Small Data Over-sampling
Improving small data prediction accuracy using the Geometric SMOTE algorithm

Georgios Douzas[1], Fernando Bacao[1], Maria Lechleitner[1]*

[1]NOVA Information Management School, Universidade Nova de Lisboa

*Corresponding Author

Postal Address: NOVA Information Management School, Campus de Campolide, 1070-312 Lisboa, Portugal

Telephone: +351 21 382 8610

In the age of the data deluge there are still many domains and applications restricted to the use of small datasets. The ability to harness these small datasets to solve problems through the use of supervised learning methods can have a significant impact in many important areas. The insufficient size of datasets usually results in unsatisfactory performance of machine learning algorithms. The current research work aims to contribute to mitigate the small dataset problem through the creation of artificial instances, which are added to the training process. The over-sampling algorithm Geometric SMOTE is applied to generate new instances and enhance the initial dataset. Experimental results show a significant improvement in accuracy when compared with the use of the original small dataset and also over other over-sampling techniques such as Random Over-sampling, SMOTE and Borderline SMOTE. These findings show that over-sampling research, developed in the context of imbalanced learning, can also be a valid option to improve accuracy in small data problems.

1 Introduction

Insufficient size of datasets is a common issue in many supervised learning tasks [Niyogi et al., 1998], [Abdul Lateh et al., 2017]. The limited availability of training samples can be caused by different factors. First, data is becoming an increasingly expensive resource [Li et al., 2007] as the process to retain them is getting more complex due to strict privacy regulations such as the General Data Protection Regulation (GDPR) [European Commission, 2019]. Additionally, the small dataset problem can be found in numerous industries where organizations simply do not have access to a reasonable amount of data. For example, manufacturing industries are usually dealing with small number of samples in early stages of product developments and health care organizations have to work with different kinds of rare diseases, where very few records are available [Abdul Lateh et al., 2017].

In machine learning, researchers are usually concerned with the design of sophisticated learning algorithms when aiming to improve prediction performance. However, increasing the sample size is often a more effective approach. A rule of thumb is that ”a dumb algorithm with lots and lots of data beats a clever one with modest amounts of it” [Domingos, 2012]. Generally, small training samples are characterized by a loose data structure with multiple information gaps. This lack of information negatively impacts the performance of machine learning algorithms [Lin et al., 2018]. Consequently, the knowledge gained from models trained with small sample sizes is considered unreliable as well as imprecise and does not lead to a robust performance [Abdul Lateh et al., 2017].

Considering the size of data, there are two types of problems: First, the insufficiency of data belonging to one class (imbalance learning problem) for a binary or multi-class classification task and second, the size of the whole dataset (small dataset problem) for any classification or regression task [Sezer et al., 2014]. In both cases, small training samples affect the performance of machine learning models [Tsai and Li, 2008]. A theoretical definition of ”small” can be found in statistical learning theory by Vapnik. A sample size is defined as small, if the ratio between the number of training samples and Vapnik-Chervonenkis (VC) dimensions is approximately less than 20. VC dimensions are determined as the maximum number of vectors that can be separated into two classes in all possible ways by a set of functions [Vapnik, 2008].

Under-representation of observations in the sample set can be solved in different ways. The use of synthetic data derived from existing observations is a promising approach to the problem [Sezer et al., 2014]. Techniques to artificially add information by extending the sample size, and eventually improving the performance of the algorithms, can translate into significant improvements in many application domains. However, it is important to note that the challenge in artificial data generation is to create data which extend the training set without creating noise [Li and Lin, 2006]. Additionally, generating artificial data will only work if the initial sample is representative of the underlying population. Figure 1 shows the relationship between population, sample and synthetic data.

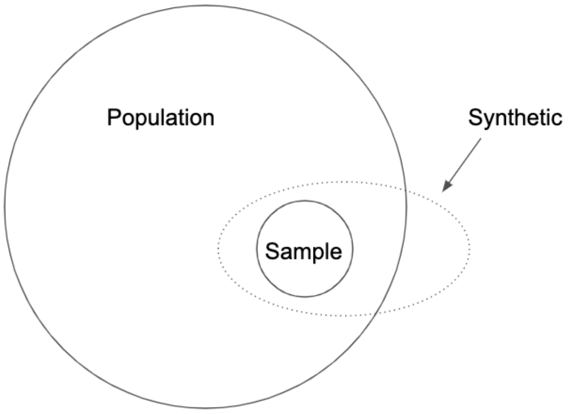

Figure 1: Relationship between population, sample and synthetic data [Li and Lin, 2006].

The next sections will describe an effective way to tackle the small dataset problem. In chapter 2, the previously studied solutions are reviewed. A detailed description of the proposed method is presented in chapter 3. This is followed by the research methodology and the experimental results in chapters 4 and 5. Finally, the paper is concluded with an analysis of the experimental results in chapter 6.

2 Related work

Several data pre-processing methods to increase the data size have been presented by the research community. In this section, the most important approaches are reviewed and the state-of-the-art to improve small dataset learning is reported. We start by describing fuzzy theories, which have historically been the most used approach to mitigate the small dataset problem. Next, we look at re-sampling mechanisms, which mainly consist of bootstrapping techniques, and finally, we review over-sampling methods, developed in the context of imbalanced learning, that can be a valuable option to increase the sample size in small datasets.

2.1 Fuzzy theory

Many artificial sample generation techniques presented in the literature are based on fuzzy theory [Abdul Lateh et al., 2017]. The fuzzy set theory defines a strict mathematical framework to generalize the classical notion of a dataset, providing a wide scope of applicability, especially in the fields of information processing and pattern classification [Zimmermann, 2010]. Based on this concept, several methods have emerged in the last decade to estimate or approximate functions which are generating artificial samples for small datasets.

The fundamental concept of creating synthetic data is called Virtual Sample Generation (VSG) and was originally proposed by [Niyogi et al., 1998]. The idea is to create additional observations based on the current set of examples by using prior information. The introduction of virtual examples expands the effective training set size and can therefore help to mitigate the learning problem. [Niyogi et al., 1998] showed that the process of creating artificial samples is mathematically equivalent to incorporating prior knowledge. They demonstrated the concept on object recognition by mathematically transforming the views of 3D-objects and therefore generating artificial samples.

Based on the above approach, several closely related studies were developed for manufacturing environments. The first method to overcome scheduling problems due to the lack of data in early stages of manufacturing systems was the creation of a Functional Virtual Population (FVP) [Li et al., 2003]. The idea was to create a number of synthetic samples within a newly defined domain range. Although, the process was manually configured, its application dramatically improved the classification accuracy of a neural network.

[Huang and Moraga, 2004] proposed the Diffusion-Neural-Network (DNN) method, an approach that fuzzifies information in order to extend a small dataset. It combines the principle of information diffusion by [Huang, 1997] with traditional Neural Networks to approximate functions. The information diffusion method partially fills the information gaps by using fuzzy theory to represent the similarities between samples and subsequently derive new ones.

In order to fully fill the information gaps, Mega-Trend-Diffusion (MTD) [Li et al., 2007] combines data trend estimation with a diffusion technique to estimate the domain range, thus avoiding over-estimation. It diffuses a set of data instead of each sample individually. It is considered as an improvement of DNN and was initially developed to improve early flexible manufacturing system scheduling accuracy. In further research, MTD is widely used as a synthetic sample generation method and is recognized as an effective way to deal with small dataset problems [Abdul Lateh et al., 2017].

MTD only considers the data for independent attributes and does not deal with their relationships. Genetic Algorithm Based Virtual Sample Generation was proposed that takes the relationship among the attributes into account and explores the integrated effects of attributes instead of dealing with them individually. The algorithm has three steps: Initially, samples are randomly selected to determine the range of each attribute by using MTD functions. Next, a Genetic Algorithm is applied to find the most feasible virtual samples. Finally, the average error of these new samples is calculated. The results outperformed the ones using MTD and also showed better performance in prediction than in the case of no generation of synthetic samples [Li and Wen, 2014, Lin and Li, 2010].

2.2 Random over-sampling

An alternative approach to fuzzy theory as well the most well-known artificial sample generation method is the Bootstrapping Procedure [Abdul Lateh et al., 2017] or Random Over-Sampling (ROS) as is known in the imbalanced learning research area. The main difference to the previously presented techniques is that ROS creates expands the training set by re-sampling instances from the original dataset with replacement [Efron and Tibshirani, 1993]. Therefore, it allows the algorithms to use the same sample more than one time to gradually revise the identified patterns in order to improve predictive accuracy. However, ROS may cause over-fitting when applied to small data because it repetitively uses the same information [Tsai and Li, 2015], [Li et al., 2018]. Nevertheless, [Ivănescu et al., 2006] applied ROS in batch process industries where it was shown that it may help mitigate the small data problem.

2.3 Informed over-sampling

A different approach to fill information gaps is informed over-sampling. It is an artificial data generation strategy originally developed in the context of machine learning to deal with the imbalanced learning problem. Therefore, its origin comes from a different research community than the fuzzy and re-sampling strategies presented above. Although the small data and imbalanced learning problems are similar, it seems that their proposed solutions had very few connections so far.

In the imbalanced learning problem, the classes of the given dataset are significantly skewed i.e. the dataset has a large number of observations in one of the classes, called majority class, and a relatively small number of observations in the other class(es), called minority class(es). This constitutes a problem for the learning phase of the algorithm resulting in low accuracy for the minority class(es). In practice, the imbalanced dataset problem is very common issue in supervised learning. Especially, in the fields of fraud detection, product categorization, disease diagnosis and customer churn prediction, an imbalanced dataset is the norm rather than the exception [He and Ma, 2013, Chen et al., 2012, Verbeke et al., 2012].

2.3.1 SMOTE

There are several methods presented in the literature that belong in the over-sampling category. The first method to be proposed and still the most popular is the Synthetic Minority Over-sampling TEchnique (SMOTE). SMOTE is based on the idea of k-nearest neighbors and linear interpolation as a data generation mechanism. More specifically, SMOTE proposes to form a line segment between neighboring minority class instances and generate synthetic data between them [v. Chawla et al., 2002]. The algorithm is very popular due to its simplicity as well as its robustness. Numerous variations have been proposed based on SMOTE, increasing its status as the staple idea in over-sampling for imbalanced learning problems [Fernandez et al., 2018]. However, SMOTE has some significant limitations when it comes to the sample generation process. In practice, the separation between majority and minority class areas is often not clearly definable. Thus, noisy samples may be generated when a minority sample lies in the region of the majority classes. Figure 2 presents a scenario where a minority instance is generated within the majority region (noisy sample).

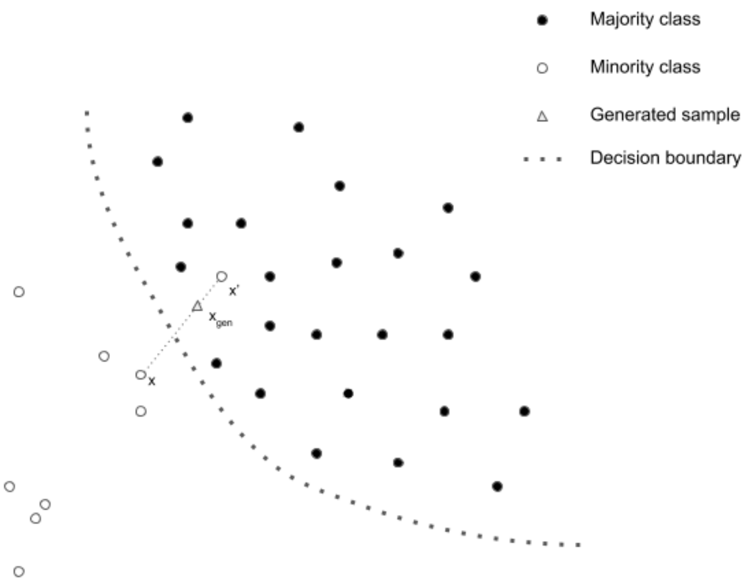

Figure 2: Generation of noisy examples.

Furthermore, redundant instances may be generated within dense minority regions, as so that they do not add any relevant information to the classifier and may lead to over-fitting. Figure 3 demonstrates an example where a minority class instance is generated in a dense minority class. This new observation belongs to the same dense cluster as the original and is therefore less useful.

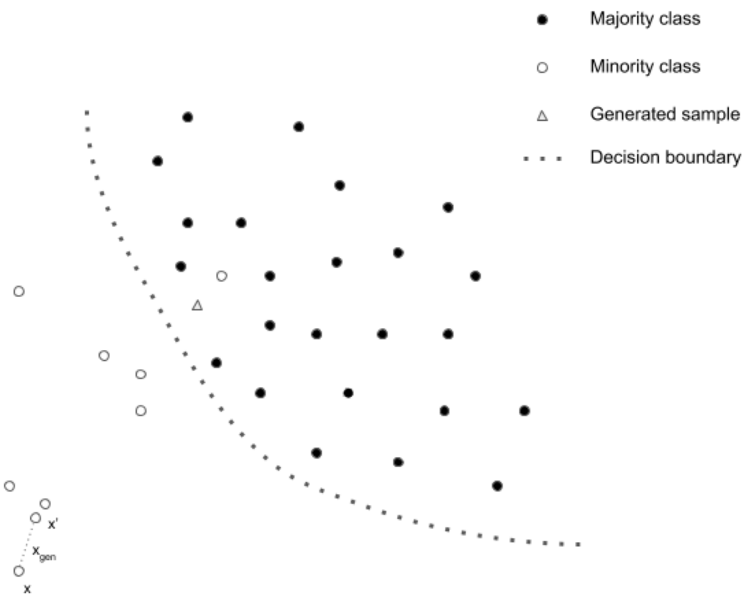

Figure 3: Generation of redundant examples.

Although SMOTE is recognized as an over-sampling technique for imbalanced datasets, it can also be used for solving the small dataset problem. [Li et al., 2018] showed that SMOTE is able to successfully fill the information gaps with synthetic samples. However, given its limitations, SMOTE did not achieve the best results within their study.

2.3.2 G-SMOTE

The novel data generation procedure Geometric SMOTE (G-SMOTE) has been presented with the objective to improve the above mentioned limitations of the SMOTE algorithm [Douzas and Bacao, 2019]. G-SMOTE can be seen as a substitute of SMOTE, enhanced by geometric properties. Compared to SMOTE, it expands the data generation area and prevents the creation of noise. Instead of connecting a minority sample and one of its minority class nearest neighbors with a line segment, the instances are generated in a geometrical region around the minority sample. Furthermore, G-SMOTE is designed to avoid the generation of noisy samples by introducing the *selection strategy* hyper-parameter. Figure 4 demonstrates the distribution of artificially created samples by SMOTE versus G-SMOTE. Increasing number of k neighbors, SMOTE tends to generate noisy samples, whereas G-SMOTE avoids this scenario.

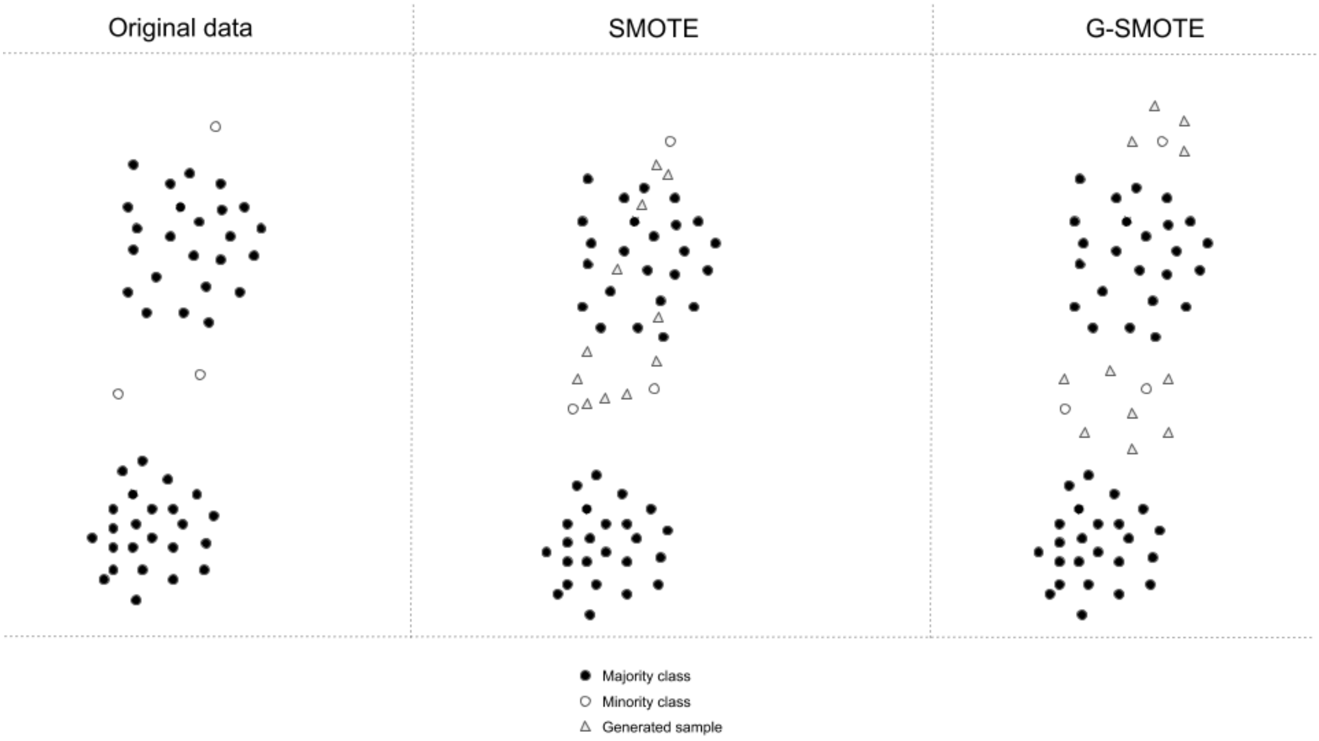

Figure 4: Three positive class instances are used by SMOTE and G-SMOTE to generate synthetic data. G-SMOTE generates non-noisy samples with higher variety than SMOTE.

The experimental study of [Douzas and Bacao, 2019] includes an extensive comparison between G-SMOTE and SMOTE using 69 imbalanced datasets and several classifiers. The results show that G-SMOTE outperforms SMOTE, Random Over-sampling and the case of no over-sampling, across all datasets, classifiers and performance metrics.

3 Proposed method

In the following section, we present G-SMOTE as a novel data generation procedure for small datasets in the case of binary classification. Originally developed for the imbalanced learning problem, we adapt the algorithm to not only re-sample the minority class, but the entire dataset independent from the class distribution.

3.1 G-SMOTE algorithm

As mentioned above, the G-SMOTE algorithm randomly generates artificial data within a geometrical region of the input space. The size of this area is derived from the distance of the selected sample to one of its nearest neighbors, whereas the shape is determined by the hyper-parameters called *truncation factor* and *deformation factor*. Additionally, the *selection strategy* hyper-parameter of G-SMOTE modifies the standard SMOTE selection process and also affects the size of the geometric region. Although the main concept can be adapted from the original method to the small dataset problem, the *selection strategy* requires some minor adjustments which will be described hereafter.

In what follows, G-SMOTE is applied to the case of binary classification tasks with the objective to generate artificial data for both classes, called arbitrarily the positive and negative class. The application for the multi-class case is also straightforward and it is based on the binarization of the problem through the one-vs-all approach. Finally, regression tasks require an extensive modification of the original G-SMOTE algorithm and they will be a topic of future research.

3.2 Adapted G-SMOTE algorithm

The inputs of the G-SMOTE algorithm are the positive and negative class samples S_{pos}, S_{neg} respectively, the three geometric hyper-parameters *truncation factor*, *deformation factor* and *selection strategy* as well as the number of generated samples for the positive class N_{pos} and for the negative class N_{neg}. A sensible choice for the last two inputs, used also in the experimental procedure below, is to preserve the class distribution in the re-sampled dataset. The adapted G-SMOTE algorithm can be generally described in the following steps:

1. An empty set S_{gen} is initialized. S_{gen} will be populated with artificial data from both classes.

2. S_{pos} is shuffled and the process described below is repeated N_{pos} times until N_{pos} artificial points have been generated.

   2.1. A positive class instance $\mathbf{x}_{center}$ is selected randomly from as N_{pos} the center of the geometric region.

   2.2. Depending on the values of α_{sel} ($positive, negative$ or *combined*), this step results in a randomly selected sample $\mathbf{x}_{surface}$ which belongs to either S_{pos} or S_{neg}.

   2.3. A random point $\mathbf{x}_{gen}$ is generated inside the hyper-spheroid centered at $\mathbf{x}_{center}$. The major axis of the hyper-spheroid is defined by $\mathbf{x}_{surface} - \mathbf{x}_{center}$ while the permissible data generation area as well as the rest of geometric characteristics are determined by the hyper-parameters *truncation factor* and *deformation factor*.

   2.4. $\mathbf{x}_{gen}$ is added to the set of generated samples $\mathbf{S}_{gen}$.

3. Step 2 is repeated using the substitution $pos \leftrightarrow neg$ until N_{neg} artificial points have been generated.

Therefore, the adapted G-SMOTE algorithm applies independently the original G-SMOTE algorithm for both the positive and negative class. The above description of the algorithm excludes mathematical formulas and details which can be found in [Douzas and Bacao, 2019]. Figure 5 shows an example of the adapted G-SMOTE data generation process for the three different values of the *selection strategy* hyper-parameter when positive class data generation is considered.

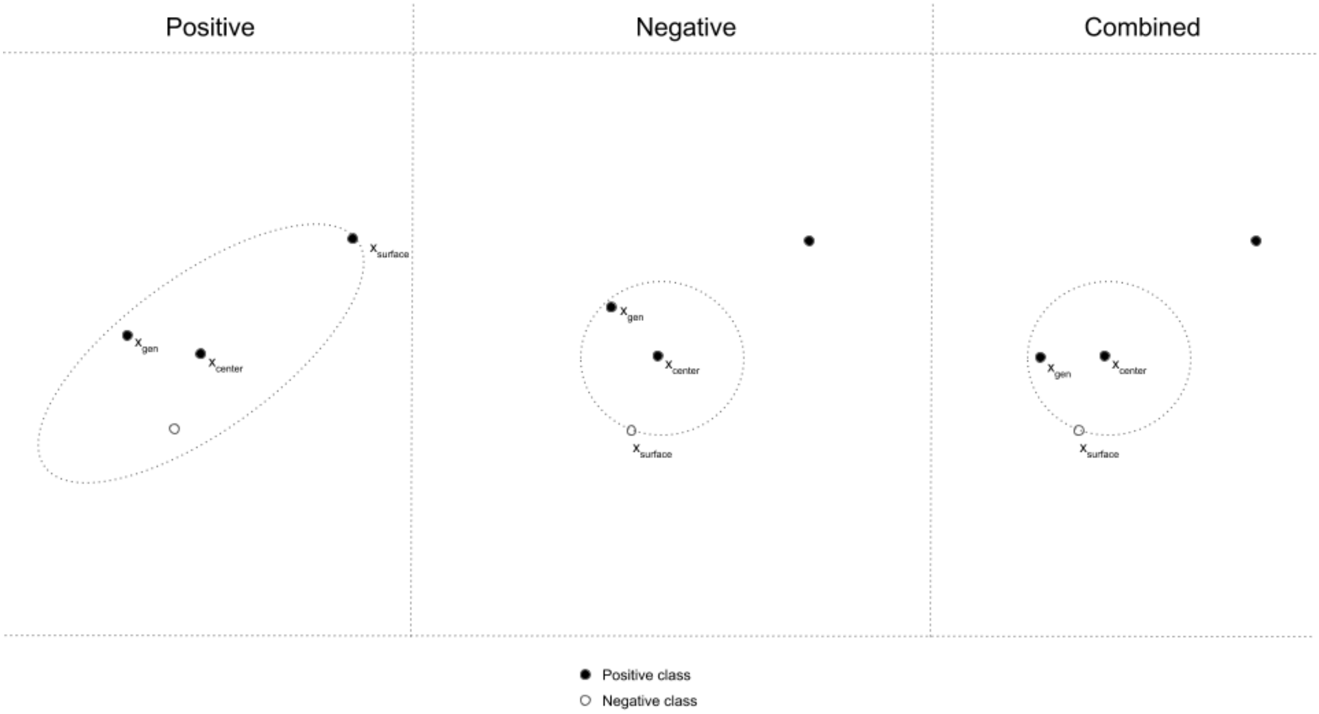

Figure 5: The G-SMOTE data generation mechanism for the three different *selection strategy* values, when positive class samples are generated values. The hyper-parameters *deformation factor* and *truncation factor* have also different values resulting in the permissible data generation areas of the figure.

4 Research methodology

The main objective of this work is to compare G-SMOTE with other over-sampling techniques when it comes to the small dataset problem. Therefore, we use a variety of datasets, evaluation measures and classifiers to evaluate the performance of over-samplers. A description of this set-up, the experimental procedure as well as the software implementation is provided in this section.

4.1 Experimental data

Ten datasets are used to test the performance of G-SMOTE which are retrieved from UCI Machine Learning Repository [Dua and Graff, 2019]. The focus on the selection of the data lies on binary classification problems with a balanced distribution of the two classes. In order to assure generalizability of the results, the datasets include different topics such as health care, finance, business and physics as well as different sample sizes. Details of the datasets are presented in the following table:

Dataset	Number of samples	Number of attributes	Area
Arcene	900	10.000	Health Care
Audit	776	18	Business
Banknote Authentication	1.372	5	Finance
Spambase	4.610	57	Business
Breast Cancer	699	10	Health Care
Indian Liver Patient	583	10	Health Care
Ionosphere	351	34	Physics
MAGIC Gamma Telescope	19.020	11	Physics
Musk	6.598	168	Physics
Parkinsons	197	23	Health Care

Table 1: Description of the datasets

4.2 Evaluation measures

To evaluate the performance of G-SMOTE, the experiment includes two different measures. First, *Accuracy* is used as one of the most common metrics for evaluating classification models [Hossin and Sulaiman, 2015]. *Accuracy* measures the ratio of correct predictions over the total number of instances. The mathematical formula is the following:

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

where TP, TN denote the number of correctly classified positive and negative instances respectively while FP, FN denote the number of misclassified negative and positive instances, respectively. The *Accuracy* metric might be inappropriate for datasets with a significant difference between the number of positive and negative classes since rare classes have a small impact to the final outcome compared to the majority classes. To make sure the contribution in the accuracies of the two classes stay relatively balanced, we include the geometric mean score (*G-Mean*) as a second measure. *G-Mean* is the geometric mean of *sensitivity* and *specificity*:

$$\mathit{G\text{-}Mean} = \sqrt{\mathit{sensitivity} \times specificity} = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}}$$

4.3 Machine learning algorithms

The experiment is conducted using several classifiers: Logistic Regression (LR) [McCullagh and Nelder, 2019], K-Nearest Neighbors (KNN) [Cover and Hart, 1967], Decision Tree (DT) [Salzberg, 1994] and Gradient Boosting (GB) [Friedman, 2001].

4.4 Experimental procedure

As explained above, the main goal of the paper is to evaluate how well various over-samplers, particularly G-SMOTE, are able to enhance small datasets with artificial samples compared to the case where the original data are used. In order to evaluate them, various classifiers and metrics are selected. We use k-fold cross-validation scores with $k = 5$ to assess the performance of the models for each combination of over-sampler and classifier. The dataset D is randomly split into k subsets (folds) $D_1, D_2, \ldots D_k$ of approximately equal size. Each fold is used as a validation set and the remaining folds are used to train the model with k iterations. This procedure is repeated until each k have been used as a validation set [Han et al., 2012]. Therefore, the experimental procedure for each of the datasets presented in table 1 and for each cross-validation stage consists of the following steps:

1. The data are divided into k-folds.

2. The $k-1$ folds are under-sampled using an undersampling ratio 50, 75, 90 and 95, corresponding to the percentage of the dataset that is removed.

3. Over-sampling is applied to the under-sampled data of the previous steps that increases their size and class distribution back to the initial.

4. The re-sampled data of the previous step are used to train the classifiers.

5. The classifiers are evaluated on the remaining fold of step 1.

The results also include the case where the classifiers are trained on the original data, called the benchmark performance. Figure 6 visualizes the experimental procedure:

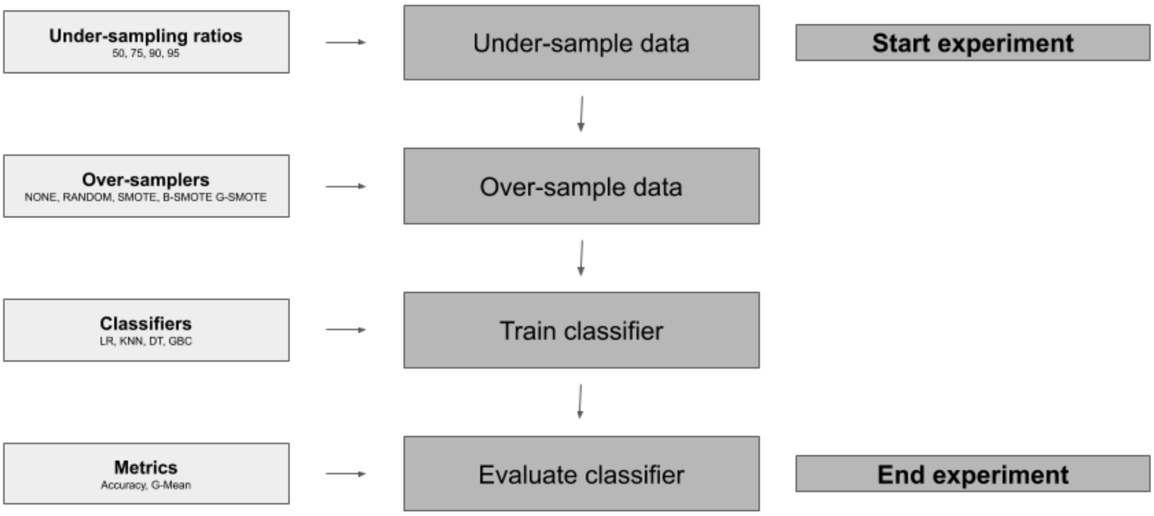

Figure 6: Visualization of experimental procedure.

This procedure results in a cross validation score for each combination of dataset, classifier, oversampler and evaluation metric. It is also repeated three times and the average cross-validation score is calculated. The algorithms used in the experiment have various hyper-parameters that yield to different scores. The maximum of these scores is reported.

In order to confirm the statistical significance of the experimental results, the Friedman test [Sheldon et al., 1996] as well as the Holm test [Demšar, 2006] are applied. Ranking scores are assigned to each over-sampling method with scores of 1 to 5 for the best and worst performing methods, respectively.

The Friedman test is a non-parametric procedure that compares the average rankings of the algorithms under the null hypothesis that all show identical performance independent of the selected classifier and evaluation metric. If the null-hypothesis is rejected to our favor, we proceed with the Holm test. The Holm test acts as a post-hoc test for the Friedman test for controlling the family-wise error rate when all algorithms are compared to a control method. It is a powerful non-parametric test in situations where we want to test whether a newly proposed method is better than existing ones. The control method in our case is the proposed G-SMOTE method and is tested under the null hypothesis that it performs similarly to the rest of over-samplers for every combination of classifier and metric.

4.5 Software Implementation

The implementation of the experimental procedure is based on Python programming language using the Scikit-Learn library [Pedregosa et al., 2011]. All functions, algorithms, experiments and results reported are provided at the GitHub repository of the project. Additionally, Research-Learn library provides a framework to implement comparative experiments, being also fully integrated with the Scikit-Learn ecosystem.

5 Results and discussion

In this section the performance of the different over-samplers and the results of the statistical tests are presented and analyzed.

5.1 Comparative presentation

The mean cross validation scores and the standard error per classifier and metric across all datasets are presented in Table 2. The scores are presented for each under-sampling ratio to evaluate how the methods perform as the dataset size diminishes. We also include the benchmark method that represents the original dataset without applying under-sampling before training the algorithm. The benchmark method is expected to obtain the best results by design.

Ratio	Classifier	Metric	NONE	RANDOM	SMOTE	B-SMOTE	G-SMOTE	BENCHMARK
50	LR	ACCURACY	0.91 ± 0.03	0.91 ± 0.03	0.91 ± 0.02	0.91 ± 0.03	0.92 ± 0.02	0.92 ± 0.02
50	LR	G-MEAN	0.88 ± 0.04	0.88 ± 0.04	0.89 ± 0.04	0.89 ± 0.04	0.89 ± 0.04	0.90 ± 0.04
50	KNN	ACCURACY	0.88 ± 0.03	0.88 ± 0.03	0.89 ± 0.03	0.88 ± 0.03	0.89 ± 0.03	0.90 ± 0.03
50	KNN	G-MEAN	0.84 ± 0.04	0.85 ± 0.04	0.86 ± 0.04	0.85 ± 0.04	0.86 ± 0.04	0.87 ± 0.04
50	DT	ACCURACY	0.88 ± 0.04	0.88 ± 0.04	0.88 ± 0.04	0.88 ± 0.04	0.90 ± 0.03	0.90 ± 0.03
50	DT	G-MEAN	0.86 ± 0.05	0.86 ± 0.05	0.87 ± 0.05	0.87 ± 0.05	0.89 ± 0.04	0.89 ± 0.03
50	GBC	ACCURACY	0.91 ± 0.04	0.92 ± 0.03	0.92 ± 0.03	0.91 ± 0.04	0.93 ± 0.03	0.94 ± 0.02
50	GBC	G-MEAN	0.90 ± 0.04	0.90 ± 0.04	0.91 ± 0.03	0.90 ± 0.04	0.92 ± 0.03	0.93 ± 0.03
75	LR	ACCURACY	0.90 ± 0.03	0.89 ± 0.03	0.89 ± 0.03	0.89 ± 0.03	0.90 ± 0.03	0.92 ± 0.02
75	LR	G-MEAN	0.86 ± 0.05	0.86 ± 0.05	0.87 ± 0.04	0.87 ± 0.04	0.87 ± 0.04	0.90 ± 0.04
75	KNN	ACCURACY	0.86 ± 0.04	0.86 ± 0.04	0.87 ± 0.04	0.85 ± 0.04	0.87 ± 0.04	0.90 ± 0.03
75	KNN	G-MEAN	0.80 ± 0.06	0.82 ± 0.05	0.84 ± 0.04	0.83 ± 0.05	0.84 ± 0.04	0.87 ± 0.04
75	DT	ACCURACY	0.86 ± 0.05	0.86 ± 0.05	0.86 ± 0.05	0.85 ± 0.06	0.89 ± 0.04	0.90 ± 0.03
75	DT	G-MEAN	0.83 ± 0.06	0.84 ± 0.05	0.84 ± 0.06	0.83 ± 0.06	0.86 ± 0.05	0.89 ± 0.03
75	GBC	ACCURACY	0.87 ± 0.05	0.88 ± 0.05	0.88 ± 0.05	0.88 ± 0.05	0.90 ± 0.04	0.94 ± 0.02
75	GBC	G-MEAN	0.85 ± 0.06	0.85 ± 0.06	0.86 ± 0.05	0.85 ± 0.06	0.89 ± 0.04	0.93 ± 0.03
90	LR	ACCURACY	0.86 ± 0.04	0.86 ± 0.04	0.86 ± 0.04	0.85 ± 0.04	0.87 ± 0.04	0.92 ± 0.02
90	LR	G-MEAN	0.81 ± 0.06	0.82 ± 0.06	0.82 ± 0.06	0.82 ± 0.05	0.83 ± 0.06	0.90 ± 0.04
90	KNN	ACCURACY	0.81 ± 0.05	0.82 ± 0.05	0.82 ± 0.05	0.81 ± 0.05	0.83 ± 0.05	0.90 ± 0.03

Ratio	Classifier	Metric	NONE	RANDOM	SMOTE	B-SMOTE	G-SMOTE	BENCHMARK
90	KNN	G-MEAN	0.69 ± 0.10	0.76 ± 0.07	0.78 ± 0.06	0.74 ± 0.09	0.78 ± 0.06	0.87 ± 0.04
90	DT	ACCURACY	0.84 ± 0.05	0.83 ± 0.05	0.83 ± 0.06	0.83 ± 0.05	0.87 ± 0.04	0.90 ± 0.03
90	DT	G-MEAN	0.81 ± 0.06	0.81 ± 0.06	0.80 ± 0.06	0.80 ± 0.06	0.84 ± 0.05	0.89 ± 0.03
90	GBC	ACCURACY	0.84 ± 0.06	0.84 ± 0.06	0.84 ± 0.06	0.84 ± 0.05	0.88 ± 0.04	0.94 ± 0.02
90	GBC	G-MEAN	0.82 ± 0.06	0.81 ± 0.06	0.81 ± 0.07	0.81 ± 0.06	0.86 ± 0.05	0.93 ± 0.03
95	LR	ACCURACY	0.83 ± 0.05	0.83 ± 0.05	0.83 ± 0.05	0.83 ± 0.04	0.84 ± 0.05	0.92 ± 0.02
95	LR	G-MEAN	0.75 ± 0.08	0.76 ± 0.07	0.76 ± 0.07	0.77 ± 0.07	0.76 ± 0.08	0.90 ± 0.04
95	KNN	ACCURACY	0.79 ± 0.05	0.79 ± 0.05	0.81 ± 0.05	0.79 ± 0.05	0.81 ± 0.05	0.90 ± 0.03
95	KNN	G-MEAN	0.60 ± 0.13	0.69 ± 0.09	0.71 ± 0.09	0.74 ± 0.06	0.73 ± 0.07	0.87 ± 0.04
95	DT	ACCURACY	0.81 ± 0.05	0.81 ± 0.05	0.82 ± 0.05	0.81 ± 0.05	0.85 ± 0.05	0.90 ± 0.03
95	DT	G-MEAN	0.77 ± 0.06	0.78 ± 0.06	0.78 ± 0.06	0.78 ± 0.06	0.81 ± 0.06	0.89 ± 0.03
95	GBC	ACCURACY	0.82 ± 0.05	0.83 ± 0.05	0.83 ± 0.05	0.82 ± 0.05	0.85 ± 0.05	0.94 ± 0.02
95	GBC	G-MEAN	0.77 ± 0.07	0.78 ± 0.07	0.78 ± 0.07	0.78 ± 0.07	0.81 ± 0.07	0.93 ± 0.03

Table 2: Results for mean cross validation scores of oversamplers (NONE corresponds to No Over-sampling, RANDOM to Random Over-sampling and B-SMOTE to Borderline SMOTE)

The above table shows that G-SMOTE outperforms all over-sampling methods almost for all combinations of classifiers and metrics. Throughout the scores we can observe that all over-samplers have a better performance as the dataset increase their size i.e. the under-sampling ratio gets smaller. Particularly, the scores of G-SMOTE are the closest to the ones of the benchmark method which implies that it is able to reconstruct the original dataset more effectively compared to the rest of the over-samplers.

Table 3 presents the mean and standard error of percentage difference between G-SMOTE and No Over-sampling. It shows that G-SMOTE performs significantly better compared to the case where no over-sampling is applied in every combination of under-sampling ratio, classifier and metric. Particularly, the performance gap increases for higher under-sampling ratios.

Ratio	Classifier	Metric	% Difference
50	LR	ACCURACY	0.52 ± 0.27
50	LR	G-MEAN	0.36 ± 0.14
50	KNN	ACCURACY	1.30 ± 0.45
50	KNN	G-MEAN	2.48 ± 0.96
50	DT	ACCURACY	2.58 ± 1.02
50	DT	G-MEAN	3.72 ± 1.61
50	GBC	ACCURACY	2.75 ± 1.42
50	GBC	G-MEAN	2.90 ± 1.46
75	LR	ACCURACY	0.40 ± 0.15
75	LR	G-MEAN	1.05 ± 0.58
75	KNN	ACCURACY	1.93 ± 0.50
75	KNN	G-MEAN	7.27 ± 4.51
75	DT	ACCURACY	4.13 ± 1.88
75	DT	G-MEAN	4.67 ± 1.97
75	GBC	ACCURACY	4.39 ± 2.51
75	GBC	G-MEAN	5.67 ± 3.00
90	LR	ACCURACY	1.41 ± 0.52
90	LR	G-MEAN	3.26 ± 1.58
90	KNN	ACCURACY	2.95 ± 1.21
90	KNN	G-MEAN	33.43 ± 26.93
90	DT	ACCURACY	4.47 ± 1.46
90	DT	G-MEAN	4.32 ± 1.88
90	GBC	ACCURACY	5.17 ± 2.48
90	GBC	G-MEAN	5.64 ± 2.35
95	LR	ACCURACY	1.40 ± 0.63
95	LR	G-MEAN	1.23 ± 3.71
95	KNN	ACCURACY	2.94 ± 1.28

Ratio	Classifier	Metric	% Difference
95	KNN	G-MEAN	23.66 ± 20.31
95	DT	ACCURACY	5.00 ± 2.04
95	DT	G-MEAN	5.18 ± 1.79
95	GBC	ACCURACY	4.11 ± 1.96
95	GBC	G-MEAN	5.25 ± 2.43

Table 3: Results for percentage difference between G-SMOTE and SMOTE

As explained in section 4, a ranking score in the range 1 to 5 is assigned to each oversampler. The mean ranking of all over-sampling methods across the datasets is presented in the following figure:

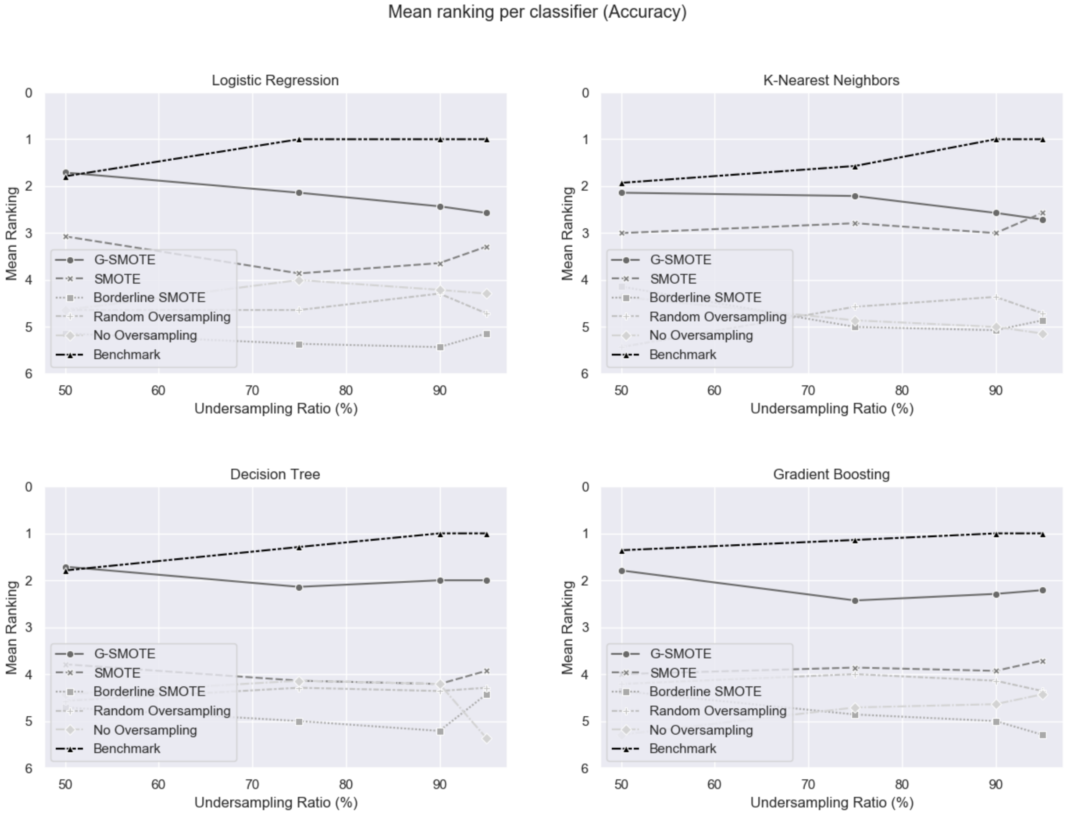

Figure 7: Mean ranking per classifier (Accuracy)

Looking at the graphs, G-SMOTE is ranked on the top place when comparing with SMOTE, Borderline SMOTE, Random Over-sampling and No Over-sampling. Additionally, G-SMOTE slightly outperforms the benchmark method using the classifiers Logistic Regression and Decision Tree in the mean ranking.

5.2 Statistical Analysis

To confirm the significance of the above presented results we apply the Friedman test as well as the Holm Test on the above results. The application of the Friedman test is presented below:

Classifier	Metric	p-value	Significance
LR	ACCURACY	1.2e-11	True
LR	G-MEAN	6.9e-08	True
KNN	ACCURACY	2.7e-12	True
KNN	G-MEAN	3.5e-13	True
DT	ACCURACY	2.9e-12	True
DT	G-MEAN	6.7e-11	True
GBC	ACCURACY	4.9e-11	True
GBC	G-MEAN	1.7e-09	True

Table 4: Results for Friedman test

Therefore, the null hypothesis of the Friedman test is rejected at a significance level of a = 0.05, i.e. the over-samplers do not perform similarly in the mean rankings for any combination of classifier and evaluation metric.

The Holm method is applied to adjust the p-values of the paired difference test with G-SMOTE algorithm as the control method. The results are shown in table 5:

Classifier	Metric	NONE	RANDOM	SMOTE	B-SMOTE
LR	ACCURACY	2.9e-04	7.6e-05	2.9e-04	5.4e-05
LR	G-MEAN	2.1e-01	2.1e-01	1.0e+00	1.0e+00
KNN	ACCURACY	2.7e-05	7.8e-08	1.4e-01	1.8e-04
KNN	G-MEAN	1.1e-02	3.3e-04	2.9e-01	2.9e-01
DT	ACCURACY	1.5e-05	1.5e-05	4.8e-05	3.3e-05
DT	G-MEAN	1.3e-05	4.4e-05	4.4e-05	4.4e-05
GBC	ACCURACY	2.2e-04	2.9e-04	5.8e-04	1.8e-04
GBC	G-MEAN	1.8e-04	3.9e-04	7.3e-04	7.3e-04

Table 5: Adjusted p-values using Holm test (B-SMOTE corresponds to Borderline SMOTE)

At a significance level of a = 0.05 the null hypothesis of the Holm's test is rejected for 25 out 32 combinations. This indicates that the proposed method outperforms all other methods in most cases.

6 Conclusions

This paper illustrates an effective solution to mitigate the small dataset problem in binary classification tasks. As shown above, the over-sampling algorithm G-SMOTE has the ability to generate high quality artificial samples and improve the prediction accuracy of the classifiers used in the experiments. This improvement relates to its capability of increasing the diversity of new instances while avoiding the generation of noisy samples. An important point is that G-SMOTE significantly improves classification performance compared to the case where only the small data are used. Also G-SMOTE outperforms standard over-sampling approaches such as Random Over-Sampling and SMOTE, being closer to the benchmark scores than any of them. Finally, G-SMOTE implementation is available as an open source project.

References

[Abdul Lateh et al., 2017] Abdul Lateh, M., Kamilah Muda, A., Izzah Mohd Yusof, Z., Azilah Muda, N., and Sanusi Azmi, M. (2017). Handling a small dataset problem in prediction model by employ artificial data generation approach: A review. *Journal of Physics: Conference Series*, 892:012016.

[Chen et al., 2012] Chen, Z.-Y., Shu, P., and Sun, M. (2012). A hierarchical multiple kernel support vector machine for customer churn prediction using longitudinal behavioral data. *European Journal of Operational Research*, 223:461–472.

[Cover and Hart, 1967] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

[Demšar, 2006] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

[Domingos, 2012] Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78.

[Douzas and Bacao, 2019] Douzas, G. and Bacao, F. (2019). Geometric smote a geometrically enhanced drop-in replacement for smote. *Information Sciences*, 501:118–135.

[Dua and Graff, 2019] Dua, D. and Graff, C. (2019). Uci machine learning repository.

[Efron and Tibshirani, 1993] Efron, B. and Tibshirani, R. (1993). *An introduction to the bootstrap*, volume 57 of *Monographs on statistics and applied probability.* Chapman & Hall, New York.

[European Commission, 2019] European Commission (2019). Data protection under gdpr.

[Fernandez et al., 2018] Fernandez, A., Garcia, S., Herrera, F., and v. Chawla, N. (2018). Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, 61:863–905.

[Friedman, 2001] Friedman, J. H. (2001). machine. *The Annals of Statistics*, 29(5):1189–1232.

[Han et al., 2012] Han, J., Kamber, M., and Pei, J. (2012). *Data mining: Concepts and techniques.* Morgan Kaufmann and Elsevier Science, Waltham, MA, 3rd ed. edition.

[He and Ma, 2013] He, H. and Ma, Y., editors (2013). *Imbalanced learning: Foundations, algorithms, and applications.* IEEE Press, Wiley, Hoboken, NJ.

[Hossin and Sulaiman, 2015] Hossin, M. and Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):01–11.

[Huang, 1997] Huang, C. (1997). Principle of information diffusion. *Fuzzy Sets and Systems*, 91(1):69–90.

[Huang and Moraga, 2004] Huang, C. and Moraga, C. (2004). A diffusion-neural-network for learning from small samples. *International Journal of Approximate Reasoning*, 35(2):137–161.

[Ivănescu et al., 2006] Ivănescu, V. C., Bertrand, J. W. M., Fransoo, J. C., and Kleijnen, J. P. C. (2006). Bootstrapping to solve the limited data problem in production control: an application in batch process industries. *Journal of the Operational Research Society*, 57(1):2–9.

[Li et al., 2003] Li, D.-C., Chen, L.-S., and Lin, Y.-S. (2003). Using functional virtual population as assistance to learn scheduling knowledge in dynamic manufacturing environments. *International Journal of Production Research*, 41(17):4011–4024.

[Li et al., 2018] Li, D.-C., Lin, W.-K., Chen, C.-C., Chen, H.-Y., and Lin, L.-S. (2018). Rebuilding sample distributions for small dataset learning. *Decision Support Systems*, 105:66–76.

[Li and Lin, 2006] Li, D.-C. and Lin, Y.-S. (2006). Using virtual sample generation to build up management knowledge in the early manufacturing stages. *European Journal of Operational Research*, 175(1):413–434.

[Li and Wen, 2014] Li, D.-C. and Wen, I.-H. (2014). A genetic algorithm-based virtual sample generation technique to improve small data set learning. *Neurocomputing*, 143:222–230.

[Li et al., 2007] Li, D.-C., Wu, C.-S., Tsai, T.-I., and Lina, Y.-S. (2007). Using mega-trend-diffusion and artificial samples in small data set learning for early flexible manufacturing system scheduling knowledge. *Computers & Operations Research*, 34(4):966–982.

[Lin et al., 2018] Lin, L.-S., Li, D.-C., Chen, H.-Y., and Chiang, Y.-C. (2018). An attribute extending method to improve learning performance for small datasets. *Neurocomputing*, 286:75–87.

[Lin and Li, 2010] Lin, Y.-S. and Li, D.-C. (2010). The generalized-trend-diffusion modeling algorithm for small data sets in the early stages of manufacturing systems. *European Journal of Operational Research*, 207:121–130.

[McCullagh and Nelder, 2019] McCullagh, P. and Nelder, J. A. (2019). *Generalized Linear Models*. Routledge.

[Niyogi et al., 1998] Niyogi, P., Girosi, F., and Poggio, T. (1998). Incorporating prior information in machine learning by creating virtual examples. *Proceedings of the IEEE*, 86(11):2196–2209.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., and Thirion, B. (2011). Scikit-learn: Machine learning in python. 12.

[Salzberg, 1994] Salzberg, S. L. (1994). C4.5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16(3):235–240.

[Sezer et al., 2014] Sezer, E. A., Nefeslioglu, H. A., and Gokceoglu, C. (2014). An assessment on producing synthetic samples by fuzzy c-means for limited number of data in prediction models. *Applied Soft Computing*, 24:126–134.

[Sheldon et al., 1996] Sheldon, M. R., Fillyaw, M. J., and Thompson, W. D. (1996). The use and interpretation of the friedman test in the analysis of ordinal-scale data in repeated measures designs. *Physiotherapy Research International*, 1(4):221–228.

[Tsai and Li, 2015] Tsai, C.-H. and Li, D.-C., editors (2015). *Improving Knowledge Acquisition Capability of M5' Model Tree on Small Datasets*. IEEE.

[Tsai and Li, 2008] Tsai, T.-I. and Li, D.-C. (2008). Utilize bootstrap in small data set learning for pilot run modeling of manufacturing systems. *Expert Systems with Applications*, 35(3):1293–1300.

[v. Chawla et al., 2002] v. Chawla, N., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002).

Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

[Vapnik, 2008] Vapnik, V. N. (2008). *The nature of statistical learning theory.* Statistics for engineering and information science. Springer, New York, 2. ed., 6. print edition.

[Verbeke et al., 2012] Verbeke, W., Dejaeger, K., Martens, D., Hur, J., and Baesens, B. (2012). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218:211–229.

[Zimmermann, 2010] Zimmermann, H.-J. (2010). Fuzzy set theory. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(3):317–332.