Geometric SMOTE for Imbalanced Datasets with Nominal and Continuous Features

Joao Fonseca[1*], Fernando Bacao[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

The development of classifiers with imbalanced datasets can be dealt with in 3 different ways. One of these, artificial data generation, is a more general approach when opposed to algorithmic modifications or cost-sensitive solutions. Since the proposal of the Synthetic Minority Oversampling TEchnique (SMOTE), various SMOTE variants and neural network-based oversampling methods have been developed. However, the available options to oversample datasets with both nominal and continuous features are limited. In this paper, we propose Geometric SMOTE for Nominal and Continuous features (G-SMOTENC), consisting of a combination between G-SMOTE and SMOTENC. Our method uses SMOTENC's encoding and generation mechanism for nominal features, while using G-SMOTE's data selection mechanism to determine the center observation and k-nearest neighbors and generation mechanism for continuous features. The G-SMOTENC's performance is compared against SMOTENC's along with 2 other baseline methods, a State-of-the-art oversampling method and no oversampling. Our experimental results, performed over 20 datasets with varying imbalance ratios, number of metric and non-metric features and target classes, show a significant improvement in the quality of the generated data when using G-SMOTENC as the oversampling method. An open source implementation of G-SMOTENC is made available in the Python programming language.

1. Introduction

There are various Machine Learning (ML) tasks that deal with highly imbalanced datasets, such as fraud transactions detection, fault detection and medical diagnosis [1]. In these situations, predicting false positives is often a more acceptable error, since the class of interest is often the minority class [2]. However, standard ML classifiers induce a bias in favor of the classes with highest frequency and limits the predictive power on lower frequency classes [3, 4]. This effect is known, in the ML community, as Imbalanced Learning.

Imbalanced learning involves a dataset with two or more target classes with uneven class frequencies, where the minority class is defined as the class with the least amount of observations and the majority is the class with the highest amount of observations [5]. There are 3 main approaches to deal with imbalanced learning [6]: (1) Cost-sensitive solutions attribute a higher misclassification costs to minority class observations to minimize higher cost errors, (2) Algorithmic level solutions modify ML classifiers to improve the learning of the minority class and (3) Resampling solutions generate synthetic minority class observations and/or remove majority class observations to balance the training dataset.

Since it is an external approach to imbalanced learning, the later method becomes particularly useful. It dismisses the required domain knowledge to build cost matrix and the technical complexity/knowledge of applying an imbalanced learning-specific classifier. Resampling can be done via undersampling, oversampling or hybrid approaches [7]. In this paper, we will focus on oversampling approaches.

Currently, the presence of nominal features in imbalanced learning tasks limit the options available to deal with class imbalance. Even though it is possible to use encoding methods such as one-hot or ordinal encoding to convert nominal features into numerical, applying a distance metric on nominal data is questionable since the data is unordered [8]. In this case, one possible approach is to use models that are able to handle different scales (*e.g.*, Decision Tree). However, this assumption may be limitative since there are few ML algorithms where this condition is verified. Another possible approach is transforming the variables to meet scale assumptions [8]. This has been explored in the Synthetic Minority Oversampling Technique for Nominal and Continuous features (SMOTENC) [9] (explained in Section 2).

In the presence of datasets with mixed-data types, the usage of most well-known resampling algorithms becomes unfeasible. This happens because these methods consider exclusively continuous data; they have not been adapted to also nominal features as well. Specifically, since the proposal of SMOTE, various other SMOTE-variants have been developed to address some of its limitations, but little significant work has been developed to resample datasets with both nominal and continuous features.

In this paper, we propose Geometric SMOTE for Nominal and Continuous features (G-SMOTENC). It generates the continuous feature values of a synthetic observation within a truncated hyper-spheroid its nominal feature values using the most common value of its nearest neighbors. In addition, G-SMOTENC uses G-SMOTE's data selection strategy, along with SMOTENC's approach to find the center observation's nearest neighbors. G-SMOTENC is a generalization of both SMOTENC and G-SMOTE [10]. Specifically, with the correct hyperparameters, G-SMOTENC can mimic the behavior of SMOTE, SMOTENC or G-SMOTE. It is implemented in the open source Python library "ML-Research" and is fully compatible with Scikit-Learn ecosystem.

The rest of this paper is structured as follows: Section 2 describes the related work and its limitations, Section 3 describes the proposed method (G-SMOTENC), Section 4 lays out the methodology used to test G-SMOTENC, Section 5 shows and discusses the results obtained in the experiment and Section 6 presents the conclusions drawn from this study.

2. Related Work

A classification problem contains n classes, having C_{maj} as the set of majority class observations (*i.e.*, observations belonging to the most common target class) and C_{min} as the set of minority class observations (*i.e.*, observations belonging to the least common target class). Typically, an oversampling algorithm will generate synthetic data in order to ensure $|C'_{min}| = |C_{maj}| = |C_i|, i \in \{1, \dots, n\}$.

Since the proposal of SMOTE, several other methods were built upon SMOTE to improve the quality of the data generated. The process of generating synthetic data using SMOTE-based algorithms can be divided into two distinct phases [11]:

1. Data selection. A synthetic observation, x^{gen}, is generated based on two existing observations. A SMOTE-based algorithm employs a given heuristic to select a non-majority class observation as the center observation, x^c, and one of its nearest neighbors, x^{nn}, selected randomly. For the case of SMOTE, x^c is randomly selected from each non-majority class.

2. Data generation. Once x^c and x^{nn} have been selected, x^{gen} is generated based on a transformation between the two selected observations. In the case of SMOTE, this transformation is a linear interpolation between the two obervations: $x^{gen} = \alpha x^c + (1-\alpha)x^{nn}, \alpha \sim \mathcal{U}(0,1)$.

Modifications to the SMOTE algorithm can be distinguished according to the phase where the modifications were applied. This distinction is especially relevant for the case of oversampling on datasets with mixed data types, since it raises the challenge of computing meaningful distances and k-nearest neighbors among observations. For example, State-of-the-art oversampling methods, such as Borderline-SMOTE [12], ADASYN [13], K-means SMOTE [14] and LR-SMOTE [15] modify the data selection mechanism and show promising results in imbalanced learning [16]. However, all of these algorithms select x^c using procedures that include calculating each observation's k-nearest neighbors or clustering methods, none of which is prepared to handle categorical data.

Modifications to SMOTE's generation mechanism are less common. A few oversampling methods, such as Safe-level SMOTE [17] and Geometric-SMOTE [10] achieve such modifications and have shown promising results [18]. However, these methods is also unable to handle datasets with categorical data. This limitation is especially true for methods combining modifications in the selection and generation mechanisms, as is the case of the Geometric Self-Organizing Maps Oversampling algorithm [19]. Other methods attempt to replace the SMOTE data generation mechanism altogether using different Generative Adversarial Networks (GAN) architectures [20, 21, 22]. However, these models are not only computationally expensive to train but also sensitive to the training initialization, ensuring a balanced training of the two networks involved is difficult, and tuning their hyperparameters is often challenging or unfeasible [23].

As discussed in Section 1, research on resampling methods with mixed data types is scarce. The original paper proposing SMOTE also proposed SMOTE for Nominal and Continuous (SMOTENC), an adaptation of SMOTE handle datasets with nominal and continuous features [9]. To determine the k-nearest neighbors of x^c, the distance is calculated by incorporating into the Euclidean distance the median of the standard deviations of the continuous features for every categorical feature with different values. Once x^c and x^{nn} have been determined, the values of the continuous features in x^{gen} are generated using the SMOTE generation mechanism, while the categorical features are given the most common values occurring in the k-nearest neighbors.

Alternatively to SMOTE-based methods, some non-informed over and undersampling methods may also be used for datasets with nominal and continuous features, specifically Random Oversampling (ROS) and Random Undersampling (RUS). These methods consist in randomly duplicating minority class observations (in the case of ROS), which can lead to overfitting [24, 25], or randomly removing majority class observations (in the case of RUS), which may lead to underfitting [26].

Only two oversampling algorithms capable of handling nominal and continuous features were found. Recently, a new SMOTE-based oversampling method for datasets with mixed data types, SMOTE-ENC [27], was proposed. This method modifies the encoding mechanism for categorical features used in the SMOTENC algorithm to account for categorical features' change of association with minority classes. The Multivariate Normal Distribution-based Oversampling for Numerical and Categorical features (MNDO-NC) [28] uses the original MNDO method [29] along with the SMOTENC encoding mechanism to find the values of the categorical features for the synthetic observation. However, the results reported in the paper showed that MNDO-NC was consistently outperformed by SMOTENC, which led us to discard this approach from further consideration.

3. Proposed Method

We propose G-SMOTENC to handle both nominal and continuous features. This an extension of the original G-SMOTE oversampler, which affects both its selection and generation mechanisms. Due to the novelty of the work, these modifications are based on the SMOTENC mechanism. However, this method can be extended with further modifications to the categorical data encoding and selection mechanisms in future work.

Similarly to G-SMOTE being an extension of SMOTE, G-SMOTENC is also an extension of SMOTENC since any method or ML pipeline using the SMOTENC generation mechanism can replace it by G-SMOTENC without any further modifications. The proposed method is described in pseudocode in Algorithm 1. The functions $SelectionMechanism$ and $GenerationMechanism$ are described in Algorithms 2 and 3, respectively.

Algorithm 1: G-SMOTENC.

Given: Dataset with binary target classes C_{min} and C_{maj}
Input: $C_{maj}, C_{min}, \alpha_{sel}, \alpha_{trunc}, \alpha_{def}$
Output: C^{gen}
begin
    $N \leftarrow |C_{maj}| - |C_{min}|$
    $C^{gen} \leftarrow \emptyset$
    while $|C^{gen}| < N$ **do**
        $x^c, x^{nn}, X^{nn} \leftarrow SelectionMechanism(C_{maj}, C_{min}, \alpha_{sel})$
        $x^{gen} \leftarrow GenerationMechanism(x^c, x^{nn}, X^{nn}, \alpha_{trunc}, \alpha_{def})$
        $C^{gen} \leftarrow C^{gen} \cup \{x^{gen}\}$

3.1. Selection Mechanism

The data selection mechanism is preceded by the numerical encoding of the categorical features. It mixes the selection mechanisms of SMOTENC and G-SMOTENC, as shown in Algorithm 2.

The selection mechanism uses the minority, majority and combined mechanisms (introduced by G-SMOTE). However, the nominal features in the minority and majority class observations, C_{maj} and C_{min} are first encoded using a one-hot encoding approach and replacing the constant 1 with the median of the standard deviations of the continuous features in C_{min} divided by 2. The nearest-neighbors (X^{nn}) of x^c are determined based on α_{sel}, which are passed on to the generation mechanism to determine the nominal features' values of x^{gen} in the generation mechanism. Simultaneously, x^{nn} is randomly selected from X^{nn} and will be used to generate x^{gen}'s continuous features' values.

Algorithm 2: G-SMOTENC's selection mechanism.

Input: $C_{maj}, C_{min}, \alpha_{sel}$
Output: x^c, x^{nn}, X^{nn}
Function *CatEncoder*(C_{maj}, C_{min}):
    $S \leftarrow$ Standard deviations of the continuous features in C_{min}
    $\sigma_{med} \leftarrow median(S)$
    forall $i \in \{maj, min\}$ **do**
        forall $f \in C_i^T$ **do**
            if f *is categorical* **then**
                $f' \leftarrow OneHotEncode(f) \times \sigma_{med}/2$
                $C_i' \leftarrow (C_i^T \setminus f)^T$
                $C_i' \leftarrow (C_i'^T \cup f')^T$
    return C'_{maj}, C'_{min}
Function *Surface*(α_{sel}, x^c, C_{maj}, C_{min}):
    if $\alpha_{sel} = minority$ **then**
        $x^{nn} \in C_{min,k}$ `// One of the` k`-nearest neighbors of` x^c `from` C_{min}
        $X^{nn} \leftarrow C_{min,k}$
    if $\alpha_{sel} = majority$ **then**
        $x^{nn} \in C_{maj,1}$ `// Nearest neighbor of` x^c `from` C_{min}
        $X^{nn} \leftarrow C_{maj,1}$
    if $\alpha_{sel} = combined$ **then**
        $x^{nn}_{min} \in C_{min,k}$
        $x^{nn}_{maj} \in C_{maj,1}$
        $x^{nn} \leftarrow argmin(||x^{nn}_{min} - x^c||, ||x^{nn}_{maj} - x^c||)$
        $X^{nn} \leftarrow C_{min,k} \cup C_{maj,1}$
    return x^{nn}, X^{nn} `//` X^{nn} `is the set of` k`-nearest neighbors`
begin
    $C'_{maj}, C'_{min} \leftarrow CatEncoder(C_{maj}, C_{min})$
    $x^c \in C'_{min}$ `// Randomly select` x^c `from` C'_{min}
    $x^{nn}, X^{nn} \leftarrow Surface(\alpha_{sel}, x^c, C'_{maj}, C'_{min})$
    Reverse encoding of nominal features in x^c, x^{nn} and X^{nn}

3.2. Generation Mechanism

G-SMOTENC's generation mechanism is shown in Algorithm 3. It divides the generation of x^{gen} into two parts: (1) generation of continuous feature values and (2) generation of nominal feature values. At this stage, the nominal features from x^c and x^{nn} are discarded. Afterwards, the continuous features are generated using G-SMOTE's generation mechanism; within a hyper-spheroid formed using the truncation and deformation hyperparameters (α_{trunc} and α_{def}, respectively). Finally, the nominal feature values are generated by the mode of each feature within the observations in X^{nn}.

G-SMOTENC contains 3 hyperparameters: the selection strategy (α_{sel}), the truncation factor (α_{trunc}) and the deformation factor (α_{def}). Figure 1 depicts the effect of those hyperparameters in the data selection and generation phases. For an in-depth definition of the hyperparameters mentioned, the reader may follow reference [10].

Algorithm 3: G-SMOTENC's generation mechanism.

Input: $x^c, x^{nn}, X^{nn}, \alpha_{trunc}, \alpha_{def}$
Output: x^{gen}
Function *Hyperball()*:
    $v_i \sim \mathcal{N}(0,1)$
    $r \sim \mathcal{U}(0,1)$
    $x^{gen} \leftarrow r^{1/p} \frac{(v_1,\ldots,v_p)}{||(v_1,\ldots,v_p)||}$
    return x^{gen}
Function *Vectors(x^c, x^{nn}, x^{gen})*:
    $e^{//} \leftarrow \frac{x^{nn}-x^c}{||x^{nn}-x^c||}$
    $x^{//} \leftarrow (x^{gen} \cdot e^{//})e^{//}$
    $x^{\perp} \leftarrow x^{gen} - x^{//}$
    return $x^{//}, x^{\perp}$
Function *Truncate($x^c, x^{nn}, x^{gen}, x^{//}, \alpha_{trunc}$)*:
    if $|\alpha_{trunc} - x^{//}| > 1$ **then**
        $x^{gen} \leftarrow x^{gen} - 2x^{//}$
    return x^{gen}
Function *Deform($x^{gen}, x^{\perp}, \alpha_{def}$)*:
    return $x^{gen} - \alpha_{def} x^{\perp}$
Function *Translate(x^c, x^{gen}, R)*:
    return $x^c + Rx^{gen}$
Function *GenNominal(X^{nn})*:
    $x^{gen}_{nom} = \emptyset$
    forall $f \in (X^{nn})^T$ **do**
        if *f is categorical* **then**
            $x^{gen}_{nom} \cup \{mode(f)\}$ `// Ties are decided with random selection`
    return x^{gen}_{nom}
begin
    Discard nominal features from x^c and x^{nn}
    $x^{gen} \leftarrow Hyperball()$
    $x^{//}, x^{\perp} \leftarrow Vectors(x^c, x^{nn}, x^{gen})$
    $x^{gen} \leftarrow Truncate(x^c, x^{nn}, x^{gen}, x^{//}, \alpha_{trunc})$
    $x^{gen} \leftarrow Deform(x^{gen}, x^{\perp}, \alpha_{def})$
    $x^{gen} \leftarrow Translate(x^c, x^{gen}, ||x^{nn}_{cont} - x^c||)$
    $x^{gen}_{nom} \leftarrow GenNominal(X^{nn})$
    $x^{gen} \leftarrow x^{gen} \cup x^{gen}_{nom}$

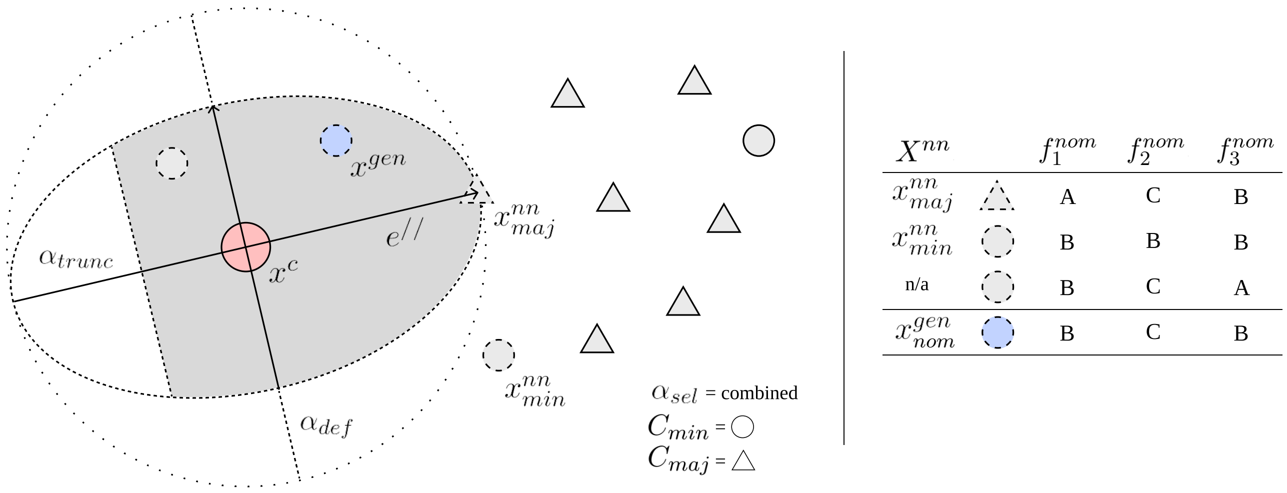

Figure 1: A visual depiction of G-SMOTENC. In this example, α_{trunc} is approximately 0.5 and α_{def} is approximately 0.4.

4. Methodology

This section describes how the evaluation of G-SMOTENC was performed. We describe the datasets used in the experiment, their source and preprocessing steps carried out in Section 4.1. We describe the resampling and classifications methods used for comparing the performance of G-SMOTENC with other relevant oversampling and undersampling mthods in Section 4.2. The performance metrics used are defined in Section 4.3. Finally, the experimental procedure is described in Section 4.4.

4.1. Experimental Data

The datasets used in this experiment were extracted from the UC Irvine Machine Learning Repository. All of the datasets are publicly available and cover a range of different domains. The selection of datasets was done to ensure that all datasets are imbalanced and contained non-metric features (*i.e.*, whether ordinal, nominal or binary). These datasets will be used to show how the performance of different classifiers varies according to the used over/undersampling method.

At an initial stage, all datasets were preprocessed manually with minimal manipulations, to avoid the application of preprocessing methods beyond the scope of this paper. This step was conducted to remove features and/or observations with missing values and identifying the non-metric features. The second stage of our preprocessing was done systematically. The resulting datasets are shown in Table 1.

Table 1: Description of the datasets collected after data preprocessing. The sampling strategy is similar across datasets. Legend: (IR) Imbalance Ratio

Dataset	Metric	Non-Metric	Obs.	Min. Obs.	Maj. Obs.	IR	Classes
Abalone	1	7	4139	15	689	45.93	18
Adult	8	6	5000	1268	3732	2.94	2
Adult (10)	8	6	5000	451	4549	10.09	2
Annealing	4	6	790	34	608	17.88	4
Census	24	7	5000	337	4663	13.84	2
Contraceptive	4	5	1473	333	629	1.89	3
Contraceptive (10)	4	5	1036	62	629	10.15	3
Contraceptive (20)	4	5	990	31	629	20.29	3
Contraceptive (31)	4	5	973	20	629	31.45	3
Contraceptive (41)	4	5	966	15	629	41.93	3
Covertype	2	10	5000	20	2449	122.45	7
Credit Approval	9	6	653	296	357	1.21	2
German Credit	13	7	1000	300	700	2.33	2
German Credit (10)	13	7	770	70	700	10.00	2
German Credit (20)	13	7	735	35	700	20.00	2
German Credit (30)	13	7	723	23	700	30.43	2
German Credit (41)	13	7	717	17	700	41.18	2
Heart Disease	5	5	740	22	357	16.23	5
Heart Disease (21)	5	5	735	17	357	21.00	5

The second part of the data preprocessing pipeline starts with the generation of artificially imbalanced datasets with different Imbalance Ratios ($IR = \frac{|C_{maj}|}{|C_{min}|}$). For each original dataset, we create its more imbalanced versions at intervals of 10, while ensuring that $|C_{min}| \geq 15$. The sampling strategy was determined for class $n \in \{1, \dots, n, \dots, m\}$ as a linear interpolation using $|C_{maj}|$ and $|C'_{min}| = \frac{|C_{maj}|}{IR_{new}}$, as shown in equation 1.

$$|C_i|^{imb} = \min(\frac{|C'_{min}| - |C_{maj}|}{n-1}.|C_i| + |C_{max}|, |C_i|) \tag{1}$$

The new, artificially imbalanced dataset, is formed by sampling observations without replacement from each C_i such that $C'_i \subseteq C_i, |C'_i| = |C_i|^{imb}$. The artificially imbalanced datasets are marked with its imbalance ratio as a suffix in Table 1.

The datasets (both original and artificially imbalanced versions) are then filtered to ensure all datasets have a minimum of 500 observations. The remaining datasets whose number of observations is larger than 5000 are randomly sampled to match this number of observations. Afterwards, for each remaining dataset we remove all observations from target classes whose frequency is lower than 15 observations. Finally, the continuous and discrete features are scaled to the range $[0, 1]$ to ensure a common range between all features.

4.2. Machine Learning Algorithms

The choice of classifiers used in the experimental procedure were based on their type (tree-based, nearest neighbors-based, linear model and ensemble-based), popularity and consistency in performance. We used Decision Tree (DT), a K-Nearest Neighbors (KNN) classifier, a Logistic Regression (LR) and a Random Forest (RF).

Given the lack of existing oversamplers that address imbalanced learning problems with mixed data types, the amount of benchmark methods used is also limited. We used the well known methods and one state-of-the-art oversampling method that are compatible with this type of datasets: SMOTENC, RUS, ROS and SMOTE-ENC. Table 2 shows the hyperparameters used for the parameter search described in Section 4.4.

Table 2: Hyperparameter definition for the classifiers and resamplers used in the experiment.

Classifier	Hyperparameter	Values
DT	min. samples split	2
	criterion	gini
	max depth	3, 6
LR	maximum iterations	10000
	multi-class	One-vs-All
	solver	saga
	penalty	None, L1, L2
KNN	# neighbors	3, 5
	weights	uniform
	metric	euclidean
RF	min. samples split	2
	# estimators	50, 100
	Max depth	3, 6
	criterion	gini
Resampler		
SMOTENC	# neighbors	3, 5
SMOTE-ENC	# neighbors	3, 5
G-SMOTENC	# neighbors	3, 5
	deformation factor	0.0, 0.25, 0.5, 0.75, 1.0
	truncation factor	-1.0, -0.5, 0.0, 0.5, 1.0
	selection strategy	"combined", "minority", "majority"
RUS	replacement	False
ROS	(no applicable parameters)	

4.3. Performance Metrics

The choice of the performance metric plays a critical role in the assessment of effect on classification tasks. The typical performance metrics, *e.g.*, Overall Accuracy (OA), are intuitive to interpret but are often inappropriate to measure a classifier's performance in an imbalanced learning context [30]. For example, to estimate an event that occurs in 1% of the dataset, a constant classifier would obtain an OA of 0.99 and still be unusable. However, this metric is still reported in some of our results to maintain a metric that is easier to interpret.

More recent surveys have found the Geometric-mean ($G\text{-}mean = \sqrt{\overline{Sensitivity} \times \overline{Specificity}}$), F1-score

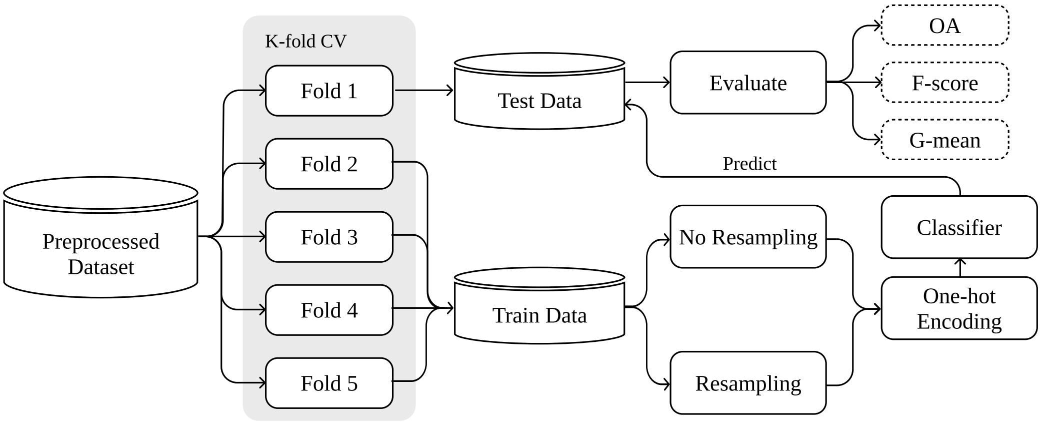

Figure 2: Experimental procedure used in this study.

($F\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$), $Sensitivity = \frac{TP}{FN+TP}$ and $Specificity = \frac{TN}{TN+FP}$ to be appropriate and common performance metrics in imbalanced learning contexts [31]. These metrics are calculated as a function of the number of False/True Positives (FP and TP) and False/True Negatives (FN and TN), having $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$. This finding is consistent with other well-known recommendations on the usage of performance metrics [32, 33]. This led us to adopt, along with OA, both F-score and G-mean as the main performance metrics for this study.

4.4. Experimental Procedure

The experimental procedure was applied similarly to all combinations of resamplers, classifiers and hyperparameter combinations across all datasets. The evaluation of the models' performance was tested using a 5-fold Cross Validation (CV) approach. The mean performance in the test set is calculated over the 5 folds and 3 different runs of the experimental procedure for each combination resampling/classifier hyperparameters. For each dataset, results of the hyperparameters that optimize the performance of a resampler/classifier are selected. These results were then used for analysis and are shown in Table 7 (see Appendix). Figure 2 shows a diagram of the experimental procedure described.

A CV run consists of a stratified partitioning (*i.e.*, each partition contains the same relative frequencies of target labels) of the dataset into five parts. A given resampler/classifier combination with a specific set of hyperparameters is fit and tested five times, using one of the partitions as a test set and the remaining ones as training set. In the ML pipeline defined for each run, the nominal features are one-hot encoded after oversampling and before passing the data to the classifier. The estimated performance consists of the average classification performance across the five different test sets.

4.5. Software Implementation

The algorithmic implementation of G-SMOTENC was written using the Python programming language and is available in the open-source package ML-Research [34], along with other utilities used to produce the experiment and outputs used in Section 5. In addition, the packages Scikit-Learn [35], Imbalanced-Learn [36] and Research-Learn were also used in the experimental procedure to get the implementations of the classifiers, benchmark over/undersamplers and run the experimental procedure. The original SMOTE-ENC implementation was retrieved from the authors' GitHub repository. The Latex code, Python scripts (including data pulling and preprocessing, experiment setup and results' analysis), as well as the datasets used are available in this GitHub repository.

5. Results and Discussion

In this section we present the experimental results. We focus on the comparison of classification performance using oversamplers whose generation mechanism is compatible with datasets containing both continuous and categorical features. The analysis of our experimental results were developed in two stages: (1) analysis of mean ranking and absolute performance and (2) statistical analysis. In Section 5.3 we discuss the main insights extracted by analysing the results reported in Sections 5.1 and 5.2.

5.1. Results

Table 3 presents the mean rankings of cross validation scores across the different combinations of oversamplers, metrics and classifiers. These results were calculated by assigning a ranking score for each oversampler from 1 (best) to 4 (worst) for each dataset, metric and classifier, based on the results reported in Table 7 (see Appendix).

Table 3: Mean rankings over the different datasets, folds and runs used in the experiment.

Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-ENC
DT	OA	1.66 ± 0.13	**1.61 ± 0.27**	3.58 ± 0.20	4.68 ± 0.15	5.42 ± 0.27	4.05 ± 0.23
DT	F-Score	**1.32 ± 0.11**	3.84 ± 0.40	3.13 ± 0.20	4.32 ± 0.19	5.47 ± 0.23	2.92 ± 0.34
DT	G-Mean	**1.68 ± 0.24**	5.84 ± 0.09	2.82 ± 0.21	2.95 ± 0.32	4.26 ± 0.32	3.45 ± 0.30
KNN	OA	2.50 ± 0.17	**1.37 ± 0.28**	4.21 ± 0.25	3.34 ± 0.35	5.68 ± 0.22	3.89 ± 0.15
KNN	F-Score	**1.37 ± 0.16**	3.95 ± 0.35	3.11 ± 0.29	3.47 ± 0.36	5.53 ± 0.23	3.58 ± 0.23
KNN	G-Mean	**1.74 ± 0.17**	5.84 ± 0.12	2.89 ± 0.23	3.76 ± 0.33	3.00 ± 0.45	3.76 ± 0.23
LR	OA	2.74 ± 0.19	**1.37 ± 0.28**	3.08 ± 0.21	4.34 ± 0.30	5.74 ± 0.17	3.74 ± 0.28
LR	F-Score	**2.11 ± 0.24**	4.53 ± 0.35	2.37 ± 0.28	3.47 ± 0.32	5.21 ± 0.27	3.32 ± 0.38
LR	G-Mean	2.13 ± 0.26	6.00 ± 0.00	3.61 ± 0.21	**2.11 ± 0.23**	3.32 ± 0.40	3.84 ± 0.28
RF	OA	1.82 ± 0.11	**1.24 ± 0.09**	3.97 ± 0.16	4.32 ± 0.21	5.92 ± 0.06	3.74 ± 0.22
RF	F-Score	**1.32 ± 0.13**	5.05 ± 0.31	3.16 ± 0.22	3.05 ± 0.31	5.37 ± 0.14	3.05 ± 0.27
RF	G-Mean	**1.68 ± 0.22**	5.79 ± 0.21	3.26 ± 0.28	2.47 ± 0.30	3.89 ± 0.35	3.89 ± 0.19

Table 4 presents the mean cross validation scores. With exception to the OA metric, G-SMOTENC either outperformed or matched the the remaining oversamplers.

Table 4: Mean scores over the different datasets, folds and runs used in the experiment

Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-ENC
DT	OA	0.74 ± 0.05	**0.75 ± 0.04**	0.68 ± 0.04	0.66 ± 0.04	0.58 ± 0.04	0.65 ± 0.04
DT	F-Score	**0.56 ± 0.04**	0.52 ± 0.04	0.54 ± 0.04	0.52 ± 0.04	0.48 ± 0.04	0.51 ± 0.04
DT	G-Mean	**0.69 ± 0.03**	0.60 ± 0.02	0.68 ± 0.03	0.67 ± 0.03	0.65 ± 0.03	0.66 ± 0.03
KNN	OA	0.69 ± 0.04	**0.73 ± 0.05**	0.67 ± 0.04	0.69 ± 0.05	0.57 ± 0.04	0.68 ± 0.05
KNN	F-Score	**0.53 ± 0.04**	0.50 ± 0.04	0.52 ± 0.04	0.52 ± 0.04	0.46 ± 0.04	0.51 ± 0.04
KNN	G-Mean	**0.66 ± 0.03**	0.58 ± 0.03	0.64 ± 0.03	0.62 ± 0.03	0.65 ± 0.03	0.63 ± 0.03
LR	OA	0.68 ± 0.05	**0.75 ± 0.04**	0.68 ± 0.05	0.66 ± 0.05	0.58 ± 0.04	0.67 ± 0.04
LR	F-Score	**0.54 ± 0.04**	0.52 ± 0.04	**0.54 ± 0.04**	0.53 ± 0.04	0.48 ± 0.04	0.52 ± 0.04
LR	G-Mean	**0.69 ± 0.02**	0.60 ± 0.03	0.68 ± 0.02	**0.69 ± 0.03**	0.67 ± 0.03	0.67 ± 0.03
RF	OA	0.74 ± 0.04	**0.76 ± 0.04**	0.69 ± 0.04	0.69 ± 0.04	0.59 ± 0.04	0.68 ± 0.05
RF	F-Score	**0.57 ± 0.04**	0.48 ± 0.04	0.55 ± 0.04	0.55 ± 0.04	0.49 ± 0.04	0.53 ± 0.04
RF	G-Mean	**0.70 ± 0.02**	0.57 ± 0.02	0.68 ± 0.03	0.69 ± 0.03	0.68 ± 0.03	0.68 ± 0.02

5.2. Statistical Analysis

To conduct an appropriate statistical analysis in an experiment with multiple datasets, it is necessary to use methods that account for the multiple comparison problem. Based on the recommendations found in [37], we applied the Friedman test along with the Holm-Bonferroni test for a post-hoc analysis.

In Section 4.3 we explained that OA, although easily interpretable, is not an appropriate performance metric for imbalanced learning problems. Therefore, the statistical analysis was developed using the two imbalance-appropriate metrics used in the study: F-Score and G-Mean. The statistical analysis started with the assessment of a statistically significant difference in performance across resampling methods using a Friedman test [38]. The results of this test are shown in Table 5. The null hypothesis is rejected in all cases.

Table 5: Results for Friedman test. Statistical significance is tested at a level of $\alpha = 0.05$. The null hypothesis is that there is no difference in the classification outcome across resamplers.

Classifier	Metric	p-value	Significance
DT	F-Score	2.2e-10	True
DT	G-Mean	1.2e-10	True
KNN	F-Score	2.3e-09	True
KNN	G-Mean	9.4e-10	True
LR	F-Score	2.1e-07	True
LR	G-Mean	9.7e-11	True
RF	F-Score	8.5e-12	True
RF	G-Mean	2.0e-10	True

We performed a Holm-Bonferroni test to understand whether the difference in performance of G-SMOTENC is statistically significant to the remaining resampling methods. The results of this test are shown in Table 6. The null hypothesis was rejected in 33 out of 40 tests.

Table 6: Adjusted p-values using the Holm-Bonferroni test. Statistical significance is tested at a level of $\alpha = 0.05$. The null hypothesis is that the benchmark methods perform similarly compared to the control method (G-SMOTENC).

Classifier	Metric	MySMOTENC	NONE	RAND-OVER	RAND-UNDER	SMOTENC
DT	F-Score	1.0e-01	**1.5e-04**	**7.3e-06**	**1.2e-06**	**1.5e-04**
DT	G-Mean	**2.3e-02**	**5.6e-07**	**2.8e-02**	**3.9e-04**	**2.7e-03**
KNN	F-Score	**5.9e-06**	**6.4e-04**	**7.2e-04**	**6.4e-04**	**2.2e-04**
KNN	G-Mean	**3.5e-03**	**1.6e-05**	**6.5e-03**	2.0e-01	**9.6e-03**
LR	F-Score	5.6e-02	**4.0e-03**	**9.2e-03**	**3.6e-04**	6.1e-01
LR	G-Mean	**4.7e-03**	**1.6e-07**	8.6e-01	2.4e-01	**4.0e-04**
RF	F-Score	**8.0e-03**	**1.7e-06**	**8.0e-03**	**1.7e-06**	**2.4e-04**
RF	G-Mean	**1.7e-03**	**3.8e-06**	2.5e-01	**2.3e-02**	**8.8e-03**

5.3. Discussion

The results reported in Section 5.1 show that G-SMOTENC consistently outperforms the remaining well-known oversampling approaches. Considering the results for the two imbalanced learning appropriate metrics in Table 3, G-Mean and F-Score, G-SMOTENC was only (on average) outperformed once by a neglectible margin. Unlike the results reported in [27], SMOTE-ENC's performance was rarely superior to SMOTENC's.

The relative difference in the classifiers' performance is better visible in Table 4. Using a RF classifier, for example, the impact of using G-SMOTENC compared to no oversampling improves, on average, 13 percentual points on G-mean and 9 percentual points using F-Score.

The difference in performance among the different oversamplers was found to be statistically significant across the different classifiers and relevant performance metrics by performing a Friedman test. The p-values of this test are reported in Table 5. The superiority of G-SMOTENC was confirmed with the p-values obtained with the Holm-Bonferroni test shown in Table 6. This test showed that G-SMOTENC outperformed with statistical significance the remaining resamplers 82.5% of the comparisons done.

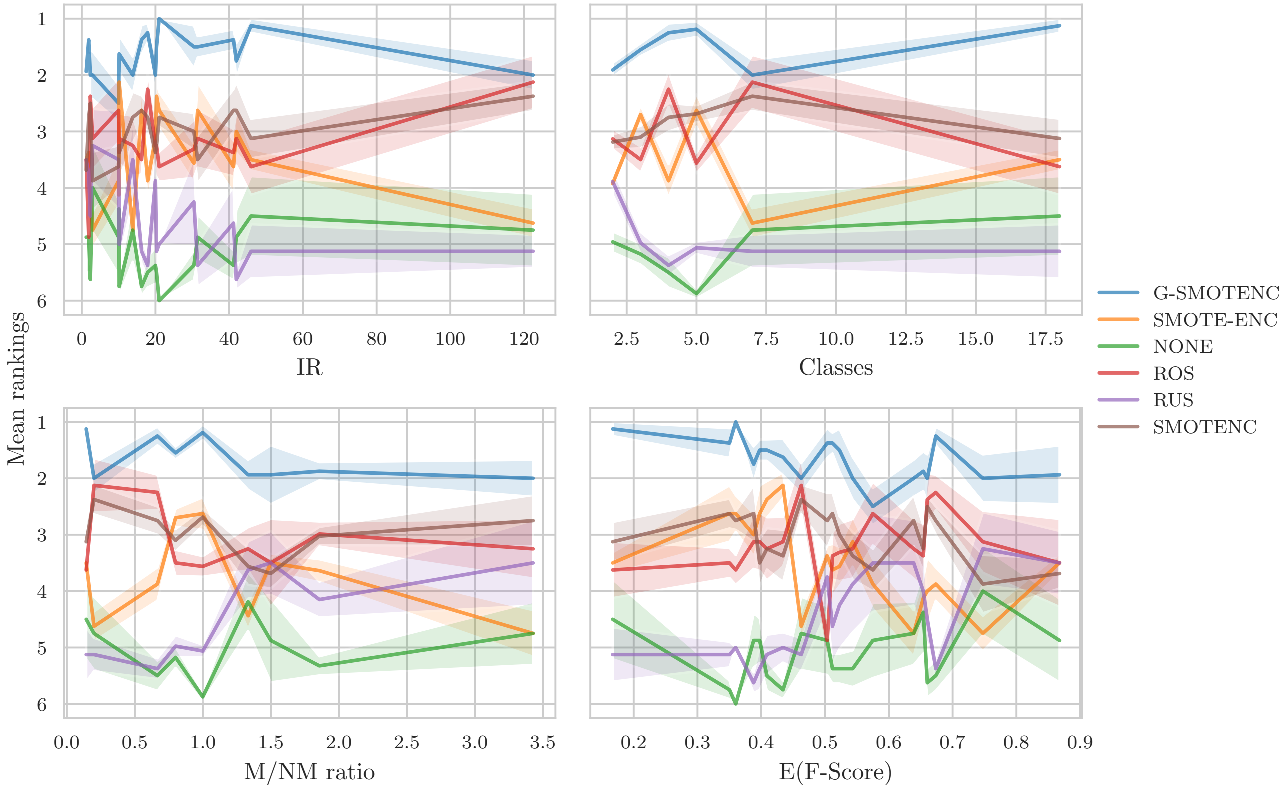

Figure 3: Average ranking of oversamplers over different characteristics of the datasets used in the experiment. Legend: IR — Imbalance Ratio, Classes — Number of classes in the dataset, M/NM ratio — ratio between the number of metric and non-metric features, E(F-Score) — Mean F-Score of dataset across all combinations of classifiers and oversamplers.

The results from this experiment expose some well-known limitations of SMOTE, which become particularly evident with SMOTENC. Specifically, the lack of diversity in the generated data and in some occasions the near-duplication of observations discussed in [10] may be a possible explanation for the performance of SMOTENC being comparable to ROS' performance, visible in Figure 3. In this same figure, 3 groups of resampling methods with comparable performance are visible: (1) G-SMOTENC, the top performing method, (2) SMOTENC, ROS and SMOTE-ENC, where SMOTE-ENC has the most inconsistent behavior and (3) RUS and no oversampling, the worst performing approaches. In addition, G-SMOTENC's superiority seems invariable to the dataset's characteristics, with little overlap with the remaining benchmark methods.

6. Conclusion

In this paper we presented G-SMOTENC, a new oversampling algorithm that combines G-SMOTE and SMOTENC. This oversampling algorithm was developed to leverage G-SMOTE's data selection and generation mechanisms into datasets with mixed data types. This was achieved by encoding and generating nominal feature values using SMOTENC's approach. G-SMOTENC's performance was tested on 20 datasets with different imbalance ratios, metric to non-metric feature ratio and number of classes and compared to no oversampling, SMOTENC, Random Oversampling, Random Undersampling and

SMOTE-ENC using a Decision Tree, K-Nearest Neighbors, Logistic Regression and Random Forest as classifiers.

The results show that G-SMOTENC performs significantly better when compared to its more popular counterparts (SMOTENC, Random Oversampling and Random Undersampling), as well as a recently proposed oversampling algorithm for mixed data types (SMOTE-ENC). The reason for this improvement in performance is related to G-SMOTENC's selection mechanism, which finds a safer region for data generation, along with it generation mechanism which increases the diversity of the generated observations when compared to SMOTENC. The G-SMOTENC implementation used in this study is available in the open source Python library "ML-Research" and is fully compatible with the Scikit-Learn ecosystem.

G-SMOTENC can be seen as a drop-in replacement of SMOTENC, since when $\alpha_{trunc} = 1$, $\alpha_{def} = 1$ and $\alpha_{sel} = minority$ the SMOTENC algorithm is reproduced. G-SMOTENC has 3 additional hyperparameters that allow for a greater customization of the selection and generation mechanisms. However, determining the optimal parameters a priori (*i.e.*, with reduced parameter tuning) is an topic for future work.

References

[1] S. Tyagi and S. Mittal, "Sampling approaches for imbalanced data classification problem in machine learning," in *Proceedings of ICRIC 2019*, pp. 209–221, Springer, 2020.

[2] P. Vuttipittayamongkol, E. Elyan, and A. Petrovski, "On the class overlap problem in imbalanced data classification," *Knowledge-based systems*, vol. 212, p. 106631, 2021.

[3] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information sciences*, vol. 250, pp. 113–141, 2013.

[4] S. Das, S. Datta, and B. B. Chaudhuri, "Handling data irregularities in classification: Foundations, trends, and future challenges," *Pattern Recognition*, vol. 81, pp. 674–693, 2018.

[5] H. Kaur, H. S. Pannu, and A. K. Malhi, "A systematic review on imbalanced data challenges in machine learning: Applications and solutions," *ACM Computing Surveys (CSUR)*, vol. 52, no. 4, pp. 1–36, 2019.

[6] A. Fernández, V. LóPez, M. Galar, M. J. Del Jesus, and F. Herrera, "Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches," *Knowledge-based systems*, vol. 42, pp. 97–110, 2013.

[7] A. N. Tarekegn, M. Giacobini, and K. Michalak, "A review of methods for imbalanced multi-label classification," *Pattern Recognition*, vol. 118, p. 107965, 2021.

[8] J. Lumijärvi, J. Laurikkala, and M. Juhola, "A comparison of different heterogeneous proximity functions and euclidean distance," in *MEDINFO 2004*, pp. 1362–1366, IOS Press, 2004.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, jun 2002.

[10] G. Douzas and F. Bacao, "Geometric smote a geometrically enhanced drop-in replacement for smote," *Information Sciences*, vol. 501, pp. 118–135, 2019.

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

[12] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," in *International conference on intelligent computing*, pp. 878–887, Springer, 2005.

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

[14] G. Douzas, F. Bacao, and F. Last, "Improving imbalanced learning through a heuristic oversampling method based on k-means and smote," *Information Sciences*, vol. 465, pp. 1–20, 2018.

[15] X. Liang, A. Jiang, T. Li, Y. Xue, and G. Wang, "Lr-smote—an improved unbalanced data set oversampling based on k-means and svm," *Knowledge-Based Systems*, vol. 196, p. 105845, 2020.

[16] J. Fonseca, G. Douzas, and F. Bacao, "Improving imbalanced land cover classification with k-means smote: Detecting and oversampling distinctive minority spectral signatures," *Information*, vol. 12, no. 7, p. 266, 2021.

[17] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem," in *Pacific-Asia conference on knowledge discovery and data mining*, pp. 475–482, Springer, 2009.

[18] G. Douzas, F. Bacao, J. Fonseca, and M. Khudinyan, "Imbalanced learning in land cover classification: Improving minority classes' prediction accuracy using the geometric smote algorithm," *Remote Sensing*, vol. 11, no. 24, p. 3040, 2019.

[19] G. Douzas, R. Rauch, and F. Bacao, "G-somo: An oversampling approach based on self-organized maps and geometric smote," *Expert Systems with Applications*, vol. 183, p. 115230, 2021.

[20] A. Salazar, L. Vergara, and G. Safont, "Generative adversarial networks and markov random fields for oversampling very small training sets," *Expert Systems with Applications*, vol. 163, p. 113819, 2021.

[21] A. Koivu, M. Sairanen, A. Airola, and T. Pahikkala, "Synthetic minority oversampling of vital statistics data with generative adversarial networks," *Journal of the American Medical Informatics Association*, vol. 27, no. 11, pp. 1667–1674, 2020.

[22] W. Jo and D. Kim, "Obgan: Minority oversampling near borderline with generative adversarial networks," *Expert Systems with Applications*, vol. 197, p. 116694, 2022.

[23] L. Gonog and Y. Zhou, "A review: generative adversarial networks," in *2019 14th IEEE conference on industrial electronics and applications (ICIEA)*, pp. 505–510, IEEE, 2019.

[24] S. Park and H. Park, "Combined oversampling and undersampling method based on slow-start algorithm for imbalanced network traffic," *Computing*, vol. 103, no. 3, pp. 401–424, 2021.

[25] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[26] A. Bansal and A. Jain, "Analysis of focussed under-sampling techniques with machine learning classifiers," in *2021 IEEE/ACIS 19th International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 91–96, IEEE, 2021.

[27] M. Mukherjee and M. Khushi, "Smote-enc: A novel smote-based method to generate synthetic data for nominal and continuous features," *Applied System Innovation*, vol. 4, no. 1, p. 18, 2021.

[28] K. Ambai and H. Fujita, "Multivariate normal distribution based over-sampling for numerical and categorical features," in *Advancing Technology Industrialization Through Intelligent Software Methodologies, Tools and Techniques: Proceedings of the 18th International Conference on New Trends in Intelligent Software Methodologies, Tools and Techniques (SoMeT)*, vol. 318, p. 107, 2019.

[29] K. Ambai and H. Fujita, "Mndo: Multivariate normal distribution based over-sampling for binary classification.," in *Advancing Technology Industrialization Through Intelligent Software Methodologies, Tools and Techniques: Proceedings of the 17th International Conference on New Trends in Intelligent Software Methodologies, Tools and Techniques (SoMeT)*, pp. 425–438, 2018.

[30] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International journal of pattern recognition and artificial intelligence*, vol. 23, no. 04, pp. 687–719, 2009.

[31] N. Rout, D. Mishra, and M. K. Mallick, "Handling imbalanced data: a survey," in *International proceedings on advances in soft computing, intelligent systems and applications*, pp. 431–443, Springer, 2018.

[32] L. A. Jeni, J. F. Cohn, and F. De La Torre, "Facing imbalanced data–recommendations for the use of performance metrics," in *2013 Humaine association conference on affective computing and intelligent interaction*, pp. 245–251, IEEE, 2013.

[33] N. Japkowicz, "Assessment metrics for imbalanced learning," *Imbalanced learning: Foundations, algorithms, and applications*, pp. 187–206, 2013.

[34] J. Fonseca, G. Douzas, and F. Bacao, "Increasing the effectiveness of active learning: Introducing artificial data generation in active learning for land use/land cover classification," *Remote Sensing*, vol. 13, no. 13, p. 2619, 2021.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[36] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017.

[37] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[38] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the american statistical association*, vol. 32, no. 200, pp. 675–701, 1937.

A. Appendix

Table 7: Wide optimal results

Dataset	Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-E
Abalone	DT	OA	0.221	**0.256**	0.190	0.203	0.207	0.191
Abalone	DT	F-Score	0.168	**0.170**	0.156	0.154	0.132	0.158
Abalone	DT	G-Mean	**0.460**	0.413	0.445	0.457	0.421	0.443
Abalone	KNN	OA	0.215	**0.237**	0.186	0.197	0.188	0.191
Abalone	KNN	F-Score	**0.167**	0.157	0.150	0.151	0.140	0.153
Abalone	KNN	G-Mean	**0.429**	0.391	0.409	0.397	0.421	0.409
Abalone	LR	OA	0.235	**0.272**	0.228	0.229	0.195	0.228
Abalone	LR	F-Score	**0.189**	0.180	0.186	0.179	0.166	0.182
Abalone	LR	G-Mean	**0.473**	0.415	0.466	0.456	0.441	0.464
Abalone	RF	OA	0.237	**0.276**	0.221	0.224	0.197	0.225
Abalone	RF	F-Score	**0.194**	0.174	0.180	0.184	0.162	0.180
Abalone	RF	G-Mean	**0.486**	0.416	0.461	0.465	0.448	0.458
Adult	DT	OA	0.830	**0.835**	0.785	0.800	0.785	0.781
Adult	DT	F-Score	**0.767**	0.763	0.754	0.755	0.744	0.749
Adult	DT	G-Mean	**0.809**	0.747	0.808	0.806	0.801	0.799
Adult	KNN	OA	0.786	**0.805**	0.781	0.763	0.761	0.767
Adult	KNN	F-Score	**0.738**	0.732	0.735	0.718	0.728	0.720
Adult	KNN	G-Mean	0.766	0.724	0.762	0.757	**0.780**	0.752
Adult	LR	OA	0.803	**0.839**	0.803	0.804	0.801	0.799
Adult	LR	F-Score	0.768	**0.773**	0.767	0.771	0.769	0.764
Adult	LR	G-Mean	0.813	0.758	0.805	**0.815**	**0.815**	0.805
Adult	RF	OA	0.820	**0.832**	0.757	0.755	0.753	0.761
Adult	RF	F-Score	**0.769**	0.739	0.727	0.729	0.728	0.732
Adult	RF	G-Mean	0.796	0.711	0.787	**0.797**	**0.797**	0.793
Adult (10)	DT	OA	**0.930**	0.928	0.822	0.789	0.775	0.819
Adult (10)	DT	F-Score	**0.711**	0.708	0.656	0.641	0.630	0.644
Adult (10)	DT	G-Mean	0.812	0.663	0.807	**0.815**	0.808	0.788
Adult (10)	KNN	OA	0.864	**0.909**	0.854	0.851	0.745	0.853
Adult (10)	KNN	F-Score	**0.667**	0.652	0.658	0.648	0.602	0.652
Adult (10)	KNN	G-Mean	0.745	0.629	0.747	0.722	**0.783**	0.712
Adult (10)	LR	OA	0.836	**0.925**	0.837	0.815	0.791	0.831
Adult (10)	LR	F-Score	0.666	**0.705**	0.667	0.663	0.647	0.665
Adult (10)	LR	G-Mean	0.804	0.663	0.787	0.811	**0.814**	0.783
Adult (10)	RF	OA	0.899	**0.924**	0.773	0.763	0.743	0.781
Adult (10)	RF	F-Score	**0.718**	0.615	0.620	0.624	0.610	0.626
Adult (10)	RF	G-Mean	**0.809**	0.579	0.786	0.806	0.806	0.786
Annealing	DT	OA	0.824	**0.843**	0.742	0.733	0.694	0.720
Annealing	DT	F-Score	**0.736**	0.643	0.732	0.724	0.683	0.718
Annealing	DT	G-Mean	**0.914**	0.738	0.909	0.906	0.880	0.901
Annealing	KNN	OA	0.849	0.847	0.829	**0.854**	0.508	0.830
Annealing	KNN	F-Score	0.780	0.724	0.747	**0.783**	0.476	0.741

Continued on next p

Table 7: Wide optimal results

Dataset	Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-E
Annealing	KNN	G-Mean	0.901	0.781	0.867	**0.909**	0.814	0.856
Annealing	LR	OA	0.572	**0.814**	0.573	0.566	0.510	0.552
Annealing	LR	F-Score	**0.620**	0.540	0.617	0.615	0.496	0.499
Annealing	LR	G-Mean	**0.851**	0.663	0.843	0.848	0.811	0.821
Annealing	RF	OA	**0.868**	**0.868**	0.729	0.733	0.637	0.759
Annealing	RF	F-Score	**0.800**	0.644	0.730	0.736	0.641	0.743
Annealing	RF	G-Mean	**0.917**	0.727	0.904	0.910	0.873	0.887
Census	DT	OA	0.942	**0.943**	0.894	0.844	0.795	0.293
Census	DT	F-Score	**0.733**	0.731	0.693	0.652	0.617	0.258
Census	DT	G-Mean	0.813	0.698	0.800	0.814	**0.817**	0.621
Census	KNN	OA	0.874	**0.933**	0.867	0.878	0.731	0.871
Census	KNN	F-Score	0.652	0.648	**0.655**	0.640	0.567	0.641
Census	KNN	G-Mean	0.767	0.620	0.768	0.733	**0.794**	0.740
Census	LR	OA	0.940	**0.949**	0.938	0.940	0.815	0.828
Census	LR	F-Score	0.760	0.743	0.760	**0.762**	0.639	0.630
Census	LR	G-Mean	0.807	0.707	0.782	0.801	**0.837**	0.794
Census	RF	OA	0.876	**0.933**	0.819	0.740	0.714	0.799
Census	RF	F-Score	**0.679**	0.483	0.636	0.580	0.562	0.614
Census	RF	G-Mean	**0.827**	0.500	0.818	0.822	0.814	0.810
Contraceptive	DT	OA	**0.563**	0.538	0.537	0.512	0.525	0.528
Contraceptive	DT	F-Score	**0.549**	0.518	0.529	0.507	0.520	0.521
Contraceptive	DT	G-Mean	**0.661**	0.630	0.646	0.630	0.641	0.638
Contraceptive	KNN	OA	0.465	**0.478**	0.455	0.435	0.468	0.461
Contraceptive	KNN	F-Score	0.460	**0.462**	0.450	0.432	0.461	0.455
Contraceptive	KNN	G-Mean	0.588	0.580	0.579	0.566	**0.590**	0.583
Contraceptive	LR	OA	**0.515**	0.514	0.514	0.510	0.510	0.513
Contraceptive	LR	F-Score	**0.512**	0.492	0.509	0.505	0.506	0.508
Contraceptive	LR	G-Mean	**0.635**	0.604	0.631	0.628	0.627	0.630
Contraceptive	RF	OA	0.553	**0.557**	0.540	0.534	0.526	0.536
Contraceptive	RF	F-Score	**0.545**	0.524	0.535	0.529	0.522	0.530
Contraceptive	RF	G-Mean	**0.659**	0.634	0.653	0.649	0.643	0.649
Contraceptive (10)	DT	OA	**0.645**	**0.645**	0.568	0.528	0.487	0.592
Contraceptive (10)	DT	F-Score	0.479	0.452	0.478	0.454	0.414	**0.490**
Contraceptive (10)	DT	G-Mean	0.644	0.584	**0.648**	0.637	0.610	**0.648**
Contraceptive (10)	KNN	OA	0.524	**0.570**	0.508	0.495	0.451	0.512
Contraceptive (10)	KNN	F-Score	**0.419**	0.404	0.410	0.404	0.368	0.413
Contraceptive (10)	KNN	G-Mean	**0.576**	0.529	0.561	0.569	0.561	0.563
Contraceptive (10)	LR	OA	0.516	**0.622**	0.506	0.489	0.476	0.503
Contraceptive (10)	LR	F-Score	**0.431**	0.375	0.426	0.425	0.411	**0.431**
Contraceptive (10)	LR	G-Mean	0.619	0.526	0.609	**0.624**	0.618	0.621
Contraceptive (10)	RF	OA	0.648	**0.651**	0.569	0.550	0.494	0.573
Contraceptive (10)	RF	F-Score	**0.500**	0.387	0.473	0.471	0.425	0.480
Contraceptive (10)	RF	G-Mean	**0.656**	0.542	0.639	0.650	0.625	0.646
Contraceptive (20)	DT	OA	**0.671**	0.659	0.612	0.556	0.456	0.620
Contraceptive (20)	DT	F-Score	**0.475**	0.430	0.459	0.428	0.371	0.470
Contraceptive (20)	DT	G-Mean	0.643	0.570	0.626	0.632	0.605	**0.645**
Contraceptive (20)	KNN	OA	0.556	**0.600**	0.529	0.541	0.442	0.543
Contraceptive (20)	KNN	F-Score	**0.399**	0.375	0.384	0.389	0.345	0.395

Continued on next pa

Table 7: Wide optimal results

Dataset	Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-E
Contraceptive (20)	KNN	G-Mean	**0.565**	0.519	0.544	0.537	0.549	0.556
Contraceptive (20)	LR	OA	0.506	**0.641**	0.508	0.486	0.440	0.514
Contraceptive (20)	LR	F-Score	**0.397**	0.375	**0.397**	0.389	0.358	0.393
Contraceptive (20)	LR	G-Mean	0.608	0.523	0.604	**0.613**	0.585	0.597
Contraceptive (20)	RF	OA	0.668	**0.674**	0.588	0.562	0.475	0.605
Contraceptive (20)	RF	F-Score	**0.473**	0.384	0.450	0.436	0.389	0.454
Contraceptive (20)	RF	G-Mean	0.659	0.535	0.641	**0.670**	0.633	0.642
Contraceptive (31)	DT	OA	0.667	**0.670**	0.608	0.604	0.440	0.644
Contraceptive (31)	DT	F-Score	**0.454**	0.441	0.438	0.453	0.346	**0.454**
Contraceptive (31)	DT	G-Mean	0.642	0.577	0.605	**0.655**	0.592	0.629
Contraceptive (31)	KNN	OA	0.563	**0.633**	0.545	0.550	0.405	0.548
Contraceptive (31)	KNN	F-Score	**0.403**	0.385	0.384	0.378	0.298	0.387
Contraceptive (31)	KNN	G-Mean	**0.574**	0.527	0.544	0.531	0.511	0.555
Contraceptive (31)	LR	OA	0.500	**0.656**	0.508	0.483	0.423	0.516
Contraceptive (31)	LR	F-Score	**0.379**	0.376	**0.379**	0.374	0.336	**0.379**
Contraceptive (31)	LR	G-Mean	**0.597**	0.523	0.579	0.585	0.580	0.574
Contraceptive (31)	RF	OA	0.681	**0.683**	0.608	0.583	0.442	0.616
Contraceptive (31)	RF	F-Score	0.450	0.378	0.434	0.435	0.349	**0.452**
Contraceptive (31)	RF	G-Mean	**0.647**	0.531	0.630	0.640	0.600	0.626
Contraceptive (41)	DT	OA	0.651	**0.666**	0.588	0.566	0.433	0.589
Contraceptive (41)	DT	F-Score	**0.459**	0.426	0.408	0.409	0.336	0.416
Contraceptive (41)	DT	G-Mean	**0.622**	0.573	0.579	0.589	0.555	0.589
Contraceptive (41)	KNN	OA	0.563	**0.611**	0.546	0.538	0.395	0.541
Contraceptive (41)	KNN	F-Score	**0.393**	0.373	0.381	0.370	0.289	0.373
Contraceptive (41)	KNN	G-Mean	0.542	0.515	**0.550**	0.526	0.515	0.531
Contraceptive (41)	LR	OA	0.525	**0.658**	0.524	0.504	0.435	0.530
Contraceptive (41)	LR	F-Score	0.389	0.375	**0.393**	0.387	0.336	**0.393**
Contraceptive (41)	LR	G-Mean	0.606	0.520	0.604	**0.627**	0.569	0.600
Contraceptive (41)	RF	OA	0.665	**0.681**	0.598	0.588	0.415	0.596
Contraceptive (41)	RF	F-Score	**0.444**	0.378	0.418	0.429	0.323	0.416
Contraceptive (41)	RF	G-Mean	0.612	0.528	**0.616**	**0.616**	0.566	0.608
Covertype	DT	OA	0.580	**0.705**	0.587	0.567	0.450	0.552
Covertype	DT	F-Score	0.484	**0.490**	0.481	0.475	0.361	0.474
Covertype	DT	G-Mean	**0.769**	0.671	0.758	0.758	0.700	0.751
Covertype	KNN	OA	0.690	**0.700**	0.683	0.699	0.454	0.636
Covertype	KNN	F-Score	0.532	0.457	0.535	**0.561**	0.367	0.484
Covertype	KNN	G-Mean	0.745	0.642	0.753	**0.763**	0.691	0.744
Covertype	LR	OA	0.637	**0.721**	0.640	0.611	0.472	0.617
Covertype	LR	F-Score	0.516	0.507	**0.526**	0.492	0.353	0.429
Covertype	LR	G-Mean	**0.792**	0.678	0.786	0.790	0.697	0.725
Covertype	RF	OA	0.598	**0.704**	0.583	0.587	0.485	0.338
Covertype	RF	F-Score	0.517	0.360	0.507	**0.519**	0.394	0.284
Covertype	RF	G-Mean	0.800	0.572	0.799	**0.804**	0.737	0.691
Credit Approval	DT	OA	**0.867**	0.847	0.862	0.861	0.865	0.862
Credit Approval	DT	F-Score	**0.867**	0.845	0.862	0.861	0.865	0.862
Credit Approval	DT	G-Mean	**0.874**	0.848	0.869	0.867	0.872	0.869
Credit Approval	KNN	OA	**0.870**	0.865	0.868	**0.870**	0.865	0.867
Credit Approval	KNN	F-Score	**0.869**	0.864	0.867	**0.869**	0.864	0.866

Continued on next p

Table 7: Wide optimal results

Dataset	Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-E
Credit Approval	KNN	G-Mean	**0.871**	0.865	0.868	**0.871**	0.866	0.867
Credit Approval	LR	OA	0.873	0.868	0.871	**0.874**	0.873	0.873
Credit Approval	LR	F-Score	0.873	0.868	0.871	**0.874**	0.873	0.873
Credit Approval	LR	G-Mean	0.877	0.873	0.877	**0.879**	0.878	0.878
Credit Approval	RF	OA	0.876	**0.877**	0.871	0.868	0.868	0.873
Credit Approval	RF	F-Score	0.876	**0.877**	0.871	0.868	0.868	0.872
Credit Approval	RF	G-Mean	**0.879**	**0.879**	0.876	0.872	0.873	0.875
German Credit	DT	OA	0.704	**0.713**	0.702	0.660	0.644	0.701
German Credit	DT	F-Score	0.662	0.608	0.654	0.633	0.623	**0.664**
German Credit	DT	G-Mean	**0.681**	0.608	0.667	0.663	0.660	0.678
German Credit	KNN	OA	0.681	**0.718**	0.682	0.670	0.641	0.657
German Credit	KNN	F-Score	**0.653**	0.628	0.650	0.636	0.616	0.626
German Credit	KNN	G-Mean	**0.675**	0.621	0.668	0.656	0.642	0.646
German Credit	LR	OA	0.727	**0.751**	0.729	0.724	0.712	0.713
German Credit	LR	F-Score	0.695	0.681	**0.697**	**0.697**	0.686	0.676
German Credit	LR	G-Mean	**0.722**	0.672	0.713	0.720	0.713	0.696
German Credit	RF	OA	**0.760**	0.741	0.739	0.737	0.700	0.726
German Credit	RF	F-Score	0.701	0.580	0.702	**0.709**	0.680	0.688
German Credit	RF	G-Mean	0.715	0.588	0.716	**0.730**	0.719	0.699
German Credit (10)	DT	OA	**0.909**	0.906	0.804	0.713	0.696	0.752
German Credit (10)	DT	F-Score	**0.575**	0.539	0.572	0.526	0.511	0.539
German Credit (10)	DT	G-Mean	0.628	0.535	0.629	**0.644**	0.631	0.593
German Credit (10)	KNN	OA	0.787	**0.913**	0.757	0.835	0.684	0.795
German Credit (10)	KNN	F-Score	0.578	**0.581**	0.558	0.573	0.528	0.560
German Credit (10)	KNN	G-Mean	0.662	0.559	0.643	0.588	**0.667**	0.597
German Credit (10)	LR	OA	0.839	**0.904**	0.831	0.799	0.682	0.829
German Credit (10)	LR	F-Score	0.619	0.596	0.610	**0.620**	0.550	**0.620**
German Credit (10)	LR	G-Mean	0.683	0.578	0.675	0.716	**0.722**	0.681
German Credit (10)	RF	OA	**0.910**	0.909	0.865	0.877	0.696	0.860
German Credit (10)	RF	F-Score	0.624	0.476	0.614	**0.661**	0.557	0.610
German Credit (10)	RF	G-Mean	0.653	0.500	0.646	0.709	**0.729**	0.628
German Credit (20)	DT	OA	**0.952**	**0.952**	0.875	0.795	0.668	0.880
German Credit (20)	DT	F-Score	0.573	0.525	0.559	0.522	0.457	**0.579**
German Credit (20)	DT	G-Mean	0.666	0.529	0.679	**0.690**	0.629	0.674
German Credit (20)	KNN	OA	0.856	**0.952**	0.826	0.905	0.679	0.872
German Credit (20)	KNN	F-Score	**0.561**	0.535	0.528	0.556	0.491	0.538
German Credit (20)	KNN	G-Mean	0.692	0.527	0.635	0.570	**0.709**	0.601
German Credit (20)	LR	OA	0.913	**0.952**	0.910	0.838	0.680	0.891
German Credit (20)	LR	F-Score	**0.596**	0.534	0.593	0.553	0.473	0.568
German Credit (20)	LR	G-Mean	0.651	0.531	0.627	0.661	**0.682**	0.616
German Credit (20)	RF	OA	**0.954**	0.952	0.920	0.931	0.709	0.920
German Credit (20)	RF	F-Score	**0.597**	0.488	0.574	0.572	0.493	0.576
German Credit (20)	RF	G-Mean	0.681	0.500	0.625	0.674	**0.691**	0.639
German Credit (30)	DT	OA	**0.968**	0.963	0.885	0.856	0.628	0.888
German Credit (30)	DT	F-Score	**0.558**	0.509	0.526	0.506	0.413	0.528
German Credit (30)	DT	G-Mean	**0.686**	0.509	0.631	0.602	0.565	0.609
German Credit (30)	KNN	OA	0.902	**0.968**	0.849	0.935	0.697	0.900
German Credit (30)	KNN	F-Score	**0.530**	0.492	0.512	0.519	0.473	0.507

Continued on next pa

Table 7: Wide optimal results

Dataset	Classifier	Metric	G-SMOTENC	NONE	SMOTENC	ROS	RUS	SMOTE-E
German Credit (30)	KNN	G-Mean	0.681	0.500	0.588	0.536	**0.705**	0.536
German Credit (30)	LR	OA	0.921	**0.967**	0.918	0.877	0.611	0.920
German Credit (30)	LR	F-Score	**0.578**	0.516	0.577	0.537	0.421	0.571
German Credit (30)	LR	G-Mean	0.649	0.510	0.650	**0.661**	0.660	0.608
German Credit (30)	RF	OA	**0.968**	**0.968**	0.942	0.954	0.705	0.947
German Credit (30)	RF	F-Score	**0.592**	0.492	0.563	0.589	0.474	0.560
German Credit (30)	RF	G-Mean	**0.689**	0.500	0.601	0.606	0.679	0.618
German Credit (41)	DT	OA	**0.976**	0.971	0.916	0.905	0.635	0.898
German Credit (41)	DT	F-Score	**0.563**	0.493	0.544	0.502	0.408	0.552
German Credit (41)	DT	G-Mean	**0.636**	0.497	0.615	0.520	0.524	0.626
German Credit (41)	KNN	OA	0.929	**0.976**	0.876	0.944	0.674	0.920
German Credit (41)	KNN	F-Score	**0.524**	0.494	0.500	0.502	0.440	0.493
German Credit (41)	KNN	G-Mean	0.593	0.500	0.558	0.516	**0.630**	0.504
German Credit (41)	LR	OA	0.940	**0.976**	0.943	0.927	0.641	0.932
German Credit (41)	LR	F-Score	0.546	0.494	**0.552**	0.515	0.420	0.516
German Credit (41)	LR	G-Mean	**0.602**	0.500	0.592	0.598	0.597	0.521
German Credit (41)	RF	OA	**0.976**	**0.976**	0.961	0.969	0.636	0.962
German Credit (41)	RF	F-Score	**0.598**	0.494	0.566	0.591	0.413	0.561
German Credit (41)	RF	G-Mean	0.621	0.500	**0.622**	0.614	0.572	0.616
Heart Disease	DT	OA	0.532	**0.566**	0.509	0.473	0.430	0.509
Heart Disease	DT	F-Score	**0.371**	0.322	0.342	0.331	0.295	0.339
Heart Disease	DT	G-Mean	**0.588**	0.534	0.563	0.545	0.515	0.548
Heart Disease	KNN	OA	0.538	**0.564**	0.535	0.534	0.504	0.528
Heart Disease	KNN	F-Score	**0.363**	0.287	0.360	0.352	0.341	0.348
Heart Disease	KNN	G-Mean	**0.571**	0.509	**0.571**	0.560	0.557	0.557
Heart Disease	LR	OA	0.558	**0.584**	0.557	0.536	0.480	0.562
Heart Disease	LR	F-Score	0.397	0.329	0.395	0.374	0.333	**0.400**
Heart Disease	LR	G-Mean	0.601	0.539	0.601	0.603	0.567	**0.610**
Heart Disease	RF	OA	0.553	**0.601**	0.546	0.539	0.480	0.555
Heart Disease	RF	F-Score	**0.385**	0.314	0.366	0.360	0.326	0.378
Heart Disease	RF	G-Mean	**0.600**	0.531	0.580	0.569	0.566	0.582
Heart Disease (21)	DT	OA	0.532	**0.566**	0.512	0.486	0.431	0.510
Heart Disease (21)	DT	F-Score	**0.376**	0.296	0.341	0.336	0.311	0.342
Heart Disease (21)	DT	G-Mean	**0.598**	0.509	0.558	0.562	0.538	0.551
Heart Disease (21)	KNN	OA	0.561	**0.569**	0.543	0.541	0.491	0.550
Heart Disease (21)	KNN	F-Score	**0.385**	0.312	0.365	0.363	0.334	0.365
Heart Disease (21)	KNN	G-Mean	**0.589**	0.520	0.570	0.566	0.546	0.570
Heart Disease (21)	LR	OA	0.573	**0.592**	0.565	0.547	0.525	0.561
Heart Disease (21)	LR	F-Score	**0.408**	0.331	0.405	0.387	0.343	0.405
Heart Disease (21)	LR	G-Mean	**0.638**	0.540	0.610	0.602	0.583	0.627
Heart Disease (21)	RF	OA	0.577	**0.608**	0.565	0.561	0.517	0.561
Heart Disease (21)	RF	F-Score	**0.417**	0.323	0.390	0.383	0.337	0.386
Heart Disease (21)	RF	G-Mean	**0.621**	0.536	0.596	0.593	0.567	0.590