right)$	0.463	0.100	0.125
SWIM-POLY $\left(\frac{2}{5}\right)$	0.464	0.096	0.120
SWIM-POLY $\left(\frac{3}{5}\right)$	0.452	0.099	0.122
SWIM-POLY $\left(\frac{4}{5}\right)$	0.454	0.104	0.132

*Precision metric obtained by less iterations due to undefined values in precision derivation.

We explore and analyze the results of the created combination sampling methods, namely SWIM-SMOTE for $\alpha = \{\frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}\}$, SWIM-SMOTE-TOMEK for $\alpha = \{\frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}\}$, and SWIM-POLY for $\alpha = \{\frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}\}$, and compare its results with the outcomes of the single sampling methods, which are: None, SMOTE, SWIM, POLYSMOTE and TOMEK.

Table 5.1.1 shows the results of the dataset *Abalone*. The table gives an overview of the different sampling methods. At first, the scores of the single sampling methods are presented, and then the outcomes of the created combination sampling methods. The highest precision value is obtained by using no sampling method, which is as expected as there is a bias towards the majority class if no sampling method is used. As there is such a bias towards the majority instances, the precision metric shows high values. Obviously, the recall metric with no sampling method has a low value of 0.055, as the highest value of the recall metric is 0.232, which is obtained with the SMOTE sampling technique. The highest MCC value is obtained with the sampling method SMOTE with a score of 0.158. As all the combination methods show less MCC results compared to SMOTE, we expect that the *Abalone* dataset contains a single sub-space of the minority dataset. If this minority set is constructed out of one set, the SMOTE sampling technique is outperforming the SWIM sampling methods due to the less sharper boundaries of the minority dataset after sampling with SMOTE.

Table 5.1.2: Results for Car dataset

D2 *Car*	Precision*	Recall	MCC
Single methods			
None	0.880	0.049	0.094
SWIM	0.734	0.166	**0.253**
SMOTE	0.588	0.158	0.216
TOMEK	0.864	0.049	0.093
POLYSMOTE	0.811	0.055	0.098
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.760	0.154	0.243
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.746	0.160	0.247
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.744	0.159	0.246
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.729	0.156	0.240
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.739	0.147	0.230
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.734	0.158	0.244
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.737	0.162	0.251
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.740	0.158	0.245
SWIM-POLY $\left(\frac{1}{5}\right)$	0.754	0.102	0.169
SWIM-POLY $\left(\frac{2}{5}\right)$	0.753	0.115	0.187
SWIM-POLY $\left(\frac{3}{5}\right)$	0.763	0.114	0.186
SWIM-POLY $\left(\frac{4}{5}\right)$	0.726	0.125	0.197

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Table 5.1.2 shows the scores of the dataset *Car*. The highest precision value is obtained without any sampling method, which is as expected. Other well performing sampling methods creating a high precision value are TOMEK and POLYSMOTE. The highest recall score is obtained with the SWIM method, namely with a score of 0.166. The highest MCC score is obtained with the SWIM oversampling method with a score of 0.253.

What stands out is that the recall score of POLYSMOTE is low compared to other sampling methods. Evidently, POLYSMOTE does not work properly on this dataset. Therefore, this can declare the scores of SWIM-POLY which are lower than other sampling methods. Further, we do not observe much differences in MCC scores between SWIM-SMOTE and SWIM-SMOTE-TOMEK.

Table 5.1.3: Results for Page dataset

D3 *Page*	Precision*	Recall	MCC
Single methods			
None	0.836	0.072	0.127
SWIM	0.240	0.309	0.238
SMOTE	0.285	0.247	0.216
TOMEK	0.810	0.084	0.147
POLYSMOTE	0.401	0.202	0.220
Combination methods			
SWIM-SMOTE $(\frac{1}{5})$	0.260	0.295	0.239
SWIM-SMOTE $(\frac{2}{5})$	0.261	0.297	0.239
SWIM-SMOTE $(\frac{3}{5})$	0.256	0.309	0.245
SWIM-SMOTE $(\frac{4}{5})$	0.255	0.300	0.239
SWIM-SMOTE-TOMEK $(\frac{1}{5})$	0.267	0.300	0.244
SWIM-SMOTE-TOMEK $(\frac{2}{5})$	0.261	0.293	0.238
SWIM-SMOTE-TOMEK $(\frac{3}{5})$	0.248	0.291	0.231
SWIM-SMOTE-TOMEK $(\frac{4}{5})$	0.249	0.302	0.237
SWIM-POLY $(\frac{1}{5})$	0.273	0.293	0.240
SWIM-POLY $(\frac{2}{5})$	0.264	0.307	**0.246**
SWIM-POLY $(\frac{3}{5})$	0.260	0.301	0.240
SWIM-POLY $(\frac{4}{5})$	0.241	0.314	0.235

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Table 5.1.3 shows the results of the *Page* dataset. Again, the highest precision metric value is obtained if the data is not treated with any sampling method, with a score of 0.836. If the data is rebalanced with an oversampling method, the precision score drops about more than 0.5. The recall score without sampling method is 0.072, which is a low score. The sampling methods may increase the recall in such a way that the reduction of the precision score is less important. The highest recall value is obtained with the SWIM-POLY$(\frac{4}{5})$ combination method, with a score of 0.314, which is way larger than the score of 0.072, without sampling method. The highest MCC metric score is obtained with the SWIM-POLY($\frac{2}{5}$) combination oversampling method. This is due to the relative high precision score of the POLYSMOTE sampling method. By increasing the value of α we observe a decrease in precision scores for all the combination methods, as a larger part of the synthetic instances are generated by the SWIM sampling method instead of POLYSMOTE. Therefore, SWIM-POLY($\frac{2}{5}$) has the highest MCC score, as it is combination of a high recall and precision score, even they are not the highest. The MCC score is 3.4% higher compared to the highest MCC score of the single sampling method SWIM.

Table 5.1.4: Results for Penbased dataset

D4 *Penbased*	Precision*	Recall	MCC
Single methods			
None	0.943	0.115	0.215
SWIM	0.604	0.601	0.578
SMOTE	0.666	0.345	0.422
TOMEK	0.931	0.118	0.214
POLYSMOTE	0.691	0.329	0.420
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.620	0.490	0.514
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.612	0.539	0.543
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.621	0.556	0.559
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.608	0.555	0.554
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.634	0.489	0.523
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.617	0.540	0.550
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.612	0.554	0.552
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.618	0.568	0.565
SWIM-POLY $\left(\frac{1}{5}\right)$	0.678	0.526	0.567
SWIM-POLY $\left(\frac{2}{5}\right)$	0.666	0.559	0.582
SWIM-POLY $\left(\frac{3}{5}\right)$	0.657	0.573	0.588
SWIM-POLY $\left(\frac{4}{5}\right)$	0.633	0.599	**0.592**

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Table 5.1.4 gives insights in the performances of the sampling methods on the dataset *Penbased*. The highest precision score is obtained by using no sampling method, with a score of 0.943. The second highest precision score is obtained with TOMEK, with a score of 0.931. Evidently, TOMEK removes a few majority instances from the training dataset, which increases the recall score compared with the recall score without using a sampling method. The highest recall score is obtained with the SWIM oversampling method, with a score of 0.601. The highest MCC score is obtained with SWIM-POLY($\frac{4}{5}$) with a score of 0.592, which is an increase of 2.4% relative to the highest single MCC score obtained by the SWIM method. The SWIM-SMOTE-TOMEK sampling methods generally creates higher MCC scores than the combination method SWIM-SMOTE. So, for the dataset *penbased* the TOMEK undersampling method is a good addition to the oversampling method SWIM-SMOTE.

Table 5.1.5 shows the results of the dataset *Shuttle*. The highest precision metric value is 0.998 which is obtained without using a sampling method and with the undersampling method TOMEK. This score is almost equal to a value of one, so almost all classified minority instance are well classified as they belong to the minority class. The recall score without using a sampling method is 0.441, which is a relative high score. Therefore, the dataset *Shuttle* might have two disjoint classes, namely of the majority and minority classes. It suggests that there are no Tomek Links in the data, but this not the case as the recall score is not equal to the recall score without using a sampling method. However, both recall scores are almost equal to each other. The highest recall score is obtained with the oversampling method SWIM, with a score of 0.908. The highest MCC score is 0.904, which is obtained with the SWIM-POLY$\left(\frac{2}{5}\right)$ oversampling combination. The MCC score is 1.5% higher than the largest MCC score of the single sampling methods, namely SWIM. What stands out is that adding the undersampling TOMEK to the combination method SWIM-SMOTE does not increase the MCC neither the precision scores.

Table 5.1.5: Results for Shuttle dataset

D5 *Shuttle*	Precision*	Recall	MCC
Single methods			
None	0.998	0.441	0.602
SWIM	0.895	0.908	0.891
SMOTE	0.992	0.734	0.829
TOMEK	0.998	0.449	0.611
POLYSMOTE	0.984	0.475	0.640
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.949	0.881	0.903
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.933	0.891	0.901
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.925	0.900	0.900
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.919	0.907	0.903
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.940	0.890	0.903
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.929	0.895	0.901
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.919	0.897	0.897
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.914	0.905	0.900
SWIM-POLY $\left(\frac{1}{5}\right)$	0.969	0.861	0.903
SWIM-POLY $\left(\frac{2}{5}\right)$	0.952	0.877	**0.904**
SWIM-POLY $\left(\frac{3}{5}\right)$	0.945	0.880	0.902
SWIM-POLY $\left(\frac{4}{5}\right)$	0.932	0.884	0.896

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Next, Table 5.1.6 shows the result for the *Spambase* dataset. The highest precision value is obtained without using any sampling method, with a score of 0.729. The second-best and third-best sampling method regarding the precision score is TOMEK and POLYSMOTE, respectively. The highest recall value is obtained by the SMOTE oversampling method, with a score of 0.190, which is a third multiple of the recall score without using a sampling method. A well performing sampling method with regard to the recall measure is the SWIM-POLY$\left(\frac{3}{5}\right)$ method.

Due to the high recall score and the well performing of the POLYSMOTE, we observe the highest MCC score by the combination method SWIM-POLY($\frac{3}{5}$) with a score of 0.235. This score is an increase of 8.3% compared to the highest MCC score of the single sampling method, which is SWIM with a score of 0.217. Again, adding TOMEK to the SWIM-SMOTE sampling method does not influence the results positively.

Table 5.1.6: Results for Spambase dataset

D6 *Spambase*	Precision*	Recall	MCC
Single methods			
None	0.729	0.057	0.097
SWIM	0.492	0.174	0.217
SMOTE	0.398	0.190	0.209
TOMEK	0.687	0.066	0.107
POLYSMOTE	0.602	0.132	0.191
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.506	0.158	0.201
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.488	0.161	0.201
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.505	0.163	0.209
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.495	0.167	0.207
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.503	0.157	0.201
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.490	0.164	0.205
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.498	0.167	0.209
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.487	0.161	0.199
SWIM-POLY $\left(\frac{1}{5}\right)$	0.516	0.172	0.222
SWIM-POLY $\left(\frac{2}{5}\right)$	0.499	0.178	0.225
SWIM-POLY $\left(\frac{3}{5}\right)$	0.517	0.185	**0.235**
SWIM-POLY $\left(\frac{4}{5}\right)$	0.505	0.174	0.219

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Table 5.1.7 gives insights in the performances of the sampling methods applied to the *Thyroid* dataset. The highest precision metric value is 0.570, which is obtained by treating the imbalanced data without any sampling method. The second-best precision metric value is obtained with the undersampling method TOMEK, which is as expected. The highest recall measure is obtained for the SWIM method, with a score of 0.563. Meanwhile, the recall score of SWIM is almost equal to the recall score of SMOTE.

Further, the highest MCC value is obtained with the SWIM single oversampling method, with a score of 0.499. Noticeable, the single POLYSMOTE sampling method shows a low precision score, which may declare the low metric values for the SWIM-POLY sampling method. Table 5.1.7 shows that the SWIM-SMOTE sampling method outcomes come very close to the results of single SWIM sampling method. Evidently, the dataset can be rebalanced best with the single SWIM method.

Table 5.1.7: Results for Thyroid dataset

D7 *Thyroid*	Precision*	Recall	MCC
Single methods			
None	0.570	0.231	0.291
SWIM	0.500	0.563	**0.499**
SMOTE	0.424	0.561	0.463
TOMEK	0.523	0.269	0.306
POLYSMOTE	0.435	0.356	0.344
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.491	0.517	0.468
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.505	0.546	0.491
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.501	0.555	0.494
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.505	0.559	0.498
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.496	0.518	0.473
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.498	0.537	0.481
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.494	0.545	0.484
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.506	0.551	0.492
SWIM-POLY $\left(\frac{1}{5}\right)$	0.488	0.448	0.430
SWIM-POLY $\left(\frac{2}{5}\right)$	0.491	0.472	0.448
SWIM-POLY $\left(\frac{3}{5}\right)$	0.509	0.501	0.473
SWIM-POLY $\left(\frac{4}{5}\right)$	0.498	0.538	0.486

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Table 5.1.8 shows the results of the *Vehicle* dataset. The highest precision metric value is obtained without any sampling method, with a score of 0.968. The precision score of TOMEK is almost equal with a score of 0.963. The highest recall metric value is obtained by the single SWIM sampling method, with a score of 0.535. The highest MCC score is obtained with the SWIM-POLY$\left(\frac{3}{5}\right)$ combination sampling method. The combination sampling method shows an increase of 6.1% compared to the highest MCC score of the single sampling method SWIM.

Further, the Table shows that the SWIM-SMOTE sampling method shows higher MCC scores for every value of α compared to the corresponding SWIM-SMOTE-TOMEK sampling method. Moreover, this also applies for the precision and recall scores. So, the results of this dataset show that adding the undersampling method TOMEK to the oversampling combination SWIM-SMOTE does not improve prediction performances.

Table 5.1.8: Results for Vehicle dataset

D8 *Vehicle*	Precision*	Recall	MCC
Single methods			
None	0.968	0.134	0.244
SWIM	0.484	0.535	0.475
SMOTE	0.643	0.362	0.421
TOMEK	0.963	0.141	0.256
POLYSMOTE	0.764	0.306	0.414
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.592	0.462	0.479
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.562	0.495	0.488
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.542	0.523	0.498
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.534	0.526	0.497
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.585	0.449	0.470
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.551	0.482	0.476
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.541	0.501	0.481
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.524	0.528	0.493
SWIM-POLY $\left(\frac{1}{5}\right)$	0.597	0.461	0.483
SWIM-POLY $\left(\frac{2}{5}\right)$	0.571	0.507	0.502
SWIM-POLY $\left(\frac{3}{5}\right)$	0.554	0.522	**0.504**
SWIM-POLY $\left(\frac{4}{5}\right)$	0.531	0.529	0.494

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Table 5.1.9 displays the results of the dataset $Yeast$. The highest precision value is observed by using no sampling technique, which is as expected as there is a bias towards the majority class without the use of a sampling method. As there is such a bias towards the majority instances, the precision metric shows high values. Obviously, the recall metric with no sampling method has a low value. The highest recall score is 0.398 obtained by the SWIM-POLY$\left(\frac{4}{5}\right)$ sampling method. The MCC value for no sampling method is 0.234. The sampling method with the highest MCC value is for the combination SWIM-POLY$\left(\frac{4}{5}\right)$ oversampling method with value 0.413.

The highest MCC value for the single methods is 0.369 for both the SWIM as for the POLYSMOTE sampling method. The combination method shows an increase of 11.9% for the MCC metric value compared to the highest MCC value of the single sampling methods. As we can see from the $Yeast$ dataset, the combination sampling method SWIM-POLY$\left(\frac{4}{5}\right)$ shows the best performance, compared to the single methods and the SWIM-SMOTE and the SWIM-SMOTE-TOMEK frameworks.

Table 5.1.9: Results for Yeast dataset

D9 *Yeast*	Precision*	Recall	MCC
Single methods			
None	0.650	0.161	0.234
SWIM	0.505	0.343	0.369
SMOTE	0.433	0.390	0.371
TOMEK	0.611	0.174	0.240
POLYSMOTE	0.516	0.339	0.369
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.522	0.335	0.367
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.513	0.341	0.369
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.514	0.328	0.361
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.507	0.335	0.364
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.517	0.317	0.354
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.511	0.334	0.365
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.510	0.344	0.375
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.511	0.343	0.373
SWIM-POLY $\left(\frac{1}{5}\right)$	0.522	0.369	0.396
SWIM-POLY $\left(\frac{2}{5}\right)$	0.513	0.386	0.407
SWIM-POLY $\left(\frac{3}{5}\right)$	0.527	0.388	0.412
SWIM-POLY $\left(\frac{4}{5}\right)$	0.515	0.398	**0.413**

*Precision metric obtained by less iterations due to undefined values in precision derivation.

Lastly, Table 5.1.10 shows the results of the International Financial Institution dataset. This dataset is the only dataset in this research which contains real life applications, and is not a known and often used dataset from a repository. As can be noted, the highest precision score is obtained with the undersampling technique Tomek Links, with a score of 0.905. The second-best option is without using any sampling method.

The recall scores are all almost around zero, so we can mention that the IFI dataset does not contain a lot of information to identify instances from the minority class. However, in this paper we do research between the scores of the different sampling techniques. Therefore, we still can interpret the results of this dataset, even if the results of the classifier model are not excellent. The highest recall score is 0.117, obtained by the SWIM-POLY($\frac{4}{5}$) sampling method. The highest MCC score is also obtained by SWIM-POLY($\frac{4}{5}$) with a score of 0.031, which is 10.7% higher than the highest MCC score of the single methods, which is 0.028 with the SWIM method. Therefore, we can mention that the combination methods can improve the results for the IFI dataset at most with 10.7% compared to the single methods.

The findings for this particular dataset may be improved by adding a feature selection process to pre-processing of the dataset, to remove irrelevant and redundant features. Further, a hyperparameter optimization may also improve the prediction performances of the classifier.

Table 5.1.10: Results for IFI dataset

D10 *IFI*	Precision*	Recall	MCC
Single methods			
None	0.548	0.000	0.000
SWIM	0.039	0.100	0.028
SMOTE	0.060	0.068	0.024
TOMEK	0.905	0.001	0.001
POLYSMOTE	0.079	0.050	0.021
Combination methods			
SWIM-SMOTE $\left(\frac{1}{5}\right)$	0.059	0.067	0.023
SWIM-SMOTE $\left(\frac{2}{5}\right)$	0.051	0.073	0.024
SWIM-SMOTE $\left(\frac{3}{5}\right)$	0.046	0.081	0.025
SWIM-SMOTE $\left(\frac{4}{5}\right)$	0.046	0.082	0.024
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	0.062	0.060	0.022
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	0.051	0.076	0.025
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	0.049	0.079	0.024
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	0.046	0.089	0.027
SWIM-POLY $\left(\frac{1}{5}\right)$	0.042	0.103	0.030
SWIM-POLY $\left(\frac{2}{5}\right)$	0.039	0.105	0.029
SWIM-POLY $\left(\frac{3}{5}\right)$	0.039	0.105	0.029
SWIM-POLY $\left(\frac{4}{5}\right)$	0.037	0.117	**0.031**

*Precision metric obtained by less iterations due to undefined values in precision derivation.

5.2 Insights and Recommendations

To give insight which method gives the most promising performances, we count the number of times each method has the highest score for each metric. Therefore, Table 5.2.1 shows that the best performing sampling method, to create a as high as possible precision value is by using no sampling method. In 8 times out of 10 datasets, the precision metric score is the highest without using any sampling method. The second-best option is to use the undersampling method TOMEK, which is as expected, as TOMEK is an undersampling technique, which does not generate new minority instances. Thence, the data is still imbalanced after resampling the data with TOMEK, which maintains the bias towards the majority class. Due to this bias, the classifier predicts too many instances belonging to the majority class. As the precision value of the sampling method *None* is equal to the precision value of the TOMEK sampling method in the dataset *Shuttle*, the number of winners in the precision column adds up to 11.

The best performing sampling method regarding the recall metric is SWIM. 5 times out of 10 datasets, the SWIM method creates the highest recall score. The second-best option is, 3 out of 10 times, the SWIM-POLY sampling method, which obviously uses the SWIM method which may declare the well performing SWIM-POLY sampling method. The third-best option to create a high recall score is the SMOTE sampling method. Apparently, the SWIM sampling method outperforms the single SMOTE method and combinations with SMOTE, regarding the recall metric. Moreover, adding POLYSMOTE to the SWIM sampling method has a positive influence on the level of the recall measure, which does not apply for adding SMOTE to SWIM.

Table 5.2.1: Number of times best sampling method by all metric scores

	Precision	Recall	MCC
Single methods			
None	9	0	0
SWIM	0	5	2
SMOTE	0	2	1
TOMEK	2	0	0
POLYSMOTE	0	0	0
Combination methods			
SWIM-SMOTE	0	0	0
SWIM-SMOTE-TOMEK	0	0	0
SWIM-POLY	0	3	7

Table 5.2.1 shows that the SWIM-POLY sampling method has 7 out of 10 times the highest MCC score. The second-best option is the SWIM sampling method, and third-best is the SMOTE oversampling method. As the MCC metric is the main metric in this research, as it captures the prediction performances of both classes, we have more interest in the outcomes of the MCC metric. The combination methods SWIM-SMOTE and SWIM-SMOTE-TOMEK do not score well, however SWIM-POLY does. Therefore, the intuition is that the POLYSMOTE has some strong and positively impact on the performance of the SWIM-POLY sampling method. So, we are interested in why SWIM-POLY works so well, over the other sampling methods SWIM-SMOTE and SWIM-SMOTE-TOMEK.

To receive more information about the different sampling methods, even for each value of α, we create a rank table. With such a rank table, we like to create more insights in the performance of the SWIM-POLY sampling method and which value of α shows the most promising effects. To create such a rank table, we ranked the scores for each dataset and for each evaluation criteria. Then, we took the average value of its ranks for each sampling method. Lastly, we made a rank out of these average scores for each metric. Table 5.2.2 displays this rank table, for all the three evaluation criteria, with *Av. Score* as the average score per sampling method, and *Rank* as the rank score of the average scores.

Table 5.2.2 shows that the number one well performing sampling method to create a high as possible precision value is using no sampling method, which is as expected and in line with the outcomes in Table 5.2.1. The second-best sampling method with the highest precision values is the TOMEK undersampling technique, which is not surprising as TOMEK is the only undersampling method, which often goes together with high precision scores. What stands out is the sampling method on the third place, which is the oversampling method POLYSMOTE.

Apparently, POLYSMOTE scores well with high precision scores over all the ten datasets. As we consider the precision rank outcome of SMOTE, we see that SMOTE is on the 11th place, which means that the sampling method show very low precision scores over all the datasets. This rank of SMOTE is way higher than the rank score of POLYSMOTE. Other well performing sampling methods regarding the precision metric are the SWIM-POLY oversampling methods. We observe that the higher the value for α, the higher the rank score for the precision metric. This is due to the rank score of SWIM, which is on place 17. The larger contribution of SWIM in a combination sampling method, the higher the rank becomes of such a combination sampling method.

However, the most well performing sampling method regarding the recall metric is SWIM. Moreover, we observe that the recall rank of the single SMOTE sampling method is on the 10th place, which is expected as SMOTE is not able to handle proper in extreme absolute imbalanced datasets. Further, we notice that the rank score increases if the value for α decreases. This is due to the larger part SWIM contributes in the combination sampling methods. If a large part of a combination sampling method is constructed with the sampling method SWIM, the recall score increases.

Table 5.2.2: Rank scores for each evaluation criteria per sampling technique

	Precision		Recall		MCC	
	Av. Score	Rank	Av. Score	Rank	Av. Score	Rank
Single methods						
None	1.1	**1**	16.9	17	16.7	17
SWIM	13.9	17	3.5	**1**	6.6	5
SMOTE	10.7	11	7.9	10	10.4	14
TOMEK	1.9	2	16.1	16	16.3	16
POLYSMOTE	4.8	3	14.5	15	14.3	15
Combination methods						
SWIM-SMOTE $\left(\frac{1}{5}\right)$	8.5	7	10.7	13	9.8	12
SWIM-SMOTE $\left(\frac{2}{5}\right)$	10.2	9	7.7	9	7.2	8
SWIM-SMOTE $\left(\frac{3}{5}\right)$	10.2	10	6.0	3	5.9	3
SWIM-SMOTE $\left(\frac{4}{5}\right)$	12.4	16	6.4	5	7.1	7
SWIM-SMOTE-TOMEK $\left(\frac{1}{5}\right)$	8.7	8	10.9	14	10.1	13
SWIM-SMOTE-TOMEK $\left(\frac{2}{5}\right)$	10.8	12	9.2	11	8.8	11
SWIM-SMOTE-TOMEK $\left(\frac{3}{5}\right)$	12.3	15	7.0	6	8.0	10
SWIM-SMOTE-TOMEK $\left(\frac{4}{5}\right)$	12.1	13	6.0	4	7.6	9
SWIM-POLY $\left(\frac{1}{5}\right)$	6.8	4	10.3	12	6.9	6
SWIM-POLY $\left(\frac{2}{5}\right)$	8.5	6	7.7	8	5.5	2
SWIM-POLY $\left(\frac{3}{5}\right)$	7.9	5	7.1	7	5.4	**1**
SWIM-POLY $\left(\frac{4}{5}\right)$	12.2	14	5.1	2	6.4	4

Table 5.2.2 shows that the sampling method with the lowest average rank is SWIM-POLY with α equal to $\frac{3}{5}$ with a score of 5.4. We notice that the SWIM-POLY$\left(\frac{3}{5}\right)$ is a combination of a well performing POLYSMOTE sampling method and SWIM sampling method. The second-best sampling method is SWIM-POLY $\left(\frac{2}{5}\right)$. In fact, the SWIM-POLY combination sampling methods for all the values of α are in the top 6 of the rank table, which is in line with the findings of Table

5.2.1. Further, we observe that expanding the SWIM-SMOTE framework with the undersampling technique TOMEK has negative influence on the MCC outcomes. Each rank score of a SWIM-SMOTE-TOMEK sampling method is higher than for its corresponding SWIM-SMOTE sampling method with the same value for α. The sampling method SWIM-SMOTE $\left(\frac{3}{5}\right)$ is on the third rank, which is a high score, however as SWIM-SMOTE was zero times the best sampling method, as can be seen in 5.2.1, the sampling method can not be seen as a good technique to handle extreme absolute imbalanced datasets. So, from Table 5.2.2, we observe that the sampling method SWIM-POLY is the best performing sampling method to handle extreme absolute imbalanced datasets. Moreover, we have no indication that the performance of the SWIM-POLY sampling method is dependent on the number of attributes or the level of IR in the data, except that the IR belongs to a dataset which is qualified as extreme imbalanced. We recommend to use the combination sampling method SWIM-POLY with α equal to $\frac{3}{5}$, as this sampling method is placed number one in the rank table, due to a good trade-off off a low precision and recall score. The higher the value for α the lower the precision scores, and the higher the recall scores. Therefore, we recommend to use an α equal to $\frac{3}{5}$ to handle extreme absolute imbalanced datasets.

6 Conclusion

In this section we provide the conclusion of this research. First, the answers of the three research questions are provided, then the extensions and future research are described.

6.1 Conclusions and Remarks

In this research we created new sampling methods to rebalance extreme absolute imbalanced datasets to improve the prediction performances of classifiers. We used several known sampling methods, such as SMOTE, SWIM, TOMEK and POLYSMOTE, which are combined into combination sampling methods to obtain higher evaluation criteria scores of the classifiers prediction. We combined SMOTE into the sampling methods SWIM-SMOTE and SWIM-SMOTE-TOMEK, and we joined the oversampling methods SWIM and POLYSMOTE into SWIM-POLY. For all the different combination sampling methods we used the parameter α, showing the fraction of the total number of synthetic instances to be generated with the SWIM sampling method.

The first sub-question in this research is: *Is it possible to outperform standalone sampling methods by creating a framework which is a combination of existing single sampling techniques for extreme absolute imbalanced datasets?* We used ten different extreme absolute imbalanced datasets, all treated with 16 different sampling methods. We have observed in 7 out of 10 different datasets that the suggested combination sampling method SWIM-POLY outperforms the other sampling methods. To obtain more insights about this finding, we created a rank table, which showed that the SWIM-POLY method works well due to the combination of well performing single sampling methods SWIM and POLYSMOTE. Therefore, we conclude that we can outperform the single sampling methods for datasets which are classified as absolute extreme imbalanced by using the combination SWIM and POLYSMOTE into SWIM-POLY.

The second sub-question in this research is *How can we create a well performing sampling method that is a combination of over- and/or undersampling methods to handle datasets that are qualified as extreme absolute imbalanced?* We created a rank table, which showed that the SWIM-POLY sampling method with a value of α equal to $\frac{3}{5}$ shows the most promising results. For SWIM-SMOTE, SWIM-SMOTE-TOMEK, and SWIM-POLY, we observed that if the value for α increases, the precision scores decreases and the recall scores increases. This is due to the working of SWIM, which is a sampling method what goes with high recall and low precision scores. When the value of α increases, a larger part of the total number of to be generated synthetic minority instances are created by SWIM, which explains the findings of raising recall and decreasing precision scores. Due to this trade-off between these two metrics, we observe that the most promising value for α is equal to $\frac{3}{5}$. Further, we can conclude that the combination sampling methods SWIM-SMOTE and SWIM-SMOTE-TOMEK are not the most promising sampling methods for these typical datasets, however, the ranks of these methods are better than the rank of the single sampling method SMOTE, which is often used in the literature. Lastly, we can conclude that adding the undersampling technique TOMEK to the SWIM-SMOTE framework does not result into higher evaluation criteria scores.

The main question in this research is as follows: *What is the effect of rebalancing methods on extreme absolute imbalanced datasets?* To answer this question, we used ten different datasets treated with 16 different sampling methods. We used the XGBoost classifier to predict the instances in the test sets, which resulted into different MCC metric scores. To evaluate the effect of sampling methods, we calculated for each dataset the change of the MCC outcome of the untreated sampling method to the largest MCC outcome, except for the IFI dataset, as this percentage of improvement of more than 3000% can be seen as an outlier. The average MCC change is 105.6% improvement, which shows that the effect of rebalancing extreme absolute imbalanced datasets is positively significant. So rebalancing datasets with sampling techniques for datasets which are qualified as extreme absolute imbalanced can be recommended to improve the prediction performances of a classifier. In this research we recommend to use the SWIM-POLY $\left(\frac{3}{5}\right)$ sampling technique, as this method shows an improvement in MCC scores of 88.6% compared to the MCC scores without sampling the data.

6.2 Extensions and Future Research

An extension of the created sampling frameworks SWIM-SMOTE, SWIM-SMOTE-TOMEK and SWIM-POLY is adding another (single) sampling method to the framework. An example is adding an undersampling technique to the SWIM-POLY framework, like TOMEK, RUS, DBMUTE or ENN. Further, other SMOTE variant techniques such as ADASYN, Borderline-SMOTE or Supervised-SMOTE, described in Kovács (2019), can be used instead of the single sampling methods SMOTE. Hereby, new frameworks are created by combining new single sampling techniques. Another option is to use the algorithmic methods to add to the frameworks, as described in Section 2.2.

Moreover, more insights in the SWIM-POLY sampling method can be created by using additional datasets. These datasets must have different characteristics, such as a larger size of the majority class, different sizes of the number of instances in the minority class, and different number of attributes belonging to the data. Lastly, we can zoom in the POLYSMOTE sampling method, and especially which topology works best for extreme absolute imbalanced datasets. In this research, we used the *star* topology, as Gazzah and Amara (2008) researched that the *star* topology performs best over the other ontology's.

Another point of future research is using datasets which are already classified as extreme absolute imbalanced. The shortcoming of this research is the down-sampling of the minority class of the used datasets, as there are no datasets available which comply with the requirements to classify the data as extreme absolute imbalanced.

7 References

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., and Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424.

Bellinger, C., Sharma, S., Japkowicz, N., and Zaïane, O. R. (2020). Framework for extreme imbalance classification: Swim—sampling with the majority class. *Knowledge and Information Systems*, 62(3):841–866.

Bunkhumpornpat, C. and Sinapiromsaran, K. (2017). Dbmute: density-based majority under-sampling technique. *Knowledge and Information Systems*, 50(3):827–850.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002a). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002b). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Chawla, N. V., Lazarevic, A., Hall, L. O., and Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, pages 107–119. Springer.

Cheng, F., Zhang, J., and Wen, C. (2016). Cost-sensitive large margin distribution machine for classification of imbalanced data. *Pattern Recognition Letters*, 80:107–112.

Chicco, D. and Jurman, G. (2020). The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13.

Collell, G., Prelec, D., and Patil, K. R. (2018). A simple plug-in bagging ensemble based on threshold-moving for classifying binary and multiclass imbalanced data. *Neurocomputing*, 275:330–340.

Dhaliwal, S. S., Nahid, A.-A., and Abbas, R. (2018). Effective intrusion detection system using xgboost. *Information*, 9(7):149.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd.

Fernández, A., García, S., del Jesus, M. J., and Herrera, F. (2008). A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, 159(18):2378–2398.

Fernández, A., Garcia, S., Herrera, F., and Chawla, N. V. (2018). Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905.

Gazzah, S. and Amara, N. E. B. (2008). New oversampling approaches based on polynomial fitting for imbalanced data sets. In *2008 the eighth iapr international workshop on document analysis systems*, pages 677–684. IEEE.

Haixiang, G., Yijing, L., Yanan, L., Xiao, L., and Jinling, L. (2016). Bpso-adaboost-knn ensemble learning algorithm for multi-class imbalanced data classification. *Engineering Applications of Artificial Intelligence*, 49:176–193.

Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer.

He, H., Bai, Y., Garcia, E. A., and Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE.

Jian, C., Gao, J., and Ao, Y. (2016). A new sampling method for classifying imbalanced data based on support vector machine ensemble. *Neurocomputing*, 193:115–122.

Kovács, G. (2019). An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, 83:105662.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232.

Randhawa, K., Loo, C. K., Seera, M., Lim, C. P., and Nandi, A. K. (2018). Credit card fraud detection using adaboost and majority voting. *IEEE access*, 6:14277–14284.

Sharma, S., Bellinger, C., Krawczyk, B., Zaiane, O., and Japkowicz, N. (2018). Synthetic oversampling with the majority class: A new perspective on handling extreme imbalance. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 447–456. IEEE.

Solberg, A. S. and Solberg, R. (1996). A large-scale evaluation of features for automatic detection of oil spills in ers sar images. In *IGARSS'96. 1996 International Geoscience and Remote Sensing Symposium*, volume 3, pages 1484–1486. IEEE.

Sun, Z., Song, Q., Zhu, X., Sun, H., Xu, B., and Zhou, Y. (2015). A novel ensemble method for classifying imbalanced data. *Pattern Recognition*, 48(5):1623–1637.

Tomek, I. et al. (1976). An experiment with the edited nearest-nieghbor rule.

Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3):408–421.

Yijing, L., Haixiang, G., Xiao, L., Yanan, L., and Jinling, L. (2016). Adapted ensemble classification algorithm based on multiple classifier system and feature selection for classifying multi-class imbalanced data. *Knowledge-Based Systems*, 94:88–104.

Yin, L., Ge, Y., Xiao, K., Wang, X., and Quan, X. (2013). Feature selection for high-dimensional imbalanced data. *Neurocomputing*, 105:3–11.

Zeng, M., Zou, B., Wei, F., Liu, X., and Wang, L. (2016). Effective prediction of three common diseases by combining smote with tomek links technique for imbalanced medical data. In *2016*

IEEE International Conference of Online Analysis and Computing Science (ICOACS), pages 225–228. IEEE.