International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Parameter Analysis of Differential Evolution Based Oversampling Approach for Highly Imbalanced Datasets

Mehmet Akif Şahman*[1]

Submitted: 20/04/2021 Accepted : 10/06/2021

Abstract: Nowadays, almost all performed activities are saved into databases. Data mining methods such as classifiers utilize these datasets for discovering hidden patterns and rules. Proposed methods for classification problems are generally developed considering approximately balanced datasets. However, imbalanced datasets that have unequal instance numbers in their classes emerge as a common problem in most real domains. Many approaches at the data level are proposed to enable better classification of imbalanced datasets. Differential Evolution Based Oversampling Approach For Highly Imbalanced Datasets (DEBOHID) is one of the proposed methods in order to handle this issue on imbalanced datasets. DEBOHID approach utilizes the crossover and mutation processes of DE for generating new synthetic samples. The parameters used by the crossover and mutation processes affect the solution quality. Therefore, in this study, solution quality in highly imbalanced datasets for different crossover and mutation parameter values of DEBOHID approach is investigated. Experimental studies are carried out by using three classifiers and two evaluation metrics for different parameter values. The obtained results are compared with well-known approaches in the literature.

Keywords: *Imbalanced data learning, Differential evolution, Oversampling, Class imbalance, Parameter Analysis*

1. Introduction

Data sets with significant differences in instance numbers of class are called imbalanced data sets. In binary imbalanced datasets, the class that has a large number of samples is called the major class, and the fewer samples are called the minor class. Learning from the imbalanced datasets is defined tough task.

Especially in the field of data mining, machine learning approaches have been developed to operate on balanced data. However, there are imbalanced datasets in many real-life domains such as fault diagnosis of the wind turbine systems [1], term/preterm electro hysterography classification [2], disease diagnosis [3], image segmentation [4], abnormal activity recognition [5], telecommunications management [6] and so on. To overcome the class imbalance problem, pre-processing methods, cost-sensitive learning methods, algorithm-cantered methods, and finally hybrid methods are used [7].

Pre-processing methods handle the class imbalance problem at the data level. In order to remedy the imbalance in the classes, three approaches are used at the data pre-processing level. These approaches are called under-sampling, over-sampling, and hybrid-sampling. To deal with the imbalance in classes, that is, until the number of instances in the classes is balanced, either the major class is under-sampled, or the minor class is over-sampled, sometimes both processes are used together (hybrid-sampling) [7]. In this study, over-sampling methods are focused. When the literature is examined, it is seen that there are many over-sampling methods. Synthetic Minority Over-sampling Technique (SMOTE) is one of the well-known methods that was proposed by Chawla et al. in 2002 [8]. In the SMOTE method, instances in the minor class are over-sampled only from the minor class until they are equivalent to the major class. While synthetic data is produced using an instance from the Minor class, the closest k neighbors to the instance in the minor class are employed. Unlike the SMOTE method, borderline-SMOTE approaches make use of instances at the borderline of minority and majority class to produce synthetic instances [9]. Although borderline-SMOTE1 exploits from minority instances at the borderline, in borderline-SMOTE2 both minority and majority samples at the borderline are used. The Safe-Level-Synthetic Minority Over-Sampling TEchnique (Safe-Level-SMOTE) approach works on the basis of SMOTE, and this approach was proposed by Bunkhumpornpat et al. In 2009 [10]. In the Safe-level-SMOTE method, five safe levels are determined according to a randomly selected instance from the nearest neighbor in the minority class. Synthetic data production is performed according to the safe level ratio of the selected instance. An adaptive synthetic (ADASYN) sampling approach was proposed by He et al. in imbalanced data sets for the two-class classification problem [11]. In ADASYN approach, synthetic data is produced according to the learning difficulty level of the instances in the minority class and the distribution of the instances in the data set.

The approaches were proposed such as SMOTE-Tomek Links (S-TL) [12], SMOTE-ENN (S-ENN) [12], and SMOTE-RSB (S-RSB) [13] by the researchers use both under-sampling and over-sampling pre-processing approaches together. S-TL, which is the hybrid-sampling method, is used to under-sampling data in both the majority and minority classes by using Tomek Links approach [14], which is used also as the under-sampling method. In the S-TL approach, synthetic data for the minority class is over-sampled

[1] *Department of Electrical and Electronics Engineering, Faculty of Technology, Selçuk University, Konya, Turkey*
ORCID ID : 0000-0002-1718-3777
** Corresponding Author Email: asahman@selcuk.edu.tr*

using the SMOTE approach. Then, both majority and minority class samples are removed in order to improve the impact of the prediction effectiveness. S-ENN uses Wilson's Edited Nearest Neighbor rule and removes the 3 closest neighbors of samples that are not classified as different from S-TL. Thus, it performs deep data cleaning. Rough Set Theory [15] is used in the S-RSB approach. After utilized SMOTE over-sampling method to generate synthetic samples, all samples are evaluated according to their similarity. Samples with similarities higher than the specified threshold value are removed and the remaining samples are determined as the result set.

Meta-heuristic methods such as Selection through CHC (EGIS-CHC) [16], Binary Particle Swarm Optimization (BPSO) [17], a hybrid evolutionary pre-processing method based (SMOTE + CHC) [18], CHC Adaptive Search Algorithm [19], Ant Colony Optimization (ACO) algorithm [20], and Artificial Bee Colony Sampling (ABC-Sampling) [21] have been used to tackle imbalanced data in the literature.

The Differential Evolution (DE) algorithm was proposed by Storn and Price [22]. The inspiration of the DE algorithm is a difference of the vectors. In the DE algorithms, such as crossover, mutation, and selection evolutionary processes are used. A differential evolution-based oversampling approach for highly imbalanced datasets (DEBOHID) was proposed by Kaya et al. in 2021 [23]. While measuring the performance of the proposed approach, 44 highly imbalanced ratio datasets from the KEEL repository [24] were taken. Performance metrics were determined as AUC and G-Mean. In addition, Support Vector Machines (SVM), k-Nearest Neighbor (kNN), and Decision Tree (DT) were used as classifiers. The classification success of DEBOHID was compared with the original data, and the SMOTE, S-RSB, S-ENN, Safe-level-SMOTE, borderline-SMOTE1, borderline-SMOTE2, S-TL, ADASYN approaches. According to the comparisons made with the results obtained, the DEBOHID approach was found to be superior.

DEBOHID uses the basic strategies of DE, and these strategies use two parameters. These parameters are the scaling factor (F), and crossover rate value (CR). In the DEBOHID study, the F value, and the CR value were determined as 0.3, 0.6, respectively. In this study, the effect of different F and CR parameter values on the success of the DEBOHID method is investigated.

The remainder of the paper is organized as follows. Evaluation metrics for classifiers are presented in section 2. The DEBOHID approach is explained in section 3. Experimental setups are given in section 4. The experimental results and discussions are given in Section 5 and finally, the work is concluded in Section 6.

2. Evaluation Metrics for Classifiers

Confusion matrix analysis is often performed to evaluate the performance of classifiers. The confusion matrix for the two-class problem is given in Table 1. In the confusion matrix, the actual positive and negative are shown in the row, and the predicted positive and negative are shown in the columns. The number of correctly classified positive and negative samples are denoted in True Positive (TP) and True Negative (TN). Likewise, wrongly classified positive and negative samples are indicated as False Positive (FP) and False Negative (FN).

The widely used metric to evaluate classification success is accuracy (Eq.1). However, in highly imbalanced datasets the accuracy metric tends to predict the majority class correctly, whereas the minority class is ignored and neglected. For this reason, it is not reliable to use the accuracy metric as the performance measure of the classifiers that are used for highly imbalanced data sets.

Table 1. Confusion matrix of two-class problem

	Predicted Negative	Predicted Positive
Actual Negative	True Negatives (TN)	False Positives (FP)
Actual Positive	False Negatives (FN)	True Positives (TP)

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{1}$$

Area Under the Curve (AUC) and geometric mean (G-Mean) are suggested in order to measure the performance of the classifiers in the literature [25]. While calculating the AUC value, TP_{rate} (Eq.3) and FP_{rate} (Eq.4) values are used.

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{2}$$

$$TP_{rate} = recall = \frac{TP}{TP + FN} \tag{3}$$

$$FP_{rate} = \frac{FP}{FP + TN} \tag{4}$$

The geometric mean of the classification success of the majority and minority class is given with the G-Mean (Eq.5) metric.

$$G - Mean = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}} \tag{5}$$

In this study, AUC and G-mean metrics will be used while evaluating the performance of classifiers for different parameter values in DEBOHID.

3. A differential evolution based oversampling approach for highly imbalanced datasets (DEBOHID)

The oversampling approach, SMOTE, utilizes only the minority class instances when generating new synthetic instances. Selected base instance from the minority class exploits one of the k nearest neighbor that is randomly chosen and situated in the minority class in order to create a new synthetic instance. Using only the attributes of one neighbor sample in the SMOTE approach can negatively affect classification success. Therefore, in the DEBOHID approach, unlike the SMOTE approach, a better training environment is provided for the classifier models by utilizing the attributes of more than one neighbor.

In DEBOHID approach, it is aimed to balance out the imbalanced data sets. DEBOHID approach performs oversampling at the data level. Instances to be used to create synthetic data are selected only from the minority class.

By DEBOHID, instances are produced for the minority class until the number of instances in the majority class is equal. Then, the balanced data set is used in classifiers. Thus, it is desired to increase the classification success.

While generating synthetic instances from minority class, mutation (Eq.6) and crossover (Eq.7) processes of the meta-heuristic

algorithm DE are utilized. The equations of these processes are given below.

$$V_i = X_{r1} + F \times (X_{r2} - X_{r3}) \;\; r1 \neq r2 \neq r3 \tag{6}$$

where, V_i represents the vector of the *i*th donor, X depicts the position vector of the population. r_1, r_2, r_3 are the randomly selected indexes, and F is the vector of the scaling factor that is determined between 0 and 2.

$$T_{i,j} = \begin{cases} V_{i,j}, & if\ rand[0,1] \leq CR\ or\ j = rj \\ X_{i,j}, & if\ rand[0,1] > CR\ and\ j \neq rj \end{cases} \tag{7}$$

where, $T_{i,j}$ is *j*th dimension of the *i*th trial vector, $V_{i,j}$ is the *j*th dimension of the *i*th donor vector, and $X_{i,j}$ is the *j*th dimension of the *i*th population vector. CR represents the crossover rate, $rand$ indicates the number generated randomly in the range [0,1], and j shows the dimension index.

Three samples are needed so as to generate the donor vector for using the mutation process. These three samples are the three closest neighbors to the minority sample to be produced. The donor vector and the selected sample are crossed over in order to generate a new sample for the minority class. The pseudo-code of the DEBOHID is given as follows (Fig.1).

```
Load the train data
Determine the minority class data as Minor from train data
Determine the majority class data as Major from train data
Determine the k value of knnsearch (k=3)
Determine the F value (scaling factor)
Determine the CR value (crossover rate)
Determine the T as the count of Minority class data
Determine the D as the dimension of the data
Determine MinB the minimum boundaries vector of attributes of the data
Determine MaxB the maximum boundaries vector of attributes of the data
Determine NOS the number of minor class samples to be generated
k=3;  // For the three nearest neighbor samples
FOR i = 1:NOS
  Selected_Sample = Minor[i];
  // knnsearch returns the three nearest neighbors sample to the selected sample.
  [N1, N2, N3] = knnsearch(Minor, Selected_Sample, k);
  Donor_Vector = N1 + F * (N2 – N3);
  FOR j = 1:D // Generate of mutation and inverse mutation vectors
        IF random(1)< CR THEN
          Mutation_Mask[j] = 1;
          Inv_Mutation_Mask[j] = 0;
        ELSE
          Mutation_Mask[j] = 0;
          Inv_Mutation_Mask[j] = 1;
        END
  END
  New_Sample=(Donor_Vector*Mutation_Mask)+(Selected_Sample* Inv_Mutation_Mask);
  FOR j = 1:D // Upper and lower bound limitation
        IF (New_Sample[j]<MinB[j]) THEN New_Sample[j]=MinB[j]; END
        IF (New_Sample[j]>MaxB[j]) THEN New_Sample[j]=MaxB[j]; END
  END
[Generated_Minor_Samples] = Add_Generated_Minor_Samples(New_Sample);
END
Balanced_Train_Data = Combine(Generated_Minor_Samples, Minor,Major);
RETURN Balanced_Train_Data;
```

Fig. 1. The pseudocode of DEBOHID [23]

4. Experimental Setup

The experimental studies are carried out on 44 highly imbalanced data sets that are selected from the UCI repository [26] in order to compare the performance of the DEBOHID with different parameter settings. The number of total samples, the number of features, the ratio of the minority class, the ratio of the majority class, and the imbalance ratio properties of the highly imbalanced data sets are given in Table 2.

All datasets given in Table 2 and used in experimental studies are binary classification datasets. When the datasets are analyzed, the rate of minority class instances in the datasets is less than 10%. It is seen that the rate of instances in the majority class is more than 90%. These rates show that the data sets to be used for experimental studies are highly imbalanced data sets.

Additionally, the imbalance ratio (Majority Class % / Minority Class %) is calculated as the ratio of the instance numbers in the majority class to the instance numbers in the minority class.

Knowledge Extraction based on Evolutionary Learning (KEEL) [24] is developed to use for data mining. The data to be used in the experimental studies are taken from the KEEL repository [27] as 5-Fold cross-validated (%80 train and %20 test). In DEBOHID, k is set as 3 for knnsearch.

In experimental studies, three (Support Vector Machines (SVM) - fitcsvm, k-Nearest Neighbor (kNN) - fitcknn, and Decision Tree (DT) - fitctree) classifiers by default settings were used in Matlab® software. Besides, AUC and G-Mean are used as a performance metric for DEBOHID under different peculiar parameters.

In this study, F and CR peculiar parameters of DEBOHID are run with different values. The value of the parameter F is changed from 0.1 to 2, increasing by 0.2. In addition, the value of the CR parameter is changed from 0.1 to 1, increasing by 0.1.

Table 2. Highly imbalanced data sets and their properties

No	Dataset Name	Total Samples	Features	Minority Class %	Majority Class %	Imbalance Ratio
1	ecoli0137vs26	281	7	2.49	97.51	39.15
2	shuttle0vs4	1829	9	6.72	93.28	13.87
3	yeastB1vs7	459	7	6.53	93.47	14.3
4	shuttle2vs4	129	9	4.65	95.35	20.5
5	glass016vs2	192	9	8.85	91.15	10.29
6	glass016vs5	184	9	4.89	95.11	19.44
7	pageblocks13vs4	472	10	5.93	94.07	15.85
8	yeast05679vs4	528	8	9.66	90.34	9.35
9	yeast1289vs7	947	8	3.16	96.84	30.5
10	yeast1458vs7	693	8	4.33	95.67	22.1
11	yeast2vs4	514	8	9.92	90.08	9.08
12	Ecoli4	336	7	6.74	93.26	13.84
13	Yeast4	1484	8	3.43	96.57	28.41
14	Vowel0	988	13	9.01	90.99	10.1
15	Yeast2vs8	482	8	4.15	95.85	23.1
16	Glass4	214	9	6.07	93.93	15.47
17	Glass5	214	9	4.2	95.8	22.81
18	Glass2	214	9	7.94	92.06	11.59
19	Yeast5	1484	8	2.96	97.04	32.78
20	Yeast6	1484	8	2.49	97.51	39.16
21	abalone19	4174	8	0.77	99.23	128.87
22	abalone918	731	8	5.75	94.25	16.4
23	cleveland0vs4	177	13	7.34	92.66	12.61
24	ecoli01vs235	244	7	2.86	97.14	9.16
25	ecoli01vs5	240	7	2.91	97.09	11
26	ecoli0146vs5	280	7	2.5	97.5	13
27	ecoli0147vs2356	336	7	2.08	97.92	10.58
28	ecoli0147vs56	332	7	2.1	97.9	12.28
29	ecoli0234vs5	202	7	3.46	96.54	9.1
30	ecoli0267vs35	224	7	3.12	96.88	9.18
31	ecoli034vs5	300	7	2.33	97.67	9
32	ecoli0346vs5	205	7	3.41	96.59	9.25
33	ecoli0347vs56	257	7	2.72	97.28	9.28
34	ecoli046vs5	203	7	3.44	96.56	9.15
35	ecoli067vs35	222	7	3.15	96.85	9.09
36	ecoli067vs5	220	7	3.18	96.82	10
37	glass0146vs2	205	9	4.39	95.61	11.05
38	glass015vs2	172	9	5.23	94.77	9.11
39	glass04vs5	92	9	9.78	90.22	9.22
40	glass06vs5	108	9	8.33	91.67	11
41	led7digit02456789vs1	443	7	1.58	98.42	10.97
42	yeast0359vs78	506	8	9.8	90.2	9.12
43	yeast0256vs3789	1004	8	9.86	90.14	9.14
44	yeast02579vs368	1004	8	9.86	90.14	9.14

5. Experimental Results and Discussions

The DE meta-heuristic algorithm is used processes such as crossover, mutation, and selection. The DEBOHID algorithm utilizes mutation and crossover processes of DE. F parameter is used for mutation process and CR parameter is used for crossover process. The change of these parameters also changes the classification success of the DEBOHID algorithm.

10 different parameter values for F parameter and 9 different parameter values for CR were determined for experimental studies. Therefore, DEBOHID is run 90 times for different parameter values. According to the F and CR parameters, 90 versions (DP01-DP90) are named as in Table 3.

First of all, the experimental studies of the 90 versions are compared within themselves according to the mean results obtained from all datasets using three classifiers and for two metrics. Obtained results are depicted in Table 4.

The first rank results for each classifier and metric are given in Table 4 in bold. Besides, in Table 4, the total rank value is given according to the rank results obtained as well.

When the results are examined in Table 4, the best results for kNN-AUC and kNN-GMean are obtained from DP14 and DP15 versions, respectively. The best results for DT-AUC and DT-GMean are obtained from DP77 and DP70 versions, respectively. The best results for SVM-AUC and SVM-GMean are obtained with DP08 and DP33 versions, respectively.

Table 3. Naming of DEBOHID versions for different parameter values

No	F	CR	Name	No	F	CR	Name	No	F	CR	Name
1	0.1	0.1	DP01	31	0.7	0.4	DP31	61	1.3	0.7	DP61
2	0.1	0.2	DP02	32	0.7	0.5	DP32	62	1.3	0.8	DP62
3	0.1	0.3	DP03	33	0.7	0.6	DP33	63	1.3	0.9	DP63
4	0.1	0.4	DP04	34	0.7	0.7	DP34	64	1.5	0.1	DP64
5	0.1	0.5	DP05	35	0.7	0.8	DP35	65	1.5	0.2	DP65
6	0.1	0.6	DP06	36	0.7	0.9	DP36	66	1.5	0.3	DP66
7	0.1	0.7	DP07	37	0.9	0.1	DP37	67	1.5	0.4	DP67
8	0.1	0.8	DP08	38	0.9	0.2	DP38	68	1.5	0.5	DP68
9	0.1	0.9	DP09	39	0.9	0.3	DP39	69	1.5	0.6	DP69
10	0.3	0.1	DP10	40	0.9	0.4	DP40	70	1.5	0.7	DP70
11	0.3	0.2	DP11	41	0.9	0.5	DP41	71	1.5	0.8	DP71
12	0.3	0.3	DP12	42	0.9	0.6	DP42	72	1.5	0.9	DP72
13	0.3	0.4	DP13	43	0.9	0.7	DP43	73	1.7	0.1	DP73
14	0.3	0.5	DP14	44	0.9	0.8	DP44	74	1.7	0.2	DP74
15	0.3	0.6	DP15	45	0.9	0.9	DP45	75	1.7	0.3	DP75
16	0.3	0.7	DP16	46	1.1	0.1	DP46	76	1.7	0.4	DP76
17	0.3	0.8	DP17	47	1.1	0.2	DP47	77	1.7	0.5	DP77
18	0.3	0.9	DP18	48	1.1	0.3	DP48	78	1.7	0.6	DP78
19	0.5	0.1	DP19	49	1.1	0.4	DP49	79	1.7	0.7	DP79
20	0.5	0.2	DP20	50	1.1	0.5	DP50	80	1.7	0.8	DP80
21	0.5	0.3	DP21	51	1.1	0.6	DP51	81	1.7	0.9	DP81
22	0.5	0.4	DP22	52	1.1	0.7	DP52	82	1.9	0.1	DP82
23	0.5	0.5	DP23	53	1.1	0.8	DP53	83	1.9	0.2	DP83
24	0.5	0.6	DP24	54	1.1	0.9	DP54	84	1.9	0.3	DP84
25	0.5	0.7	DP25	55	1.3	0.1	DP55	85	1.9	0.4	DP85
26	0.5	0.8	DP26	56	1.3	0.2	DP56	86	1.9	0.5	DP86
27	0.5	0.9	DP27	57	1.3	0.3	DP57	87	1.9	0.6	DP87
28	0.7	0.1	DP28	58	1.3	0.4	DP58	88	1.9	0.7	DP88
29	0.7	0.2	DP29	59	1.3	0.5	DP59	89	1.9	0.8	DP89
30	0.7	0.3	DP30	60	1.3	0.6	DP60	90	1.9	0.9	DP90

Table 4. Friedman rank test results

Name	kNN-AUC	kNN-GMean	DT-AUC	DT-GMean	SVM-AUC	SVM-GMean	Total	Name	kNN-AUC	kNN-GMean	DT-AUC	DT-GMean	SVM-AUC	SVM-GMean	Total
DP01	38	44	88	87	69	55	381	DP46	37	42	74	76	85	84	398
DP02	15	17	87	88	55	52	314	DP47	55	55	41	50	50	57	308
DP03	14	16	77	79	40	43	269	DP48	34	30	54	57	60	66	301
DP04	7	7	80	81	35	47	257	DP49	8	8	48	41	62	62	229
DP05	9	12	85	85	33	37	261	DP50	5	6	31	34	43	54	173
DP06	13	11	76	75	14	11	200	DP51	39	32	66	58	11	12	218
DP07	3	4	69	72	16	15	179	DP52	52	47	34	32	30	40	235
DP08	**19**	**20**	**78**	**77**	**1**	**2**	**197**	DP53	76	77	28	33	26	23	263
DP09	20	28	89	90	45	49	321	DP54	83	81	53	52	36	25	330
DP10	29	39	90	89	88	88	423	DP55	66	67	49	66	77	80	405
DP11	22	24	86	86	63	60	341	DP56	62	64	57	59	44	41	327
DP12	18	19	84	78	67	61	327	DP57	27	26	51	51	24	24	203
DP13	4	5	70	68	59	58	264	DP58	40	38	25	26	68	69	266
DP14	**1**	**2**	**64**	**62**	**9**	**10**	**148**	DP59	53	48	5	7	28	29	170
DP15	**2**	**1**	**63**	**60**	**19**	**19**	**164**	DP60	75	76	11	9	72	59	302
DP16	26	23	58	53	4	5	169	DP61	69	70	16	19	41	39	254
DP17	16	14	61	61	37	31	220	DP62	74	75	12	17	39	33	250
DP18	23	27	68	67	53	48	286	DP63	87	87	46	49	22	22	313
DP19	45	51	82	82	86	85	431	DP64	43	46	71	71	70	67	368
DP20	36	33	72	70	31	30	272	DP65	57	61	43	45	75	75	356
DP21	10	9	75	74	23	27	218	DP66	77	74	13	18	42	46	270
DP22	24	22	39	38	2	3	128	DP67	63	58	38	24	47	53	283
DP23	17	13	24	31	18	21	124	DP68	41	43	14	13	21	20	152
DP24	**12**	**15**	**18**	**23**	**20**	**17**	**105**	DP69	67	69	17	16	38	28	235
DP25	6	3	45	46	12	14	126	**DP70**	**80**	**80**	**3**	**1**	**61**	**72**	**297**
DP26	31	29	65	64	6	6	201	DP71	81	83	27	20	48	38	297
DP27	30	41	32	39	13	8	163	DP72	89	89	35	29	82	81	405
DP28	59	63	83	83	74	71	433	DP73	50	60	81	84	79	78	432
DP29	35	37	62	65	76	77	352	DP74	61	57	22	30	32	36	238
DP30	33	35	56	63	52	45	284	DP75	47	45	30	27	65	68	282
DP31	28	25	60	55	8	7	183	DP76	56	56	19	21	87	86	325
DP32	25	21	2	4	49	42	143	**DP77**	**49**	**50**	**1**	**2**	**80**	**79**	**261**
DP33	**21**	**18**	**42**	**42**	**3**	**1**	**127**	DP78	82	84	40	28	66	70	370
DP34	48	36	6	15	34	35	174	DP79	78	78	7	3	54	51	271
DP35	71	68	4	12	57	44	256	DP80	86	86	50	47	73	76	418
DP36	70	71	33	37	5	4	220	DP81	90	90	23	22	58	65	348
DP37	46	49	79	80	78	74	406	DP82	64	65	73	73	90	90	455
DP38	42	40	67	69	51	56	325	DP83	54	52	44	44	56	64	314
DP39	44	34	26	43	64	63	274	DP84	60	59	29	25	46	50	269
DP40	11	10	37	40	27	32	157	DP85	51	53	15	6	83	83	291
DP41	32	31	59	56	15	16	209	DP86	72	72	21	11	29	34	239
DP42	68	66	36	35	10	13	228	DP87	79	79	20	14	71	73	336
DP43	58	54	8	8	7	9	144	DP88	84	85	10	5	81	82	347
DP44	65	62	9	10	25	26	197	DP89	85	82	52	36	84	87	426
DP45	73	73	47	54	17	18	282	DP90	88	88	55	48	89	89	457

When the results are examined in Table 4 according to the total rank value obtained, the DP24 version of DEBOHID has the best rank value. The results obtained can be evaluated in different ways, but it seems appropriate to choose the version with a higher total rank among the DEBOHID versions for subsequent comparisons. Therefore, the DP24 version is compared with well-known methods of sampling in literature such as SMOTE, S-TL, S-ENN, Border1, Border2, Safelevel, ADASYN, and S-RSB *, and the original DEBOHID (DP15) as well for three classifiers and two metrics. The results of the sampling approaches used in the comparisons in experimental studies were taken from the study of Kaya et al. [23]. The mean results obtained from classifiers and evaluation metrics are given (Table 5- Table 10). The best results obtained by the methods are indicated in bold.

In Table 5-Table 10, information on the mean of the methods obtained from all imbalanced datasets, Friedman test, and how many data sets they won (winner/total) are also given.

In both of the results obtained from the kNN classifier, the highest winning value was obtained with DP24 version (Table 5, Table 6). According to Friedman test results, it is seen that DP15 has the best value. Although the highest mean value for KNN-AUC is obtained with DP15, the best result for kNN-GMean is obtained with the S-ENN method.

Table 5. The mean AUC values of Sampling Methods in the kNN classifier

Dataset Name	Original	SMOTE	S-TL	S-ENN	Border1	Border2	Safelevel	ADASYN	S-RSB*	DP15	DP24
ecoli0137vs26	0.8500	0.8154	0.8118	0.8136	0.8318	0.8391	0.8172	0.8191	**0.8581**	0.8227	0.8209
shuttle0vs4	**0.9960**	**0.9960**	**0.9960**	**0.9960**	**0.9960**	**0.9960**	0.9951	0.9951	**0.9960**	**0.9960**	0.9957
yeastB1vs7	0.5167	0.7156	0.6943	0.7040	0.6417	0.6545	0.6683	0.7121	0.7086	0.7408	**0.7512**
shuttle2vs4	0.6000	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9673	0.9960	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.5552	0.6686	0.6369	0.6655	0.6893	0.6760	0.6848	0.6493	**0.6940**	0.6810	0.6252
glass016vs5	0.7386	0.9686	0.9157	0.9186	**0.9771**	0.8714	0.9457	0.9600	0.9243	0.9243	0.9743
pageblocks13vs4	0.8076	0.9383	0.9250	**0.9416**	0.9351	0.9351	0.9047	0.9360	0.9106	0.9317	0.9306
yeast05679vs4	0.6667	0.8096	0.8033	0.8138	0.8028	0.8049	0.7974	0.8065	0.7996	**0.8313**	0.8185
yeast1289vs7	0.4995	0.6322	0.6923	**0.6972**	0.6107	0.5481	0.6256	0.6617	0.6557	0.6404	0.5926
yeast1458vs7	0.4992	0.6771	0.6824	0.6354	0.5487	0.5032	0.6279	0.6233	0.6430	**0.6983**	0.6794
yeast2vs4	0.8195	0.8898	**0.9055**	0.8974	0.8915	0.8725	0.8736	0.8969	0.8709	0.8985	0.8996
Ecoli4	0.8484	0.8965	0.8965	0.8997	0.8905	0.8905	0.8870	0.8870	0.9215	0.9060	**0.9326**
Yeast4	0.5741	0.7959	0.8238	0.8087	0.7401	0.7554	0.8224	0.8069	0.8069	**0.8252**	0.7920
Vowel0	0.9772	0.9978	**0.9994**	**0.9994**	**0.9994**	**0.9994**	0.9783	0.9911	0.9961	**0.9994**	**0.9994**
Yeast2vs8	0.7739	0.8295	0.8176	0.8241	0.7502	0.7653	0.7970	**0.8372**	0.8198	0.7872	0.7525
Glass4	0.7808	0.9001	0.9001	0.9051	0.8917	0.8942	0.8704	0.9052	0.9151	**0.9226**	0.9151
Glass5	0.6951	0.9537	0.9585	0.9537	0.8780	0.9256	0.9268	0.9463	**0.9659**	0.8756	0.9207
Glass2	0.4848	0.7948	0.7562	0.7688	0.7001	0.6756	0.7437	**0.8029**	0.7763	0.7174	0.6751
Yeast5	0.8497	0.9663	0.9649	0.9663	0.9510	0.9500	0.9757	0.9653	0.9656	0.9674	**0.9774**
Yeast6	0.7387	0.8736	0.8705	0.8705	0.8566	0.8719	**0.8749**	0.8708	0.8698	0.8367	0.8708
abalone19	0.5000	0.5865	0.5982	0.5817	0.6129	0.5458	0.5939	0.5855	0.5637	0.5734	**0.6184**
abalone918	0.5681	0.7720	0.7675	0.7703	0.7355	0.6616	0.7663	0.7800	**0.8212**	0.7815	0.7941
cleveland0vs4	0.4937	0.5935	0.5621	0.5845	0.6089	0.6247	0.5986	0.5373	0.5499	**0.6916**	0.6847
ecoli01vs235	0.8300	0.8964	0.8468	0.8941	0.8386	0.8386	0.8450	0.8645	0.8223	**0.9036**	0.8873
ecoli01vs5	0.9000	0.9045	0.8977	0.9000	0.8932	0.8886	0.8909	0.8614	0.9045	**0.9159**	**0.9159**
ecoli0146vs5	0.8981	0.9038	0.9000	0.9058	0.8942	0.8923	0.8962	0.8904	0.9019	**0.9173**	0.9154
ecoli0147vs2356	0.8467	0.8712	0.8696	0.8760	0.8821	0.9020	0.8329	0.8598	0.8612	0.9122	**0.9138**
ecoli0147vs56	0.8384	0.8875	0.8711	0.8728	0.9037	**0.9069**	0.8825	0.8744	0.8776	0.8923	0.8955
ecoli0234vs5	0.8944	0.8975	0.8975	0.9031	0.8890	0.8890	0.8920	0.8561	0.9031	**0.9113**	**0.9113**
ecoli0267vs35	0.7875	0.8679	0.8654	0.8654	0.8501	0.8452	0.8079	0.8531	0.8778	0.8728	**0.8852**
ecoli034vs5	0.8750	0.9028	0.8944	0.9028	0.8944	0.8917	0.8750	0.8583	0.8972	0.9139	**0.9167**
ecoli0346vs5	0.8750	0.9061	0.8980	0.9088	**0.9223**	0.9196	0.8845	0.8318	0.9088	0.9142	0.9142
ecoli0347vs56	0.8757	0.8768	0.8768	0.8833	0.9006	0.9006	0.8725	0.8618	0.8769	**0.9055**	0.8677
ecoli046vs5	0.9000	0.9060	0.9032	0.9061	0.8919	0.8891	0.8868	0.8788	0.9060	0.9142	**0.9169**
ecoli067vs35	0.8350	**0.8975**	0.8600	0.8900	0.8300	0.8475	0.8100	0.8800	0.8875	0.8825	0.8700
ecoli067vs5	0.8475	0.8525	0.8575	0.8375	**0.9000**	0.8975	0.8550	0.8675	0.8675	0.8925	0.8600
glass0146vs2	0.5118	**0.7807**	0.7367	0.7166	0.6809	0.6883	0.7464	0.7087	0.7246	0.7326	0.7405
glass015vs2	0.5269	0.7427	0.7298	0.7608	0.6462	0.6202	0.7220	0.7427	0.7382	0.7153	**0.7737**
glass04vs5	0.8500	0.9445	0.9507	0.9570	0.9816	0.9570	0.9154	0.9445	0.9511	**0.9820**	**0.9820**
glass06vs5	0.7450	0.9750	**0.9850**	**0.9850**	0.9350	0.8900	0.9597	0.9697	0.9297	0.9800	**0.9850**
led7digit02456789vs1	0.5393	0.6386	0.6491	**0.8576**	0.5881	0.5856	0.5494	0.6361	0.6230	0.6621	0.6609
yeast0359vs78	0.6390	**0.7671**	0.7406	0.7303	0.7076	0.7007	0.7273	0.7505	0.7283	0.7425	0.7247
yeast0256vs3789	0.7624	0.7781	0.7759	**0.7944**	0.7643	0.7787	0.7698	0.7798	0.7787	0.7894	0.7805
yeast02579vs368	0.9023	**0.9133**	0.9016	0.8916	0.8871	0.8971	0.8933	0.8834	0.9044	0.8947	0.8875
Mean	0.7389	0.8449	0.8383	0.8467	0.8232	0.8156	0.8240	0.8324	0.8387	**0.8476**	0.8460
Friedman Test	7.7614	2.6932	3.7614	2.6932	4.7500	5.0341	5.5341	4.7500	3.2727	**1.6477**	2.1023
Winner / Total	1/44	5/44	5/44	8/44	6/44	4/44	1/44	2/44	6/44	14/44	**15/44**

Table 6. The mean G-Mean values of Sampling Methods in the kNN classifier

Dataset Name	Original	SMOTE	S-TL	S-ENN	Border1	Border2	Safelevel	ADASYN	S-RSB*	DP15	DP24
ecoli0137vs26	0.7414	0.7075	0.7047	0.7059	0.7234	0.7309	0.7098	0.7118	**0.7585**	0.7139	0.7120
shuttle0vs4	**0.9960**	**0.9960**	**0.9960**	**0.9960**	**0.9960**	**0.9960**	0.9951	0.9951	**0.9960**	**0.9960**	0.9957
yeastB1vs7	0.0816	0.7104	0.6883	0.7004	0.5656	0.5751	0.6604	0.7086	0.7009	0.7310	**0.7409**
shuttle2vs4	0.2000	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9661	0.9960	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.2242	0.6526	0.6214	0.6473	0.6507	0.6311	0.6712	0.6298	**0.6719**	0.6477	0.5743
glass016vs5	0.6181	0.9680	0.9099	0.9120	**0.9768**	0.8563	0.9441	0.9589	0.9170	0.9170	0.9739
pageblocks13vs4	0.7833	0.9372	0.9222	**0.9406**	0.9343	0.9344	0.9023	0.9346	0.9091	0.9308	0.9297
yeast05679vs4	0.5732	0.8080	0.8018	0.8124	0.7890	0.7874	0.7941	0.8050	0.7965	**0.8302**	0.8161
yeast1289vs7	0.0000	0.5800	0.6731	**0.6762**	0.4583	0.3196	0.5757	0.6272	0.6271	0.6167	0.5484
yeast1458vs7	0.0000	0.6685	0.6736	0.6220	0.3661	0.1577	0.6159	0.6025	0.6322	**0.6909**	0.6625
yeast2vs4	0.7988	0.8882	**0.9053**	0.8948	0.8876	0.8655	0.8727	0.8962	0.8678	0.8973	0.8982
Ecoli4	0.8182	0.8929	0.8934	0.8959	0.8841	0.8843	0.8843	0.8837	0.9195	0.9023	**0.9296**
Yeast4	0.3451	0.7843	**0.8195**	0.7971	0.6964	0.7199	0.8182	0.7955	0.7998	0.8175	0.7808
Vowel0	0.9766	0.9978	**0.9994**	**0.9994**	**0.9994**	**0.9994**	0.9780	0.9910	0.9961	**0.9994**	**0.9994**
Yeast2vs8	0.7283	0.8149	0.8044	0.8102	0.7117	0.7226	0.7696	**0.8297**	0.8073	0.7579	0.7284
Glass4	0.7342	0.8904	0.8915	0.8968	0.8809	0.8836	0.8635	0.8976	0.9064	**0.9141**	0.9065
Glass5	0.4794	0.9520	0.9573	0.9523	0.7775	0.9194	0.9231	0.9443	**0.9649**	0.8586	0.9115
Glass2	0.0000	0.7876	0.7428	0.7519	0.6819	0.6330	0.7405	**0.7926**	0.7740	0.7046	0.6451
Yeast5	0.8349	0.9658	0.9644	0.9658	0.9503	0.9493	0.9754	0.9648	0.9651	0.9670	**0.9772**
Yeast6	0.6801	0.8686	0.8657	0.8658	0.8413	0.8582	**0.8700**	0.8662	0.8653	0.8300	0.8658
abalone19	0.0000	0.4278	0.5037	0.4254	0.4418	0.2599	0.4317	0.4275	0.3737	0.4488	**0.5330**
abalone918	0.3195	0.7648	0.7583	0.7595	0.6897	0.5638	0.7623	0.7728	**0.8151**	0.7537	0.7768
cleveland0vs4	0.0000	0.5216	0.4888	0.5131	0.5371	0.5488	0.5273	0.4683	0.3792	**0.6034**	0.5887
ecoli01vs235	0.8090	0.8920	0.8342	0.8888	0.8235	0.8235	0.8389	0.8624	0.7998	**0.8964**	0.8766
ecoli01vs5	0.8878	0.9006	0.8940	0.8962	0.8816	0.8773	0.8879	0.8520	0.9009	**0.9114**	**0.9114**
ecoli0146vs5	0.8859	0.8933	0.8898	0.8955	0.8823	0.8806	0.8864	0.8799	0.8913	**0.9071**	0.9052
ecoli0147vs2356	0.8321	0.8660	0.8644	0.8692	0.8751	0.8969	0.8254	0.8547	0.8577	0.9078	**0.9095**
ecoli0147vs56	0.8007	0.8852	0.8692	0.8711	0.9003	**0.9032**	0.8803	0.8729	0.8757	0.8896	0.8927
ecoli0234vs5	0.8789	0.8889	0.8886	0.8939	0.8733	0.8733	0.8835	0.8444	0.8943	**0.9015**	**0.9015**
ecoli0267vs35	0.7558	0.8639	0.8619	0.8614	0.8324	0.8280	0.7944	0.8489	**0.8752**	0.8622	0.8747
ecoli034vs5	0.8560	0.8949	0.8867	0.8941	0.8834	0.8806	0.8663	0.8503	0.8892	0.9050	**0.9074**
ecoli0346vs5	0.8610	0.9014	0.8937	0.9042	**0.9173**	0.9149	0.8793	0.8214	0.9046	0.9094	0.9094
ecoli0347vs56	0.8570	0.8645	0.8645	0.8702	0.8882	0.8884	0.8593	0.8499	0.8652	**0.8929**	0.8558
ecoli046vs5	0.8878	0.8952	0.8924	0.8961	0.8796	0.8772	0.8769	0.8686	0.8965	0.9037	**0.9064**
ecoli067vs35	0.7891	**0.8847**	0.8478	0.8778	0.7968	0.8183	0.7807	0.8673	0.8753	0.8525	0.8408
ecoli067vs5	0.8321	0.8488	0.8540	0.8349	**0.8963**	0.8939	0.8495	0.8630	0.8647	0.8886	0.8554
glass0146vs2	0.1000	**0.7743**	0.7307	0.6977	0.5709	0.6450	0.7359	0.6932	0.7076	0.7175	0.7241
glass015vs2	0.1155	0.7259	0.7165	0.7487	0.5301	0.4923	0.7086	0.7280	0.7223	0.6963	**0.7570**
glass04vs5	0.8243	0.9413	0.9482	0.9547	0.9811	0.9547	0.9080	0.9413	0.9484	**0.9817**	**0.9817**
glass06vs5	0.6207	0.9744	**0.9848**	**0.9848**	0.9277	0.8742	0.9587	0.9691	0.9224	0.9796	**0.9848**
led7digit02456789vs1	0.2170	0.5305	0.5601	**0.8515**	0.3034	0.3025	0.3169	0.5290	0.4994	0.5653	0.5652
yeast0359vs78	0.5298	**0.7638**	0.7391	0.7273	0.6815	0.6676	0.7254	0.7471	0.7258	0.7373	0.7194
yeast0256vs3789	0.7283	0.7744	0.7738	**0.7925**	0.7532	0.7644	0.7674	0.7790	0.7754	0.7850	0.7752
yeast02579vs368	0.8980	**0.9130**	0.9009	0.8904	0.8850	0.8944	0.8928	0.8828	0.9041	0.8939	0.8868
Mean	0.5886	0.8286	0.8246	**0.8328**	0.7819	0.7670	0.8039	0.8160	0.8191	0.8299	0.8281
Friedman Test	7.9886	2.5795	3.5341	2.7159	5.0227	5.4318	5.1364	4.6477	3.0455	**1.7614**	2.1364
Winner / Total	1/44	5/44	6/44	8/44	6/44	4/44	1/44	2/44	7/44	13/44	**14/44**

When the results obtained from the DT classifier are examined, it is seen that the DP24 version is superior in average, Friedman test, and winner/total values (Table 7, Table 8). After DP24 version results, the best average, Friedman test, and winner/total values are obtained with the original DEBOHID (DP15). In these comparisons, the obtained Friedman test and winner/total values affect each other. The success of the DP24 version might have seemed higher if the original DEBOHID version had not been included in the benchmarks. For example, although the ADASYN approach may not have the worst average value in Table 7, it may have the lowest winner/total value.

Table . 7 The mean AUC values of Sampling Methods in the DT classifier

Dataset Name	Original	SMOTE	S-TL	S-ENN	Border1	Border2	Safelevel	ADASYN	S-RSB*	DP15	DP24
ecoli0137vs26	**0.8427**	0.6818	0.7709	0.7781	**0.8427**	0.7409	0.7154	0.5800	0.6336	0.7390	0.8299
shuttle0vs4	**1.0000**	0.9997	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9991	0.9997	0.9997	0.9997	**1.0000**
yeastB1vs7	0.7100	0.5797	0.6123	0.6844	0.6220	0.6101	0.6459	0.6572	0.6681	0.6383	**0.7409**
shuttle2vs4	0.9500	0.9918	0.9960	**1.0000**	0.9500	**1.0000**	0.9298	0.9960	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.5548	0.6226	0.6195	0.6421	0.5290	0.6017	0.5819	0.6367	**0.7345**	0.6071	0.6093
glass016vs5	0.8329	0.8300	0.9329	0.8629	0.8386	**0.9386**	0.8129	0.8686	0.9214	0.8443	0.8943
pageblocks13vs4	0.9955	0.9475	0.9755	0.9565	**0.9978**	0.9600	0.9653	0.9354	0.9630	**0.9978**	0.9755
yeast05679vs4	0.6540	**0.8094**	0.7585	0.7751	0.7019	0.6821	0.7572	0.7104	0.7441	0.7304	0.7450
yeast1289vs7	0.6353	0.6118	0.6348	0.5870	0.5793	0.6096	0.5856	0.6274	0.6026	**0.6389**	0.5783
yeast1458vs7	0.5259	0.5025	0.5117	0.4949	0.5448	0.5061	0.5769	0.5783	0.5518	**0.6087**	0.5314
yeast2vs4	0.8475	0.8428	0.8652	**0.9016**	0.8353	0.8324	0.8759	0.8369	0.8876	0.8347	0.8471
Ecoli4	0.8624	0.8608	0.8592	0.8278	0.8389	0.8203	0.7997	0.8263	**0.8810**	0.8389	0.8310
Yeast4	0.6484	0.6965	**0.7558**	0.6944	0.6754	0.7030	0.7162	0.6712	0.6975	0.7138	0.7308
Vowel0	0.9422	0.9444	**0.9727**	0.9633	0.9039	0.9533	0.9483	0.9655	0.9589	0.9561	0.9528
Yeast2vs8	0.7696	0.7545	0.7773	**0.8066**	0.7402	0.7870	0.7796	0.7100	0.7730	0.7762	0.7882
Glass4	0.8567	0.8984	0.8818	**0.9392**	0.8450	0.9067	0.9051	0.8460	0.8917	0.9350	0.8585
Glass5	0.8427	0.9183	0.9110	0.9207	0.8451	0.7854	0.8037	0.9280	0.8659	0.8976	**0.9476**
Glass2	0.5376	0.6904	0.6060	0.7132	0.6052	0.6058	0.6429	0.6965	**0.8075**	0.7390	0.6448
Yeast5	0.8201	0.8521	0.8847	0.9076	0.8337	0.8309	0.9281	0.8885	0.8632	0.9021	**0.9330**
Yeast6	0.6823	0.7974	0.7936	0.8089	0.7506	0.7649	**0.8242**	0.7705	0.7953	0.7633	0.7686
abalone19	0.4978	0.5513	0.5308	0.5341	0.4888	0.5087	0.5153	0.5358	0.5283	0.5938	**0.6166**
abalone918	0.6911	0.8039	0.7340	0.7151	0.6837	0.7274	0.7623	0.7202	0.7411	0.7508	**0.8254**
cleveland0vs4	0.7888	0.7198	0.7355	0.6833	**0.8906**	0.7416	0.8282	0.6561	0.7489	0.7139	0.8159
ecoli01vs235	0.8114	0.7709	**0.8959**	0.8173	0.8136	0.8632	0.7932	0.7936	0.8045	0.8155	0.8914
ecoli01vs5	0.8636	0.8250	0.8159	0.8795	0.8636	0.8614	0.8750	0.7682	0.8727	0.8864	**0.9045**
ecoli0146vs5	0.7308	0.7981	0.8481	0.8692	0.7904	0.8308	0.7846	0.8731	0.8558	**0.8750**	**0.8750**
ecoli0147vs2356	0.8219	0.8174	0.8642	**0.8674**	0.8071	0.8236	0.8146	0.8293	0.8475	0.8524	0.8588
ecoli0147vs56	0.7886	0.8324	0.7993	0.8122	0.8654	**0.8854**	0.7923	0.8393	0.8392	0.8837	0.8605
ecoli0234vs5	0.7806	0.8834	0.8892	0.8862	0.8584	0.8195	0.8202	0.8724	0.8562	0.8501	**0.8945**
ecoli0267vs35	0.7952	**0.8577**	0.7903	0.8254	0.8078	0.8078	0.7879	0.7754	0.8080	0.8304	0.8080
ecoli034vs5	0.8056	0.8278	0.8667	0.8500	0.7889	0.8111	0.8500	0.8667	**0.8972**	0.8611	0.8694
ecoli0346vs5	0.8392	0.8676	0.8703	0.8730	0.8446	0.8419	0.8568	0.8345	**0.9041**	0.8507	0.8649
ecoli0347vs56	0.7692	0.8476	0.8611	0.8675	0.8470	0.8449	0.8325	0.8454	**0.9077**	0.8834	0.9077
ecoli046vs5	0.8141	0.8119	0.8508	0.8591	0.8336	0.8586	0.8592	**0.8924**	0.8592	0.8782	0.8923
ecoli067vs35	0.8550	0.8175	0.8125	0.8300	0.7850	0.8275	0.8250	0.8100	0.8225	**0.8625**	0.8450
ecoli067vs5	0.7700	0.8775	0.8375	**0.8900**	0.8025	0.8600	0.8650	0.8650	0.8300	0.8275	0.8650
glass0146vs2	0.6120	0.7539	0.7685	0.6757	0.5793	0.5492	0.6751	0.7346	0.7137	**0.8137**	0.7470
glass015vs2	0.5914	0.6309	0.6796	0.7207	0.6933	0.6419	0.6519	0.7022	**0.7304**	0.6785	0.6624
glass04vs5	**0.9941**	0.9401	0.9761	0.9577	**0.9941**	0.9938	0.9632	0.9574	0.9463	**0.9941**	**0.9941**
glass06vs5	0.9350	0.9550	0.9647	0.9597	0.9450	0.9897	0.9287	0.9545	0.9800	**0.9950**	**0.9950**
led7digit02456789vs1	0.8788	0.8869	0.8808	0.8372	0.8955	0.8931	0.8871	0.8958	0.8689	**0.9088**	0.8932
yeast0359vs78	0.6804	0.5998	0.6479	0.6365	0.6072	0.6638	**0.7268**	0.6009	0.6410	0.6534	0.6723
yeast0256vs3789	0.7483	0.7252	0.7402	**0.7718**	0.7159	0.7056	0.7644	0.7394	0.7193	0.7619	0.7355
yeast02579vs368	0.8715	0.8963	0.9090	**0.9168**	0.8696	0.8782	0.8910	0.8918	0.8854	0.8724	0.8932
Mean	0.7783	0.7984	0.8111	0.8136	0.7852	0.7924	0.7987	0.7951	0.8147	0.8194	**0.8271**
Friedman Test	5.8068	4.7614	3.1818	2.5909	5.8409	4.9545	4.6591	4.5795	3.2955	2.5455	**1.7841**
Winner / Total	3/44	2/44	4/44	9/44	5/44	4/44	2/44	1/44	8/44	10/44	**12/44**

Table . 8 The mean G-Mean values of Sampling Methods in the DT classifier

Dataset Name	Original	SMOTE	S-TL	S-ENN	Border1	Border2	Safelevel	ADASYN	S-RSB*	DP15	DP24
ecoli0137vs26	0.7346	0.4731	0.6625	0.6662	**0.7346**	0.5365	0.5193	0.2749	0.3346	0.5314	0.7259
shuttle0vs4	**1.0000**	0.9997	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9991	0.9997	0.9997	0.9997	**1.0000**
yeastB1vs7	0.6602	0.4340	0.5588	0.6366	0.5209	0.4430	0.5984	0.6054	0.6223	0.5826	**0.7268**
shuttle2vs4	0.9414	0.9917	0.9960	**1.0000**	0.9414	**1.0000**	0.9220	0.9960	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.3080	0.5755	0.5754	0.6013	0.3046	0.4527	0.5612	0.5837	**0.7142**	0.4575	0.4834
glass016vs5	0.7306	0.7265	0.9265	0.7826	0.7365	**0.9316**	0.7178	0.8575	0.9131	0.7365	0.7942
pageblocks13vs4	0.9955	0.9466	0.9747	0.9553	**0.9977**	0.9582	0.9648	0.9320	0.9625	**0.9977**	0.9746
yeast05679vs4	0.5622	**0.8030**	0.7461	0.7575	0.6498	0.6122	0.7504	0.6688	0.7243	0.6999	0.7300
yeast1289vs7	0.4672	0.4620	0.5543	0.4244	0.4332	0.4648	0.5069	0.5348	0.5091	**0.5815**	0.4658
yeast1458vs7	0.2397	0.2326	0.3351	0.1863	0.3213	0.0810	0.4637	0.4294	0.3193	**0.5304**	0.3252
yeast2vs4	0.8332	0.8387	0.8600	**0.8998**	0.8213	0.8183	0.8752	0.8290	0.8838	0.8272	0.8411
Ecoli4	0.8347	0.8480	0.8414	0.8105	0.8192	0.7988	0.7658	0.7999	**0.8659**	0.8189	0.8130
Yeast4	0.5336	0.6470	**0.7314**	0.6401	0.5961	0.6424	0.6687	0.6084	0.6318	0.6663	0.6993
Vowel0	0.9393	0.9434	**0.9726**	0.9618	0.8962	0.9525	0.9472	0.9653	0.9586	0.9548	0.9509
Yeast2vs8	0.7253	0.7083	0.7365	**0.7727**	0.6911	0.7551	0.7460	0.5939	0.7352	0.7371	0.7546
Glass4	0.8230	0.8937	0.8714	**0.9372**	0.8173	0.8951	0.8968	0.8241	0.8815	0.9263	0.8473
Glass5	0.7340	0.9096	0.9006	0.9123	0.7372	0.5851	0.7007	0.9190	0.7698	0.7975	**0.9390**
Glass2	0.2510	0.6651	0.5594	0.6973	0.3870	0.4595	0.6129	0.6715	**0.7936**	0.7226	0.5932
Yeast5	0.7966	0.8416	0.8797	0.9036	0.8178	0.8141	0.9271	0.8822	0.8554	0.8969	**0.9304**
Yeast6	0.5877	0.7625	0.7592	0.7841	0.6984	0.7144	**0.7974**	0.7273	0.7723	0.7265	0.7344
abalone19	0.0000	0.2741	0.1878	0.1940	0.0000	0.0749	0.1544	0.1943	0.2291	0.4786	**0.5182**
abalone918	0.6011	0.7941	0.7082	0.6840	0.5854	0.6770	0.7474	0.6896	0.7103	0.7312	**0.8171**
cleveland0vs4	0.7602	0.6896	0.7013	0.6425	**0.8706**	0.7047	0.8231	0.5347	0.7115	0.6855	0.8020
ecoli01vs235	0.7774	0.7393	**0.8934**	0.7914	0.7762	0.8480	0.7822	0.7779	0.7928	0.7882	0.8836
ecoli01vs5	0.8362	0.8077	0.7996	0.8679	0.8448	0.8425	0.8660	0.7240	0.8674	0.8743	**0.8949**
ecoli0146vs5	0.6017	0.7589	0.8156	0.8428	0.7487	0.8129	0.7532	**0.8648**	0.8388	0.8493	0.8647
ecoli0147vs2356	0.7958	0.8036	0.8583	**0.8616**	0.7889	0.7886	0.8031	0.8084	0.8380	0.8449	0.8507
ecoli0147vs56	0.7547	0.8231	0.7824	0.7986	0.8547	**0.8787**	0.7722	0.8320	0.8301	0.8765	0.8468
ecoli0234vs5	0.7550	0.8737	0.8815	0.8757	0.8307	0.8037	0.8073	0.8628	0.8335	0.8343	**0.8863**
ecoli0267vs35	0.7675	**0.8473**	0.7693	0.8116	0.7848	0.7850	0.7765	0.7579	0.7939	0.8142	0.7885
ecoli034vs5	0.7652	0.8138	0.8528	0.8346	0.7468	0.7768	0.8332	0.8569	**0.8889**	0.8513	0.8589
ecoli0346vs5	0.8206	0.8614	0.8650	0.8669	0.8250	0.8125	0.8534	0.8169	**0.8995**	0.8389	0.8602
ecoli0347vs56	0.7331	0.8162	0.8305	0.8351	0.8143	0.8089	0.8029	0.8381	**0.9056**	0.8784	0.9024
ecoli046vs5	0.7843	0.7804	0.8431	0.8497	0.8056	0.8423	0.8354	0.8814	0.8486	0.8672	**0.8828**
ecoli067vs35	0.7505	0.7831	0.7795	0.8009	0.7257	0.7957	0.7945	0.7731	0.7894	**0.8307**	0.8123
ecoli067vs5	0.6442	0.8746	0.8285	**0.8853**	0.7012	0.8472	0.8623	0.8544	0.8146	0.7984	0.8600
glass0146vs2	0.3904	0.7398	0.7530	0.6470	0.3587	0.3142	0.6529	0.7273	0.6817	**0.7922**	0.6540
glass015vs2	0.3774	0.4629	0.5868	0.6928	0.6493	0.5209	0.5521	0.6579	**0.6960**	0.5535	0.6143
glass04vs5	**0.9940**	0.9371	0.9755	0.9565	**0.9940**	0.9936	0.9620	0.9558	0.9445	**0.9940**	**0.9940**
glass06vs5	0.9261	0.9517	0.9628	0.9576	0.9364	0.9896	0.9253	0.9527	0.9795	**0.9949**	**0.9949**
led7digit02456789vs1	0.8704	0.8801	0.8746	0.8361	0.8872	0.8847	0.8856	0.8911	0.8620	**0.9057**	0.8892
yeast0359vs78	0.6258	0.5322	0.6243	0.5899	0.5209	0.5951	**0.7136**	0.5396	0.6006	0.6163	0.6463
yeast0256vs3789	0.7134	0.7014	0.7264	**0.7614**	0.6809	0.6603	0.7578	0.7261	0.6929	0.7506	0.7242
yeast02579vs368	0.8650	0.8936	0.9070	**0.9158**	0.8633	0.8716	0.8891	0.8890	0.8817	0.8676	0.8903
Mean	0.6956	0.7532	0.7784	0.7757	0.7163	0.7237	0.7624	0.7525	0.7768	0.7843	**0.7969**
Friedman Test	6.4432	4.7159	3.0000	2.6023	6.1250	5.3295	4.2273	4.2955	3.0568	2.5568	**1.6477**
Winner / Total	2/44	2/44	4/44	9/44	5/44	4/44	2/44	1/44	8/44	9/44	**12/44**

For SVM-AUC, the DP24 version obtained the highest mean and ranking value, while the highest winner total value is obtained by the Border1 method (Table 9).

When the SVM-Gmean results are examined, it will be seen that the best average value, the best-ranking value, and the highest total winner value is obtained by DP24 version, DP15 version, and Border1, respectively (Table 10).

According to the results from Table 5 - Table 10, the worst winner/total value of the DP24 version was obtained in the SVM classifier and G-Mean metric. The best winner/total value for the DP24 version is obtained from the kNN classifier and the AUC metric.

Table . 9 The mean AUC values of Sampling Methods in the SVM classifier

Dataset Name	Original	SMOTE	S-TL	S-ENN	Border1	Border2	Safelevel	ADASYN	S-RSB*	DP15	DP24
ecoli0137vs26	**0.8500**	0.7935	0.7935	0.7971	0.8263	0.8244	0.8327	0.7917	0.8398	0.8172	0.8191
shuttle0vs4	**1.0000**	**1.0000**	0.9960	**1.0000**	**1.0000**	**1.0000**	0.9991	0.9994	0.9997	**1.0000**	0.9960
yeastB1vs7	0.5000	0.7636	0.7543	0.7419	0.6718	0.6707	**0.7733**	0.7651	0.7605	0.7384	0.7264
shuttle2vs4	**1.0000**	0.9793	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9470	0.9795	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.5000	0.5171	0.5407	0.5352	0.6281	0.5979	0.5138	0.5221	**0.6293**	0.6157	0.6100
glass016vs5	0.4971	0.9486	0.9486	0.9514	**0.9800**	0.9629	0.9314	0.9514	0.9486	0.9686	0.9600
pageblocks13vs4	0.4728	0.4090	0.4840	0.2948	0.4604	0.4606	0.3069	0.3565	0.3934	**0.6004**	0.5690
yeast05679vs4	0.5000	0.7865	0.7912	0.7854	0.7943	**0.7953**	0.7944	0.7829	0.7844	0.7838	0.7827
yeast1289vs7	0.5000	0.7189	**0.7263**	0.6910	0.6762	0.6757	0.7064	0.7123	0.6946	0.6921	0.7014
yeast1458vs7	0.5000	0.6343	0.6146	0.6357	0.6308	0.6081	0.6328	0.6146	**0.6388**	0.6333	0.6363
yeast2vs4	0.6691	**0.8964**	0.8953	0.8907	0.8896	0.8874	0.8885	0.8677	0.8864	0.8805	0.8885
Ecoli4	0.5750	**0.9716**	0.9402	0.9668	0.9342	0.9295	0.9541	0.9102	0.9620	0.9576	0.9560
Yeast4	0.5000	**0.8434**	0.8265	0.8338	0.8251	0.8223	0.8286	0.8212	0.8131	0.8104	0.8190
Vowel0	0.8950	0.9699	0.9700	**0.9705**	0.9200	0.9244	0.9505	0.9616	0.9649	0.9683	0.9600
Yeast2vs8	**0.7739**	0.7664	0.7653	0.7664	0.7065	0.7141	**0.7739**	0.7394	0.7631	0.7718	0.7696
Glass4	0.5592	0.9101	0.8977	0.9027	**0.9226**	0.9176	0.8704	0.9002	0.9002	0.9101	0.9077
Glass5	0.5000	0.9366	0.9366	0.9439	**0.9732**	0.9561	0.9366	0.9463	0.9415	0.9683	0.9659
Glass2	0.5000	0.6155	0.6777	0.6309	0.6050	0.5880	0.6546	**0.6803**	0.6212	0.6085	0.6283
Yeast5	0.5000	0.9635	0.9622	0.9642	**0.9667**	0.9660	0.9608	0.9608	0.9625	0.9635	0.9635
Yeast6	0.5000	0.8730	0.8723	0.8737	0.8791	0.8777	0.8730	0.8628	**0.8870**	0.8820	0.8816
abalone19	0.5000	0.7453	0.7601	0.7623	0.6914	0.6963	0.7821	0.7786	0.7826	**0.7894**	0.7705
abalone918	0.5000	0.8581	0.8545	0.8493	**0.8833**	0.8760	0.8174	0.8530	0.8727	0.8761	0.8776
cleveland0vs4	0.7478	0.9167	0.9167	0.9166	0.7947	0.8426	0.8950	**0.9314**	0.8735	0.9260	0.8927
ecoli01vs235	0.8359	0.8777	0.8732	0.8845	0.8468	0.8673	0.8555	0.8255	0.8732	**0.8918**	**0.8918**
ecoli01vs5	0.8364	0.8591	0.8341	0.8614	**0.8864**	0.8568	0.8614	0.8068	0.8364	0.8750	0.8750
ecoli0146vs5	0.8635	0.8885	0.8712	0.8519	0.8808	0.8769	0.8769	0.8423	0.8827	0.8981	**0.9019**
ecoli0147vs2356	0.8267	0.8812	0.8645	0.8661	0.8555	0.8539	0.8796	0.8228	**0.8929**	0.8658	0.8658
ecoli0147vs56	0.8719	0.8812	0.8730	0.8530	0.8821	**0.9021**	0.9012	0.8054	0.8779	0.8775	0.8759
ecoli0234vs5	0.8667	0.8891	0.8810	0.8946	0.8806	**0.9029**	0.8920	0.8204	0.8865	0.9002	0.8946
ecoli0267vs35	0.8526	0.8304	0.8108	0.8604	0.8353	0.8155	0.8507	0.8137	**0.8883**	0.8379	0.8628
ecoli034vs5	0.8611	0.8611	0.8639	0.8639	**0.9333**	0.9000	0.8611	0.8111	0.8611	0.8722	0.8778
ecoli0346vs5	0.8696	0.8899	0.8926	0.8676	0.8588	0.8588	0.8899	0.7993	0.8872	**0.9088**	**0.9088**
ecoli0347vs56	0.8935	0.8947	0.8883	0.8925	0.9007	0.8985	0.9040	0.8651	0.9019	**0.9099**	0.9055
ecoli046vs5	0.8696	0.8896	0.8843	0.8897	0.8892	0.8809	0.8951	0.8291	0.8788	0.9032	**0.9087**
ecoli067vs35	0.8525	0.8425	0.8650	0.8525	0.8350	0.8000	0.8400	0.8000	0.8325	0.8700	**0.8875**
ecoli067vs5	0.8425	0.8525	0.8350	0.8400	**0.8825**	0.8475	0.8650	0.7800	0.8475	0.8775	0.8750
glass0146vs2	0.5000	0.6067	0.6237	0.6227	0.6277	**0.6785**	0.6013	0.6174	0.6306	0.6052	0.6305
glass015vs2	0.5000	0.5126	0.5094	**0.5320**	0.4927	0.5261	0.5191	0.4868	0.5132	0.5000	0.5202
glass04vs5	0.8500	0.9445	0.9449	0.9449	**0.9816**	0.9570	0.9570	0.9507	0.9691	**0.9816**	**0.9816**
glass06vs5	0.6500	0.9387	0.9437	0.9387	**0.9689**	0.9087	0.9387	0.9387	0.9387	0.9589	0.9642
led7digit02456789vs1	**0.9056**	0.8863	0.8969	0.8760	0.8789	0.8770	0.8883	0.8745	0.8839	0.8748	0.8760
yeast0359vs78	0.6067	0.7484	0.7254	0.7417	0.7482	0.7450	**0.7506**	0.7025	0.7462	0.7439	0.7392
yeast0256vs3789	0.5486	0.7957	0.7913	**0.8007**	0.7929	0.7923	0.7940	0.7717	0.7951	0.7907	0.7946
yeast02579vs368	0.8006	0.9007	0.9041	**0.9107**	**0.9107**	0.9063	0.9007	0.8638	0.9057	0.9057	0.9096
Mean	0.6942	0.8293	0.8296	0.8261	0.8279	0.8238	0.8260	0.8072	0.8329	0.8412	**0.8416**
Friedman Test	6.9659	3.4205	4.4318	3.4432	3.2273	4.0795	3.8182	6.5341	3.3523	2.4432	**2.2841**
Winner / Total	5/44	4/44	2/44	6/44	**13/44**	6/44	3/44	2/44	6/44	8/44	7/44

Table. 10 The mean G-Mean values of Sampling Methods in the SVM classifier

Dataset Name	Original	SMOTE	S-TL	S-ENN	Border1	Border2	Safelevel	ADASYN	S-RSB*	DP15	DP24
ecoli0137vs26	0.7414	0.6878	0.6878	0.6925	0.7189	0.7188	0.7383	0.6857	**0.7431**	0.7083	0.7102
shuttle0vs4	**1.0000**	**1.0000**	0.9960	**1.0000**	**1.0000**	**1.0000**	0.9991	0.9994	0.9997	**1.0000**	0.9960
yeastB1vs7	0.0000	0.7608	0.7508	0.7400	0.6079	0.6074	**0.7691**	0.7603	0.7544	0.7368	0.7251
shuttle2vs4	**1.0000**	0.9785	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9440	0.9789	**1.0000**	**1.0000**	**1.0000**
glass016vs2	0.0000	0.2911	0.4529	0.3110	0.5151	**0.5793**	0.4363	0.3621	0.5617	0.4750	0.4690
glass016vs5	0.0000	0.9471	0.9470	0.9501	**0.9797**	0.9618	0.9288	0.9501	0.9471	0.9680	0.9589
pageblocks13vs4	0.3674	0.2984	0.3675	0.1667	0.3542	0.3514	0.2532	0.2607	0.2772	**0.4689**	0.4305
yeast05679vs4	0.0000	0.7815	0.7881	0.7804	0.7867	0.7879	**0.7914**	0.7796	0.7791	0.7775	0.7769
yeast1289vs7	0.0000	0.7156	**0.7214**	0.6851	0.6429	0.6418	0.7031	0.7090	0.6914	0.6865	0.6970
yeast1458vs7	0.0000	0.6250	0.6033	**0.6300**	0.6063	0.5688	0.6240	0.6028	0.6298	0.6227	0.6252
yeast2vs4	0.5464	**0.8954**	0.8943	0.8890	0.8878	0.8858	0.8870	0.8666	0.8849	0.8783	0.8869
Ecoli4	0.3000	**0.9711**	0.9384	0.9662	0.9308	0.9262	0.9528	0.9075	0.9611	0.9558	0.9542
Yeast4	0.0000	**0.8430**	0.8255	0.8325	0.8204	0.8179	0.8274	0.8203	0.8112	0.8071	0.8166
Vowel0	0.8876	0.9696	0.9696	**0.9701**	0.9170	0.9209	0.9498	0.9607	0.9645	0.9680	0.9595
Yeast2vs8	0.7283	0.7219	0.7209	0.7219	0.6547	0.6693	0.7283	**0.7291**	0.7196	0.7266	0.7247
Glass4	0.2309	0.9020	0.8896	0.8949	**0.9142**	0.9090	0.8627	0.8932	0.8923	0.9019	0.9003
Glass5	0.0000	0.9342	0.9342	0.9421	**0.9724**	0.9549	0.9342	0.9446	0.9394	0.9673	0.9647
Glass2	0.0000	0.5502	0.5929	0.5610	0.5117	0.5041	0.5906	**0.5995**	0.4884	0.4708	0.5566
Yeast5	0.0000	0.9628	0.9614	0.9636	**0.9661**	0.9654	0.9600	0.9599	0.9618	0.9628	0.9628
Yeast6	0.0000	0.8703	0.8696	0.8709	0.8729	0.8715	0.8704	0.8593	**0.8852**	0.8791	0.8786
abalone19	0.0000	0.7399	0.7572	0.7592	0.6557	0.5926	0.7781	0.7759	0.7784	**0.7868**	0.7674
abalone918	0.0000	0.8567	0.8529	0.8476	**0.8786**	0.8720	0.8137	0.8514	0.8712	0.8740	0.8753
cleveland0vs4	0.6159	0.9125	0.9124	0.9130	0.7541	0.8315	0.8902	**0.9285**	0.8660	0.9221	0.8750
ecoli01vs235	0.7967	0.8690	0.8628	0.8762	0.8176	0.8546	0.8452	0.7999	0.8628	**0.8826**	0.8819
ecoli01vs5	0.8185	0.8496	0.8198	0.8517	**0.8757**	0.8403	0.8518	0.7952	0.8222	0.8649	0.8649
ecoli0146vs5	0.8360	0.8785	0.8618	0.8424	0.8555	0.8508	0.8676	0.8373	0.8737	0.8892	**0.8926**
ecoli0147vs2356	0.8076	0.8779	0.8613	0.8628	0.8492	0.8474	0.8765	0.8145	**0.8897**	0.8604	0.8601
ecoli0147vs56	0.8607	0.8778	0.8709	0.8503	0.8754	0.8958	**0.8980**	0.8019	0.8757	0.8709	0.8695
ecoli0234vs5	0.8543	0.8811	0.8736	0.8859	0.8701	**0.8942**	0.8841	0.8125	0.8784	0.8913	0.8859
ecoli0267vs35	0.8328	0.8153	0.7996	0.8518	0.8175	0.8005	0.8456	0.8050	**0.8863**	0.8213	0.8522
ecoli034vs5	0.8453	0.8372	0.8402	0.8396	**0.9310**	0.8915	0.8376	0.8047	0.8372	0.8483	0.8525
ecoli0346vs5	0.8560	0.8856	0.8880	0.8616	0.8473	0.8473	0.8854	0.7904	0.8832	**0.9046**	0.9042
ecoli0347vs56	0.8860	0.8877	0.8818	0.8864	0.8945	0.8928	0.8982	0.8566	0.8966	**0.9040**	0.8995
ecoli046vs5	0.8564	0.8814	0.8751	0.8809	0.8785	0.8712	0.8868	0.8201	0.8701	0.8949	**0.8998**
ecoli067vs35	0.7479	0.7437	0.8339	0.7536	0.7326	0.7006	0.7426	0.7040	0.8017	0.8397	**0.8577**
ecoli067vs5	0.8131	0.8493	0.8325	0.8374	**0.8768**	0.8357	0.8629	0.7715	0.8441	0.8724	0.8701
glass0146vs2	0.0000	0.5274	0.5321	0.5545	0.6034	**0.6557**	0.5180	0.5428	0.5641	0.5576	0.5686
glass015vs2	0.0000	0.3902	0.4049	0.3991	0.4778	**0.5100**	0.3887	0.3866	0.3391	0.4075	0.4804
glass04vs5	0.8243	0.9415	0.9422	0.9422	**0.9811**	0.9554	0.9552	0.9484	0.9680	**0.9811**	**0.9811**
glass06vs5	0.4243	0.9357	0.9410	0.9357	**0.9674**	0.8993	0.9357	0.9357	0.9357	0.9572	0.9630
led7digit02456789vs1	**0.9002**	0.8852	0.8947	0.8709	0.8725	0.8685	0.8857	0.8730	0.8817	0.8701	0.8715
yeast0359vs78	0.4595	0.7432	0.7192	0.7354	0.7415	0.7390	**0.7448**	0.6934	0.7411	0.7368	0.7316
yeast0256vs3789	0.3025	0.7882	0.7841	**0.7935**	0.7844	0.7838	0.7866	0.7692	0.7875	0.7827	0.7871
yeast02579vs368	0.7722	0.8991	0.9028	**0.9098**	0.9095	0.9051	0.8988	0.8627	0.9044	0.9044	0.9087
Mean	0.4753	0.8059	0.8104	0.8025	0.8047	0.8018	0.8075	0.7866	0.8125	0.8201	**0.8226**
Friedman Test	7.2614	3.4318	4.3068	3.5000	3.1250	4.2614	3.7386	6.0568	3.5114	**2.3977**	2.4091
Winner / Total	3/44	4/44	2/44	6/44	**12/44**	6/44	4/44	3/44	5/44	8/44	5/44

6. Conclusions

DEBOHID is an oversampling method for generating synthetic data in imbalanced datasets. DEBOHID uses neighbors near its minority class members while creating new samples. In the procedure of generating new synthetic samples, DEBOHID exploits the crossover and mutation processes of the DE meta-heuristic algorithm. In crossover and mutation processes, the CR and F parameters, respectively, affect the solution quality in the production of new synthetic samples. When DEBOHID is proposed, the CR and F parameters are determined as 0.6 and 0.3, respectively. In this study, the performance of 90 different DEBOHID versions in 44 highly imbalanced datasets was tested by changing the CR and F parameters. In experimental studies, the best rank value was obtained with DP24 version according to Friedman test. DP24 version has been compared with both the sampling well-known methods in the literature and the proposed DEBOHID (DP15, CR (0.6) and F (0.3)). When the results were evaluated, it was seen that DP24 version can produce better results for both DP15 version and other sampling methods.

Acknowledgments

The author wishes to thank the Scientific Research Projects

Coordinatorship at Selçuk University and The Scientific and Technological Research Council of Turkey for their institutional supports. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

1. Jiang, N. and N. Li, A wind turbine frequent principal fault detection and localization approach with imbalanced data using an improved synthetic oversampling technique. International Journal of Electrical Power & Energy Systems, 2021. **126**: p. 106595.

2. Vandewiele, G., et al., *Overly optimistic prediction results on imbalanced data: a case study of flaws and benefits when applying over-sampling.* Artificial Intelligence in Medicine, 2021. **111**: p. 101987.

3. Zeng, N., et al., *A new switching-delayed-PSO-based optimized SVM algorithm for diagnosis of Alzheimer's disease.* Neurocomputing, 2018. **320**: p. 195-202.

4. Zeng, N., et al., *An improved particle filter with a novel hybrid proposal distribution for quantitative analysis of gold immunochromatographic strips.* IEEE Transactions on Nanotechnology, 2019. **18**: p. 819-829.

5. Gao, X., et al., *Adaptive weighted imbalance learning with application to abnormal activity recognition.* Neurocomputing, 2016. **173**: p. 1927-1935.

6. Zakaryazad, A. and E. Duman, *A profit-driven Artificial Neural Network (ANN) with applications to fraud detection and direct marketing.* Neurocomputing, 2016. **175**: p. 121-131.

7. Kaur, H., H.S. Pannu, and A.K. Malhi, *A systematic review on imbalanced data challenges in machine learning: Applications and solutions.* ACM Computing Surveys (CSUR), 2019. **52**(4): p. 1-36.

8. Chawla, N.V., et al., *SMOTE: synthetic minority over-sampling technique.* Journal of artificial intelligence research, 2002. **16**: p. 321-357.

9. Han, H., 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*. 2005. Springer.

10. Bunkhumpornpat, C., 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*. 2009. Springer.

11. He, H., et al. *ADASYN: Adaptive synthetic sampling approach for imbalanced learning*. in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. 2008. IEEE.

12. Batista, G.E., 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, 2004. **6**(1): p. 20-29.

13. Ramentol, E., et al., *SMOTE-RSB*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory.* Knowledge and information systems, 2012. **33**(2): p. 245-265.

14. Tomek, I., *Two modifications of CNN.* 1976.

15. Pawlak, Z., *Rough sets.* International journal of computer & information sciences, 1982. **11**(5): p. 341-356.

16. Garcia, S., et al., *Evolutionary-based selection of generalized instances for imbalanced classification.* Knowledge-Based Systems, 2012. **25**(1): p. 3-12.

17. Yang, P., et al. *A particle swarm based hybrid system for imbalanced medical data sampling*. in *BMC genomics*. 2009. Springer.

18. Wong, G.Y., F.H. Leung, and S.-H. Ling. *A novel evolutionary preprocessing method based on over-sampling and under-sampling for imbalanced datasets*. in *iecon 2013-39th Annual Conference of the ieee Industrial Electronics Society*. 2013. IEEE.

19. Eshelman, L.J., *The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination*, in *Foundations of genetic algorithms*. 1991, Elsevier. p. 265-283.

20. Yu, H., J. Ni, and J. Zhao, *ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data.* Neurocomputing, 2013. **101**: p. 309-318.

21. Braytee, A., et al. *ABC-sampling for balancing imbalanced datasets based on artificial bee colony algorithm*. in *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*. 2015. IEEE.

22. Price, K.V., R.M. Storn, and J.A. Lampinen, *The differential evolution algorithm.* Differential evolution: a practical approach to global optimization, 2005: p. 37-134.

23. Kaya, E., et al., *DEBOHID: A differential evolution based oversampling approach for highly imbalanced datasets.* Expert Systems with Applications, 2020: p. 114482.

24. Alcalá-Fdez, J., et al., *KEEL: a software tool to assess evolutionary algorithms for data mining problems.* Soft Computing, 2009. **13**(3): p. 307-318.

25. Haixiang, G., et al., *Learning from class-imbalanced data: Review of methods and applications.* Expert Systems with Applications, 2017. **73**: p. 220-239.

26. Asuncion, A. and D. Newman, *UCI machine learning repository*. 2007, Irvine, CA, USA.

27. Alcalá-Fdez, J., et al., *Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework.* Journal of Multiple-Valued Logic & Soft Computing, 2011.