Annals of Operations Research manuscript No.
(will be inserted by the editor)

Geometric Small Data Oversampling Technique

A new algorithm for generating synthetic samples to improve small data prediction accuracy

Georgios Douzas · Fernando Bacao · Maria Lechleitner

Received: date / Accepted: date

Abstract 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 training data usually results in unsatisfactory performance of machine learning algorithms. The current research work aims to contribute to mitigate the small data problem through the creation of artificial instances, which are added to the training process. The proposed algorithm, Geometric Small Data Oversampling Technique, uses geometric regions around existing samples to generate new high quality instances. Experimental results show a significant improvement in accuracy when compared with the use of the initial small dataset and also over other oversampling techniques such as Random OverSampling, SMOTE and Borderline SMOTE.

Keywords Machine Learning · Classification · Small Data Problem · Oversampling

1 Introduction

Insufficient size of datasets is a common issue in many supervised learning tasks [27], [17]. The limited availability of training samples can be caused by different factors. First, data is becoming an increasingly expensive resource [23] as the process to retain them is getting more complex due to strict privacy regulations such as the General Data Protection Regulation (GDPR) [9]. 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 a small number of samples in the early stages of product development while health care organizations have to work with different kinds of rare diseases, where very few records are available [17].

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" [5]. Generally, a small number of training samples is characterized by a loose data structure with multiple information gaps. This lack of information negatively impacts the performance of machine learning algorithms [24]. 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 [17].

Considering the size of data, there are two types of problems: First, the insufficiency of data belonging to one or more of the classes (imbalance learning problem) for a binary or multi-class classification task and second, the

G. Douzas
NOVA Information Management School
Campus de Campolide, 1070-312 Lisboa, Portugal
Tel.: +351 21 382 8610
E-mail: gdouzas@novaims.unl.pt

F. Bacao
NOVA Information Management School
Campus de Campolide, 1070-312 Lisboa, Portugal
Tel.: +351 21 382 8610
E-mail: bacao@novaims.unl.pt

M. Lechleitner
NOVA Information Management School
Campus de Campolide, 1070-312 Lisboa, Portugal
Tel.: +351 21 382 8610
E-mail: mm.lechleitner@gmail.com

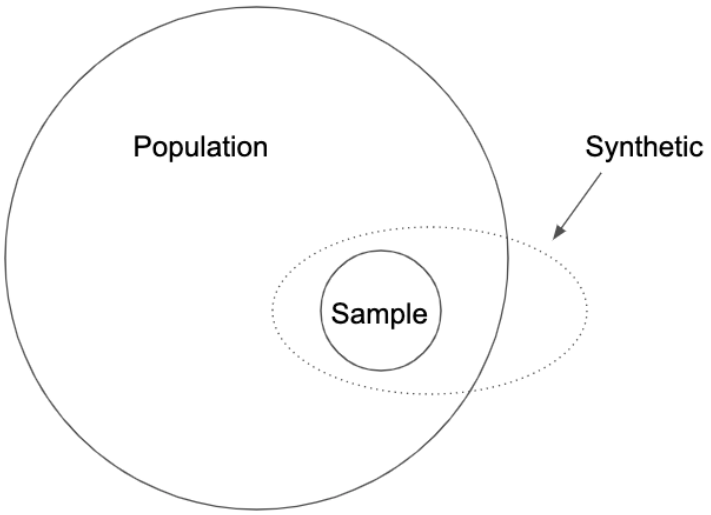

Fig. 1: Relationship between population, sample and synthetic data [21].

size of the whole dataset (small data problem) for any classification or regression task [30]. In both cases, small training samples affect the performance of machine learning models [32].

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 [33].

Under-representation of observations in the sample set can be solved in different ways. 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 [30]. 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 [21]. 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.

The next sections will describe an effective way to tackle the small data problem. Specifically, the focus in this paper is 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. On the other hand, regression tasks require an extensive modification of the data generation process and they will be a topic of future research.

In section 2, the previously studied solutions are reviewed, while a detailed description of the proposed method is presented in section 3. This is followed by the research methodology and the experimental results in sections 4 and 5. Finally, the conclusions of the paper are presented in section 6.

2 Related work

Several methods to increase the data size have been presented by the research community. In this section, the most important approaches to deal with the small data problem are presented. We start by describing fuzzy theories, which have historically been the most used approach. Next, we look at the resampling mechanism, which mainly consists of bootstrapping techniques, and finally, we review oversampling methods 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 [17]. 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 [34]. 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 [27]. The introduction of virtual examples expands the effective training set size and can therefore help to mitigate the learning problem. [27] showed that the process of creating artificial samples is mathematically equivalent to incorporating prior knowledge. The concept was applied on object recognition by 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) [19]. A number of synthetic samples was created, within a newly defined domain range. Although, the process was manually configured, its application dramatically improved the classification accuracy of a neural network.

[15] 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 [2] 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) [23] combines data trend estimation with a diffusion technique to estimate the domain range, thus avoiding overestimation. 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 was widely used as a synthetic sample generation method and was recognized as an effective way to deal with small datasets [17].

A drawback of MTD is that only considers the data attributes as independent 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 [22], [25].

2.2 Random OverSampling

An alternative approach to fuzzy theory as well the most well-known artificial sample generation method is the Bootstrapping Procedure [17] or Random OverSampling (ROS) as is known in the imbalanced learning research area. The main difference to the previously presented techniques is that ROS expands the training set by duplicating instances from the original dataset [8]. The selection is done with replacement, thus it allows the algorithms to use the same sample more than one time. However, ROS may cause overfitting when applied to small data because it repetitively uses the same information [31], [20]. Nevertheless, [16] applied ROS in batch process industries where it was shown that it may help mitigate the small data problem.

2.3 Informed oversampling

A different approach to fill information gaps is informed oversampling. It is a family of artificial data generation approaches, that create new instances and not copies of the existing ones like ROS, 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 bootstrapping methods presented above.

Although the small data and imbalanced learning problems have common characteristics, it seems that their proposed solutions had very few connections so far. In this paper, we provide such a connection by extending the application of the data generation process of the selected oversampling algorithms to the majority class.

There are several informed oversampling algorithms presented in the literature. The first method to be proposed and still the most popular is the Synthetic Minority Oversampling TEchnique (SMOTE) [1]. Numerous variants of SMOTE have been created, increasing its status as the staple idea in oversampling [10], with one of the most popular and effective variants being Borderline SMOTE (B-SMOTE) [12].

Contrary to the above SMOTE variants, the oversampling algorithm Geometric SMOTE (G-SMOTE) [6], not only modifies the selection process of existing instances but also substitutes the SMOTE data generation mechanism. The experimental study of [6] includes an extensive comparison between oversampling approaches using 69 imbalanced datasets and several classifiers. The results provide empirical evidence supporting the idea that G-SMOTE constitutes a significant improvement over previous oversampling methods.

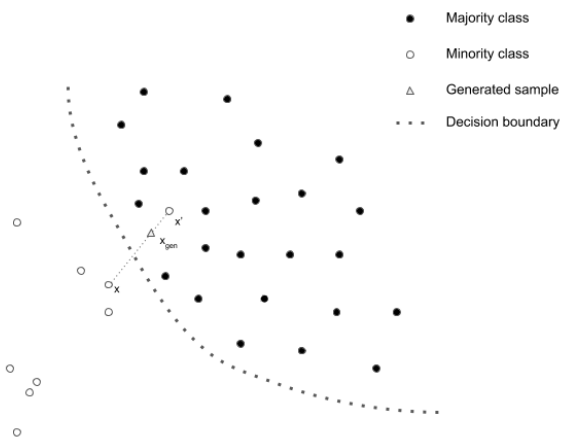

Fig. 2: Generation of a noisy example x_{gen} as a linear interpolation of x and x'.

2.3.1 SMOTE data generation mechanism

SMOTE algorithm is based on the concepts of k-nearest neighbors and the linear interpolation between existing samples 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. The algorithm is very popular due to its simplicity as well as its robustness. B-SMOTE retains the same data generation mechanism as SMOTE but modifies the selected minority class instances by generating samples closer to the decision boundary.

In the case of the small data problem, when SMOTE and B-SMOTE are used, the data generation process is trivially extended to include not only the minority classes but also the majority class. Using this approach, [20] showed that SMOTE is able to successfully fill the information gaps of small data with synthetic samples. However, given its limitations as described below, SMOTE did not achieve the best results within their study.

A drawback of the SMOTE data generation process is that in practice, the separation between majority and minority class areas is often not clearly definable. Consequently, noisy samples may be generated when a minority sample lies in the region of the majority classes. 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 overfitting. Figure 2 presents an example of the first case, i.e. noisy sample generation.

2.3.2 G-SMOTE data generation mechanism

G-SMOTE can be seen as a substitute of the SMOTE data generation mechanism, enhanced by geometric properties. Compared to SMOTE, it expands the data generation area and prevents the creation of noise. Specifically, 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 initially selected minority sample. Furthermore, G-SMOTE is designed to avoid the generation of noisy samples by introducing the *selection strategy* hyperparameter.

Figure 3 demonstrates the distribution of artificially created samples by SMOTE versus G-SMOTE. When the number of k-nearest neighbors is increased, SMOTE tends to generate noisy samples, whereas G-SMOTE avoids this scenario.

Similarly to SMOTE and its variants, the G-SMOTE data generation process can be applied to the small data problem, although it requires a non-trivial modification. The details are provided in section 3.

3 Proposed method

In this section, we present Geometric Small Data Oversampling Technique (GSDOT) as a novel data generation procedure suitable for the small data problem. While GSDOT uses the G-SMOTE data generation mechanism, to generate data, it does so for the entire dataset, independent from the class distribution. This way, GSDOT constitutes a new algorithm modified to generate data for all the classes in the dataset.

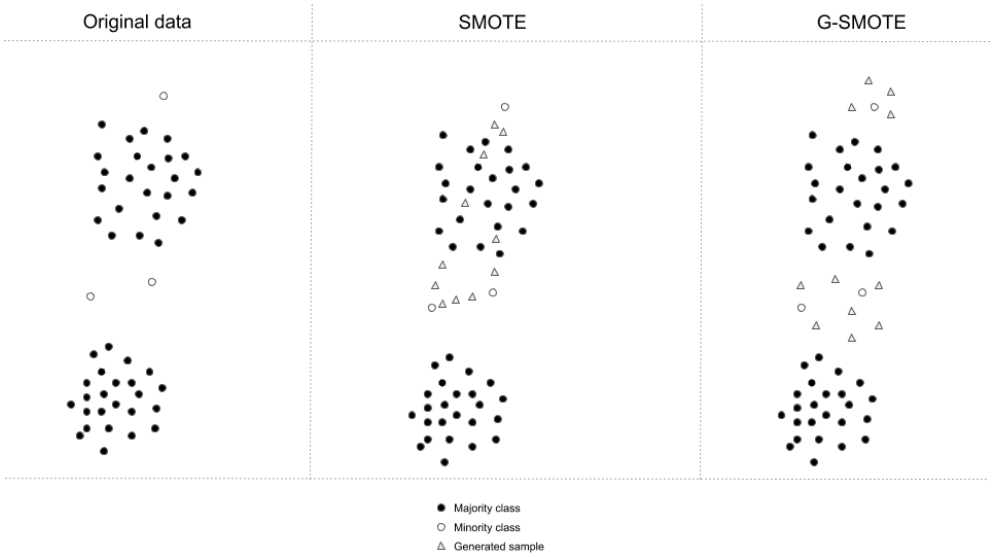

Fig. 3: Three positive class instances are used by SMOTE and G-SMOTE to generate synthetic data. Contrary to SMOTE, G-SMOTE generates non-noisy samples with high variety.

3.1 Data generation area

GSDOT algorithm randomly generates artificial data within a geometric 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 hyperparameters called *truncation factor* and *deformation factor*. Additionally, the *selection strategy* hyperparameter modifies the standard SMOTE selection process and also affects the size of the geometric region.

3.2 GSDOT algorithm

The inputs of the GSDOT algorithm are sets of 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 resampled dataset. The GSDOT 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 S_{pos} as 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 hyperspheroid 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 hyperparameters *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.

3.3 Considerations

As it is shown above, GSDOT algorithm applies independently the G-SMOTE data generation process for both the positive and negative classes. The above description of step 2, that constitutes the data generation mechanism, excludes mathematical formulas and details which can be found in [6]. Figure 4 shows an example of the GSDOT data generation process for the three different values of the *selection strategy* hyperparameter when positive class data generation is considered.

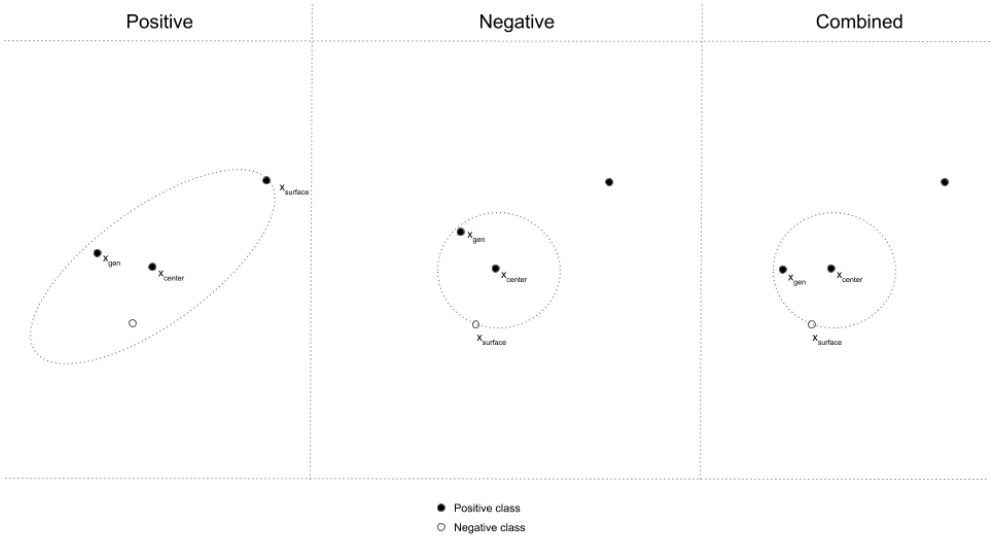

Fig. 4: The GSDOT data generation mechanism for the three different *selection strategy* values, when positive class samples are generated. The hyperparameters *deformation factor* and *truncation factor* have also different values resulting in the permissible data generation areas of the figure.

Table 1: Description of the datasets.

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

4 Research methodology

The main objective of this work is to compare GSDOT to other oversampling algorithms for the small data problem. Therefore, we use a variety of datasets, metrics and classifiers to evaluate the performance of oversamplers. A description of this set-up, the experimental procedure as well as the software implementation is provided in this section.

4.1 Experimental data

The ten datasets used to test the performance of GSDOT are retrieved from UCI Machine Learning Repository [7]. The focus on their selection lies on binary classification problems with a balanced distribution of the two classes. In order to assure generalizability of the results, the datasets are related to different topics such as health care, finance, business and physics. Details of the datasets are presented in table 1:

The approach to test whether oversamplers, and particularly GSDOT, are able to produce high quality artificial data, is to generate randomly undersampled versions of the above datasets and try to reconstruct them. Specifically, random sampling of 50%, 75%, 90% and 95% is applied on them, called undersampling ratio, followed by their enhancement with artificial data that are created from the various oversampling methods. The details of the process are presented in subsection 4.4.

4.2 Evaluation metrics

To evaluate the performance of GSDOT, the experiment includes two different metrics. The first choice is *Accuracy* as it is one of the most common metrics for the evaluation of classification models [14]. *Accuracy* measures the ratio of correct predictions over the total number of instances. The mathematical formula is the following:

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

where TP, TN, FP, FN denote the number of correctly classified positive, negative and misclassified negative, positive instances, respectively. *Accuracy* 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*:

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

4.3 Machine learning algorithms

Three oversampling algorithms are used in the experiment along with GSDOT. ROS is chosen for its simplicity, while SMOTE is selected for being the most widely used oversampler. B-SMOTE is also included since it represents the most popular modification of the original SMOTE algorithm. Finally, no oversampling is also applied as an additional baseline method. The oversampling algorithms are used to generate artificial data for both the positive and negative class.

For the evaluation of the oversampling methods, a variety of classifiers are included to ensure that the results are independent of their characteristics. Specifically, the experiment is conducted using the following four classifiers: Logistic Regression (LR) [26], K-Nearest Neighbors (KNN) [3], Decision Tree (DT) [29] and Gradient Boosting (GB) [11].

4.4 Experimental procedure

As explained above, the main goal of the paper is to evaluate how well GSDOT algorithm, as presented in subsection 3.2, compares to other oversampling methods, when small datasets are enhanced with artificial samples.

The performance of the classifiers is assessed using k-fold cross-validation scores with $k = 5$. Each dataset D is randomly splitted into k subsets (folds) $D_1, D_2, \cdots, D_k$ of approximately equal size. Each fold is used as a validation set and the remaining folds are used to train the model. The process is repeated in k stages, until each D_k is used as a validation set [13]. The experimental procedure for an arbritary dataset and cross-validation stage is described below:

1. The $k - 1$ folds are undersampled using an undersampling ratio of 50%, 75%, 90% and 95%, equal to the percentage of the dataset that is removed (1). Alternatively, no undersampling is applied and the original data are presented to the classifiers, a case identified as BENCHMARK (2).
2. Oversampling is applied to the undersampled data (3) of the previous step that increases their size and class distribution back to the initial (4). Alternatively, no oversampling is applied and the small data are presented to the classifiers, a case identified as NONE (5).
3. The resampled data of the previous step as well as the data from two special cases as described above are used to train the classifiers.
4. The classifiers are evaluated on the remaining fold of step 1.

Figure 5 visualizes the 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 across runs. The algorithms used in the experiment have various hyperparameters that yield different scores. The maximum of these scores is reported.

In order to confirm the statistical significance of the experimental results, the Friedman test as well as the Holm test [4] are applied. Ranking scores are assigned to each oversampling method, as well as the BENCHMARK and NONE cases, 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 GSDOT 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.

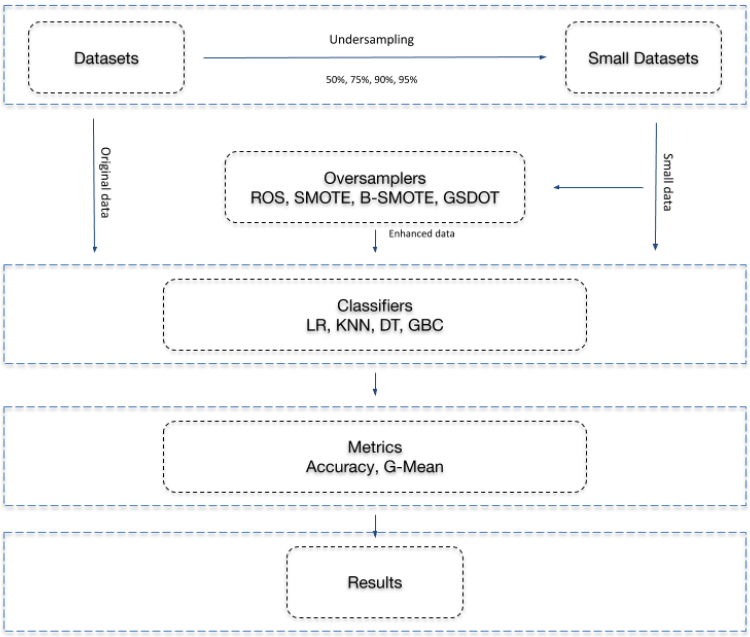

Fig. 5: Visualization of the experimental procedure.

4.5 Software Implementation

The implementation of the experimental procedure was based on the Python programming language, using the Scikit-Learn [28] and Imbalanced-Learn [18] libraries. All functions, algorithms, experiments and results reported are provided at the GitHub repository of the project. Additionally, the Research-Learn library provides a framework to implement comparative experiments, also being fully integrated with the Scikit-Learn ecosystem.

5 Results and discussion

In this section the performance of the different oversamplers and the results of the statistical tests are presented and analyzed.

5.1 Comparative presentation

The mean cross validation scores and the standard error across all datasets per classifier, metric and undersampling ratio (Ratio) are presented in Table 2. The Ratio is included in order to evaluate how the methods perform as the dataset size diminishes. As explained above, we also include the BENCHMARK method that represents the performance of the classfiers on the original dataset. The BENCHMARK method is expected to obtain the best results by design. The highest scores for each row, excluding the BENCHMARK scores, are highlighted.

Table 2: Results for mean cross validation scores of all methods.

Ratio	Classifier	Metric	NONE	RANDOM	SMOTE	B-SMOTE	GSDOT	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

Ratio	Classifier	Metric	NONE	RANDOM	SMOTE	B-SMOTE	GSDOT	BENCHMARK
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
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 shows that GSDOT outperforms all other methods, almost for all combinations of classifiers and metrics. Throughout the scores we can observe that all methods have a better performance as the dataset increase their size i.e. the Ratio gets smaller. Particularly, the scores of GSDOT 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 oversamplers.

Table 3 presents the mean and standard error of percentage difference between GSDOT and NONE. It shows that GSDOT performs significantly better compared to the case where no oversampling is applied for every combination of undersampling ratio, classifier and metric. Particularly, the performance gap increases for higher undersampling ratios.

Table 3: Results for percentage difference between GSDOT and NONE.

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
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

A ranking score in the range 1 to 5 is assigned to each oversampler as well as the two special case NONE and BENCHMARK. The mean ranking across the datasets of all methods is presented in table 4:

Table 4: Results for mean rankings of all methods.

Ratio	Classifier	Metric	NONE	RANDOM	SMOTE	B-SMOTE	GSDOT	BENCHMARK
50	LR	ACCURACY	4.64	4.64	3.07	5.14	**1.71**	1.79
50	LR	G-MEAN	5.14	4.57	**2.57**	4.14	2.71	1.86
50	KNN	ACCURACY	4.36	5.43	3.0	4.14	**2.14**	1.93
50	KNN	G-MEAN	4.71	5.0	3.0	4.0	**2.43**	1.86
50	DT	ACCURACY	4.43	4.57	3.79	4.71	**1.71**	1.79
50	DT	G-MEAN	4.79	4.64	3.36	4.64	**1.86**	1.71
50	GBC	ACCURACY	5.29	4.21	4.0	4.36	**1.79**	1.36
50	GBC	G-MEAN	5.21	4.5	3.93	4.21	**1.86**	1.29
75	LR	ACCURACY	4.0	4.64	3.86	5.36	**2.14**	1.0
75	LR	G-MEAN	4.43	4.86	3.71	4.57	**2.29**	1.14
75	KNN	ACCURACY	4.86	4.57	2.79	5.0	**2.21**	1.57
75	KNN	G-MEAN	5.43	4.57	2.57	4.57	**2.29**	1.57
75	DT	ACCURACY	4.14	4.29	4.14	5.0	**2.14**	1.29
75	DT	G-MEAN	4.43	4.0	4.14	4.86	**2.43**	1.14
75	GBC	ACCURACY	4.71	4.0	3.86	4.86	**2.43**	1.14
75	GBC	G-MEAN	4.86	4.14	4.0	4.43	**2.43**	1.14
90	LR	ACCURACY	4.21	4.29	3.64	5.43	**2.43**	1.0
90	LR	G-MEAN	5.14	4.29	3.86	4.43	**2.29**	1.0
90	KNN	ACCURACY	5.0	4.36	3.0	5.07	**2.57**	1.0
90	KNN	G-MEAN	5.43	4.57	2.57	5.0	**2.43**	1.0
90	DT	ACCURACY	4.21	4.36	4.21	5.21	**2.0**	1.0
90	DT	G-MEAN	4.5	4.07	4.36	4.93	**2.14**	1.0
90	GBC	ACCURACY	4.64	4.14	3.93	5.0	**2.29**	1.0
90	GBC	G-MEAN	4.43	4.14	4.14	5.0	**2.29**	1.0
95	LR	ACCURACY	4.29	4.71	3.29	5.14	**2.57**	1.0
95	LR	G-MEAN	4.64	4.79	3.29	4.43	**2.86**	1.0
95	KNN	ACCURACY	5.14	4.71	**2.57**	4.86	2.71	1.0
95	KNN	G-MEAN	5.57	4.29	3.0	4.29	**2.86**	1.0
95	DT	ACCURACY	5.36	4.29	3.93	4.43	**2.0**	1.0
95	DT	G-MEAN	5.14	4.29	3.86	4.43	**2.29**	1.0
95	GBC	ACCURACY	4.43	4.36	3.71	5.29	**2.21**	1.0
95	GBC	G-MEAN	4.5	4.5	3.64	4.86	**2.5**	1.0

The highest rankings for each row, excluding the BENCHMARK case, are highlighted. Looking at the table, GSDOT is ranked on the top place when comparing with NONE, ROS, SMOTE and B-SMOTE.

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 in table 5:

Table 5: Results for Friedman test.

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

Therefore, the null hypothesis of the Friedman test is rejected at a significance level of a = 0.05, i.e. the oversamplers 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 GSDOT algorithm as the control method. The results are shown in table 6:

Table 6: Adjusted p-values using Holm method.

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

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

Many domains and applications continue to be limited to the use of small datasets. The insufficient size of training data usually results in inferior performance of machine learning algorithms. This paper proposes an effective solution to mitigate the small data problem in classification tasks. As shown above, the GSDOT algorithm 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 the algorithm's capability of increasing the diversity of new instances while avoiding the generation of noisy samples. An important point is that GSDOT significantly improves classification performance compared to the case where only the small data are used, for every combination of undersampling ratio, classifier and metric as shown in table 2. Specifically, the full experimental results show that there is not a single instance where using the small data outperformed GSDOT. Table 3 also shows that the performance gap increases for higher undersampling ratios. This is a clear indication that, when using a small dataset, it is safe and appropriate to apply the the GSDOT algorithm, in order to generate artificial samples and improve the performance of classifiers. Also GSDOT outperforms standard oversampling approaches such as ROS and SMOTE, being closer to the BENCHMARK scores than any of them. As presented in table 2, in 30 out of 32 combinations of classifiers and metrics, GSDOT outperforms all other methods. Finally, the statistical analysis of the experiments, tables 5 and 6, confirms the dominance of the proposed algorithm. The GSDOT implementation is available as an open source project, so that the research community and data science practitioners can make use of it to improve the performance of machine learning algorithms.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research **16**, 321–357 (2002). DOI 10.1613/jair.953. URL https://doi.org/10.1613/jair.953
2. Chongfu, H.: Principle of information diffusion. Fuzzy Sets and Systems **91**(1), 69–90 (1997). DOI 10.1016/s0165-0114(96)00257-6. URL https://doi.org/10.1016/s0165-0114(96)00257-6
3. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory **13**(1), 21–27 (1967). DOI 10.1109/tit.1967.1053964. URL https://doi.org/10.1109/tit.1967.1053964
4. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006). URL http://dl.acm.org/citation.cfm?id=1248547.1248548
5. Domingos, P.: A few useful things to know about machine learning. Communications of the ACM **55**(10), 78 (2012). DOI 10.1145/2347736.2347755. URL https://doi.org/10.1145/2347736.2347755
6. Douzas, G., Bacao, F.: Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE. Information Sciences **501**, 118–135 (2019). DOI 10.1016/j.ins.2019.06.007. URL https://doi.org/10.1016/j.ins.2019.06.007
7. Dua, D., Graff, C.: UCI machine learning repository (2017). URL http://archive.ics.uci.edu/ml

8. Efron, B., Tibshirani, R.: An introduction to the bootstrap. No. 57 in Monographs on statistics and applied probability. Chapman & Hall, New York (1993)
9. European Commission, Directorate-General for Justice and Consumers: The GDPR: new opportunities, new obligations : what every business needs to know about the EU's General Data Protection Regulation. (2018). URL https://data.europa.eu/doi/10.2838/97649. OCLC: 1039888381
10. Fernandez, A., Garcia, S., Herrera, F., Chawla, N.V.: SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. Journal of Artificial Intelligence Research **61**, 863–905 (2018). DOI 10.1613/jair.1.11192. URL https://doi.org/10.1613/jair.1.11192
11. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. The Annals of Statistics **29**(5), 1189–1232 (2001). DOI 10.1214/aos/1013203451
12. Han, H., Wang, W.Y., Mao, B.H.: Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In: International Conference on Intelligent Computing, pp. 878–887. Springer, Berlin, Heidelberg (2005). DOI 10.1007/11538059_91. URL http://link.springer.com/10.1007/11538059{_}91
13. Han, J., Kamber, M.: Data mining: concepts and techniques, 3rd ed edn. Elsevier, Burlington, MA (2012)
14. Hossin, M., M.N, S.: A review on evaluation metrics for data classification evaluations. International Journal of Data Mining & Knowledge Management Process **5**(2), 01–11 (2015). DOI 10.5121/ijdkp.2015.5201. URL https://doi.org/10.5121/ijdkp.2015.5201
15. Huang, C., Moraga, C.: A diffusion-neural-network for learning from small samples. International Journal of Approximate Reasoning **35**(2), 137–161 (2004). DOI 10.1016/j.ijar.2003.06.001. URL https://doi.org/10.1016/j.ijar.2003.06.001
16. Ivănescu, V.C., Bertrand, J.W.M., Fransoo, J.C., Kleijnen, J.P.C.: 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 (2006). DOI 10.1057/palgrave.jors.2601966. URL https://doi.org/10.1057/palgrave.jors.2601966
17. Lateh, M.A., Muda, A.K., Yusof, Z.I.M., Muda, N.A., Azmi, M.S.: Handling a small dataset problem in prediction model by employ artificial data generation approach: A review. Journal of Physics: Conference Series **892**, 012016 (2017). DOI 10.1088/1742-6596/892/1/012016. URL https://doi.org/10.1088/1742-6596/892/1/012016
18. Lemaître, G., Nogueira, F., Aridas, C.K.: Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. Journal of Machine Learning Research **18**(17), 1–5 (2017). URL http://jmlr.org/papers/v18/16-365.html
19. Li, D.C., Chen, L.S., Lin, Y.S.: Using functional virtual population as assistance to learn scheduling knowledge in dynamic manufacturing environments. International Journal of Production Research **41**(17), 4011–4024 (2003). DOI 10.1080/0020754031000149211. URL https://doi.org/10.1080/0020754031000149211
20. Li, D.C., Lin, W.K., Chen, C.C., Chen, H.Y., Lin, L.S.: Rebuilding sample distributions for small dataset learning. Decision Support Systems **105**, 66–76 (2018). DOI 10.1016/j.dss.2017.10.013. URL https://doi.org/10.1016/j.dss.2017.10.013
21. Li, D.C., Lin, Y.S.: Using virtual sample generation to build up management knowledge in the early manufacturing stages. European Journal of Operational Research **175**(1), 413–434 (2006). DOI 10.1016/j.ejor.2005.05.005. URL https://doi.org/10.1016/j.ejor.2005.05.005
22. Li, D.C., Wen, I.H.: A genetic algorithm-based virtual sample generation technique to improve small data set learning. Neurocomputing **143**, 222–230 (2014). DOI 10.1016/j.neucom.2014.06.004. URL https://doi.org/10.1016/j.neucom.2014.06.004
23. Li, D.C., Wu, C.S., Tsai, T.I., Lina, Y.S.: 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 (2007). DOI 10.1016/j.cor.2005.05.019. URL https://doi.org/10.1016/j.cor.2005.05.019
24. Lin, L.S., Li, D.C., Chen, H.Y., Chiang, Y.C.: An attribute extending method to improve learning performance for small datasets. Neurocomputing **286**, 75–87 (2018). DOI 10.1016/j.neucom.2018.01.071. URL https://doi.org/10.1016/j.neucom.2018.01.071
25. Lin, Y.S., Li, D.C.: The generalized-trend-diffusion modeling algorithm for small data sets in the early stages of manufacturing systems. European Journal of Operational Research **207**(1), 121–130 (2010). DOI 10.1016/j.ejor.2010.03.026. URL https://doi.org/10.1016/j.ejor.2010.03.026
26. McCullagh, P., Nelder, J.: Generalized Linear Models. Routledge (2019). DOI 10.1201/9780203753736. URL https://doi.org/10.1201/9780203753736
27. Niyogi, P., Girosi, F., Poggio, T.: Incorporating prior information in machine learning by creating virtual examples. Proceedings of the IEEE **86**(11), 2196–2209 (1998). DOI 10.1109/5.726787. URL https://doi.org/10.1109/5.726787
28. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12** (2011)
29. Salzberg, S.L.: C4.5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. Machine Learning **16**(3), 235–240 (1994). DOI 10.1007/bf00993309. URL https://doi.org/10.1007/bf00993309
30. Sezer, E., Nefeslioglu, H., Gokceoglu, C.: An assessment on producing synthetic samples by fuzzy c-means for limited number of data in prediction models. Applied Soft Computing **24**, 126–134 (2014). DOI 10.1016/j.asoc.2014.06.056. URL https://doi.org/10.1016/j.asoc.2014.06.056
31. Tsai, C.H., Li, D.C. (eds.): Improving Knowledge Acquisition Capability of M5' Model Tree on Small Datasets. IEEE (2015)
32. Tsai, T.I., Li, D.C.: Utilize bootstrap in small data set learning for pilot run modeling of manufacturing systems. Expert Systems with Applications **35**(3), 1293–1300 (2008). DOI 10.1016/j.eswa.2007.08.043. URL https://doi.org/10.1016/j.eswa.2007.08.043
33. Vapnik, V.N.: The nature of statistical learning theory, 2nd ed edn. Statistics for engineering and information science. Springer, New York (2008)
34. Zimmermann, H.J.: Fuzzy set theory. Wiley Interdisciplinary Reviews: Computational Statistics **2**(3), 317–332 (2010). DOI 10.1002/wics.82. URL https://doi.org/10.1002/wics.82