ULUSLARARASI 3B YAZICI TEKNOLOJİLERİ
VE DİJİTAL ENDÜSTRİ DERGİSİ
INTERNATIONAL JOURNAL OF 3D PRINTING
TECHNOLOGIES AND DIGITAL INDUSTRY

ISSN:2602-3350 (Online)
URL: https://dergipark.org.tr/ij3dptdi

BODE: A BOUNDARY AWARE DIFFERENTIAL EVOLUTION BASED HYBRID OVERSAMPLING TECHNIQUE FOR IMBALANCED DATA CLASSIFICATION

Yazarlar (Authors): Muhammed Abdulhamid Karabiyik*

Bu makaleye şu şekilde atıfta bulunabilirsiniz (To cite to this article): *Karabiyik M. A., "Bode: A Boundary Aware Differential Evolution Based Hybrid Oversampling Technique for Imbalanced Data Classification" Int. J. of 3D Printing Tech. Dig. Ind., 9(2): 320-330, (2025).*

DOI: 10.46519/ij3dptdi.1703106

Araştırma Makale/ Research Article

Erişim Linki: (To link to this article): https://dergipark.org.tr/en/pub/ij3dptdi/archive

BODE: A BOUNDARY AWARE DIFFERENTIAL EVOLUTION BASED HYBRID OVERSAMPLING TECHNIQUE FOR IMBALANCED DATA CLASSIFICATION

Muhammed Abdulhamid Karabiyik[a]*

[a]Niğde Ömer Halidemir University, Bor Vocational School, Computer Techologies Department, TURKEY

* *Corresponding Author: abdulhamidkarabiyik@ohu.edu.tr*

(**Received:** 20.05.25; **Revised:** 11.07.25; **Accepted:** 27.07.25)

ABSTRACT

Class imbalance presents a persistent challenge in supervised learning, often degrading classifier performance on underrepresented classes. This study introduces BODE, a hybrid oversampling method that combines boundary-aware instance selection, differential evolution-based perturbation, and density-constrained filtering. By targeting critical minority instances near decision boundaries, BODE generates diverse yet structurally valid synthetic samples. Experiments on 44 benchmark datasets using k-NN, Decision Tree, and SVM classifiers demonstrate that BODE consistently outperforms eleven widely used oversampling methods. Evaluated solely using the AUC metric, BODE achieves the highest average performance across all classifiers, with 28, 33, and 26 dataset-level wins, respectively. These results confirm BODE's robustness and generalization capability, particularly in challenging scenarios involving overlapping or sparse decision regions.

Keywords: Imbalanced Learning, Oversampling, Differential Evolution, Decision Boundary.

1. INTRODUCTION

In many real-world classification problems, the distribution of classes is highly imbalanced, with the minority class significantly underrepresented [1]. This imbalance often leads to biased models that perform well on the majority class while failing to detect rare but critical minority instances. Such failures are especially detrimental in domains like medical diagnosis, fraud detection, and fault prediction, where accurate identification of minority cases is essential for minimizing risk and ensuring reliable decision-making [2–4].

To mitigate the impact of class imbalance, various oversampling techniques have been proposed, with SMOTE and its variants being among the most widely adopted. These methods generate synthetic samples to augment the minority class, aiming to balance the dataset without discarding valuable majority information [5]. However, most interpolation-based approaches fail to preserve the decision boundaries and often generate overlapping or noisy samples, which may degrade classifier performance rather than improve it.

Recent studies have explored more sophisticated strategies, such as boundary-aware sampling and evolutionary algorithms, to improve synthetic sample quality. Boundary-focused methods aim to reinforce critical regions near decision boundaries, while Differential Evolution (DE) offers a robust framework for generating diverse and informative samples. Yet, these approaches are typically applied in isolation, and the lack of integration between boundary sensitivity and adaptive generation limits their overall effectiveness [6–9].

To overcome these limitations, BODE introduces a hybrid oversampling strategy that integrates boundary-aware instance selection with Differential Evolution-based synthetic sample generation. The method begins by detecting minority instances near decision boundaries through a k-nearest neighbors analysis. Controlled perturbations are then applied to these critical points using evolutionary operations, generating synthetic samples that are both diverse and well-positioned. This design enhances the distinction

between classes while reducing the risk of noise and redundancy.

The effectiveness of BODE has been validated through extensive experiments conducted on 44 benchmark datasets from the KEEL repository. Performance was evaluated using the Area Under the ROC Curve (AUC) metric, which is well-suited for imbalanced classification tasks. Across multiple classifiers, including k-Nearest Neighbors, Decision Trees, and Support Vector Machines, BODE consistently outperformed a range of established oversampling techniques, demonstrating its robustness and superior boundary modeling capability.

2. RELATED WORK

Synthetic Minority Oversampling Technique (SMOTE) and its variants have been widely used to address class imbalance by generating new minority class samples through interpolation. Extensions such as Borderline-SMOTE, Safe-Level-SMOTE, and ADASYN aim to refine sample generation by focusing on boundary regions, local density, or instance difficulty [10–12]. However, these methods often suffer from overlapping synthetic instances and fail to adequately capture complex decision boundaries. To address such limitations, filtering-based methods like SMOTE-ENN and SMOTE-TomekLinks have been proposed, yet they still rely on static interpolation rules that limit adaptability in highly imbalanced or noisy datasets [13-14].

To improve flexibility and diversity in synthetic sample generation, evolutionary algorithms such as DE have been incorporated into oversampling frameworks. DE-based methods like DEBOHID have shown promise by exploring the feature space in an adaptive manner [9]. Other hybrid approaches, including swarm intelligence, genetic algorithms, and cluster-guided sampling, attempt to balance diversity and representativeness. However, most of these methods either ignore decision boundary awareness or lack mechanisms to constrain synthetic instances within meaningful regions. This creates a gap for methods that integrate evolutionary generation with explicit boundary sensitivity — a gap that BODE aims to fill.

3. PROPOSED METHOD

Imbalanced datasets often suffer from insufficient representation of minority class patterns, especially in regions close to decision boundaries. Conventional oversampling methods generate synthetic samples based on uniform strategies that disregard local data geometry or class overlap. This may lead to the creation of redundant or misleading samples that fail to improve classification performance.

To address these challenges, the proposed method BODE combines two complementary strategies: (1) the use of DE to generate diverse synthetic samples, and (2) the application of a density-aware boundary constraint to ensure the reliability of those samples. This hybrid approach prioritizes borderline regions while filtering out unsafe or noisy generations. The core components of BODE are presented in the following subsections.

3.1. Synthetic Sample Generation Via Differential Evolution

To generate diverse and informative synthetic samples, BODE employs DE, a population-based optimization algorithm. For each selected borderline minority instance x_{target} three distinct minority samples x_{r1}, x_{r2}, x_{r3} are randomly selected to construct a mutant vector using the DE/rand/1 strategy, as defined in Equation 1:

$$v = x_{r1} + F \cdot (x_{r2} - x_{r3}) \quad (1)$$

Here, $F \in (0,1)$ is a mutation factor that controls the amplification of the differential variation [15]. The resulting mutant vector v is then combined with the target instance x_{target} using a binomial crossover to produce a trial vector uuu, as shown in Equation 2:

$$x_{trial}(j) = \begin{cases} v[j] & if \ \ rand_j < CR \\ x_i[j] & otherwise \end{cases} \quad (2)$$

In this expression, CR represents the crossover rate, and $rand_j$ is a uniformly distributed random number generated independently for each feature dimension j. The resulting trial vector v serves as a candidate synthetic sample. This mechanism introduces controlled diversity while preserving the structural coherence of the minority class [16].

In our implementation, we used a mutation factor F of 0.5 and a crossover rate CR of 0.9, which are commonly adopted values for maintaining a balance between exploration and exploitation in the evolutionary search process [Reference]. The population size was set to 50, ensuring sufficient diversity without introducing excessive computational cost. These parameter settings were kept constant across all datasets to maintain consistency and reproducibility of the results. Sensitivity analyses confirmed that these values provided stable and robust performance across various data distributions[17].

3.2. Step-by-Step Execution Of BODE

The BODE algorithm operates through a sequence of procedures that ensure synthetic samples are generated in informative and safe regions of the minority class distribution.

The process begins with identifying the structural center of the minority class. Kernel Density Estimation (KDE) is applied to approximate the densest region of minority samples [18]. Based on this estimation, a circular boundary is drawn around the computed center to define a safe generation area. This approach helps prevent sample creation in sparse or unreliable regions. The overall structure and the defined boundary are illustrated in Figure 1.

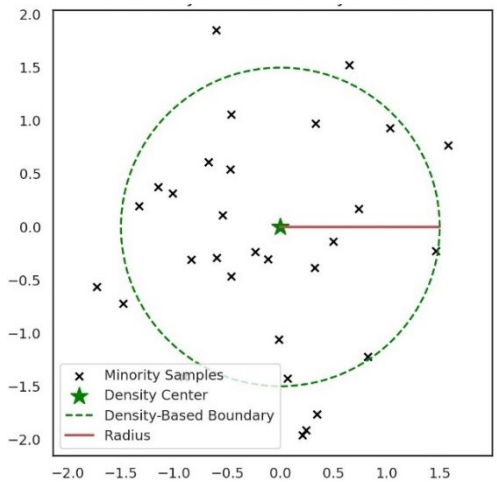

Figure 1. Density-based boundary detection on minority class samples. The central region is identified using KDE, and a circular boundary is drawn around the estimated density center to define a safe area for synthetic sample generation.

Following boundary construction, synthetic samples are generated using the DE strategy described in Section 3.1. For each selected minority instance, a new sample is created through mutation and crossover operations involving neighboring instances. This mechanism introduces diversity while preserving class consistency. The sample generation process is visualized in Figure 2.

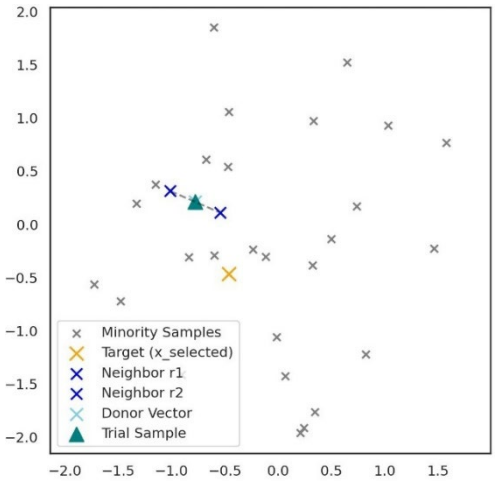

Figure 2. Synthetic sample generation using Differential Evolution. A target instance and three random neighbors are selected from the minority class, and a trial sample is generated through mutation and crossover operations.

Once candidate samples are produced, each one is validated based on its location relative to the previously defined boundary. Samples falling inside the safe zone are accepted, while those located outside are rejected to minimize the risk of noise or class overlap. The acceptance and rejection of synthetic instances are illustrated in Figure 3.

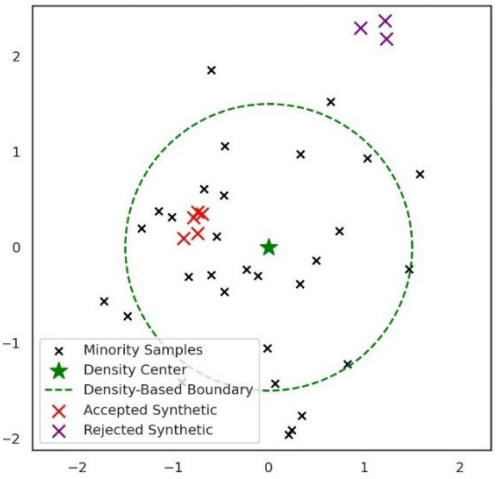

Figure 3. Validation of synthetic samples based on the boundary constraint. Samples generated inside the safe region are accepted, while those outsides are rejected to preserve class consistency and prevent noise.

4. EXPERIMENTAL EVALUATION

To evaluate the performance of BODE, experiments were conducted on a collection of benchmark imbalanced datasets obtained from the KEEL repository [19]. These datasets span a wide range of domains and vary significantly in size, dimensionality, and class distribution. Each dataset presents a binary classification problem in which the minority class is underrepresented to varying degrees.

The degree of class imbalance in each dataset is quantified using the imbalance ratio (IR), defined as the ratio of majority to minority class instances. A higher IR indicates a more severe imbalance, making the classification task more challenging.

A complete list of the datasets used in the experiments, along with their number of instances, features, and imbalance ratios, is provided in Table 1.

Table 1. Overview of the 44 Benchmark Datasets Used in the Experiments

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	led7digit02456789vs	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

Three widely used classification algorithms were employed to assess the impact of oversampling: k-Nearest Neighbors (k-NN), Decision Tree (DT), and Support Vector Machine (SVM) [20–22]. These classifiers were selected due to their distinct learning mechanisms and frequent use in imbalanced learning literature. k-NN serves as a distance-based non-parametric method, DT provides a rule-based approach sensitive to data structure, and SVM offers a margin-maximizing framework that is particularly relevant for high-dimensional and imbalanced data. This diversity allows for a comprehensive evaluation of BODE's effectiveness across different classification paradigms.

Model performance was evaluated using the AUC, which measures a classifier's ability to discriminate between positive (minority) and negative (majority) classes. AUC is a threshold-independent metric that reflects overall ranking quality and is particularly useful for evaluating classifiers on imbalanced datasets [23].

In simplified binary classification settings, AUC can be estimated using the following relationship, shown in Equation 3:

$$\mathrm{AUC} = \frac{1 + \mathrm{TPrate} - \mathrm{FPrate}}{2} \tag{3}$$

where TPR is the true positive rate and FPR is the false positive rate. This formulation captures the trade-off between sensitivity and specificity, providing a robust evaluation criterion when class distributions are skewed. For this reason, AUC was used as the sole performance metric in all experimental comparisons.

In this study, the performance of the proposed BODE method was compared against a diverse set of baseline oversampling techniques. The Original setting, where no oversampling was applied, served as a control to assess the impact of sample generation. SMOTE, the most widely used synthetic sampling method, and its two hybrid extensions—SMOTE-TomekLinks (S-TL) and SMOTE-ENN (S-ENN)—were included to represent noise reduction and boundary-cleaning strategies. Borderline-SMOTE1 (Border1) and Borderline-SMOTE2 (Border2) were selected for their emphasis on generating samples near decision boundaries. Safe-Level SMOTE and ADASYN were included as density-sensitive methods, which adjust the location or frequency of generated samples based on local minority instance distributions. In addition, SMOTE-RSB (S-RSB) was chosen for its relative safe boundary enhancement, and DEBOHID, an evolutionary oversampling technique, was included as a recent alternative leveraging directional perturbation. This comprehensive set of eleven methods enabled a robust comparison across traditional, boundary-focused, density-aware, and evolutionary oversampling paradigms.

5. RESULTS AND DISCUSSION

This section presents a comparative analysis of the proposed BODE method against ten widely used oversampling techniques. The evaluation was conducted using three different classifiers: k-Nearest Neighbors (k-NN), Decision Tree (DT), and Support Vector Machine (SVM). The performance of each method was assessed using the AUC metric, which provides a robust, threshold-independent measure of classification quality in imbalanced settings. Detailed results are reported in Tables 2, 3, and 4, corresponding to the outcomes obtained with k-NN, DT, and SVM, respectively. In each table, the best-performing oversampling method for each dataset is highlighted in bold, allowing for a quick visual comparison of model effectiveness across multiple settings.

According to the results reported in Table 2, the BODE method demonstrates strong and consistent performance when used with the k-Nearest Neighbors (k-NN) classifier. It achieves the highest AUC score in 28 out of 44 datasets, making it the most frequently successful oversampling technique in this

setting. In addition to its high win count, BODE also obtains the highest average AUC score of 0.8613, outperforming all baseline methods across the board.

Classical oversampling techniques such as SMOTE and ADASYN yield relatively lower and less stable results, particularly on datasets with complex or overlapping decision boundaries. Similarly, boundary-focused approaches like Borderline-SMOTE1 and SMOTE-TomekLinks show competitive performance in certain cases but fail to match BODE's consistency across diverse scenarios. The k-NN classifier, being highly sensitive to local neighborhood structure, benefits significantly from BODE's ability to generate boundary-aware and noise-filtered synthetic samples. These results confirm that BODE is particularly well-suited for nearest-neighbor-based learning under class imbalance.

The results in Table 3 further emphasize the effectiveness of BODE when used with the Decision Tree (DT) classifier. BODE achieves the highest AUC score in 33 out of 44 datasets, clearly outperforming all other oversampling methods in terms of win count. Additionally, it attains the highest average AUC value of 0.8625, reinforcing its robustness and adaptability across a wide variety of data distributions.

Compared to traditional techniques such as SMOTE, ADASYN, and Safe-Level SMOTE, BODE exhibits significantly better consistency, especially on datasets with complex or noisy structures. Even advanced methods like DEBOHID, which employs evolutionary strategies, fall behind BODE in both accuracy and frequency of top performance. The success of BODE in the DT setting can be attributed to its density-aware generation mechanism, which helps preserve the structural purity required for optimal tree-based splitting. These findings suggest that BODE is particularly effective in scenarios where decision trees are sensitive to sample quality near class boundaries.

As shown in Table 4, the proposed BODE method also performs strongly when evaluated with the Support Vector Machine (SVM) classifier. It achieves the highest AUC score in 26 out of 44 datasets, demonstrating its consistent effectiveness across a range of imbalanced scenarios. With an average AUC of 0.8559, BODE outperforms all other oversampling techniques, including those specifically designed for boundary enhancement.

Although methods such as Borderline-SMOTE1, SMOTE-RSB, and DEBOHID exhibit competitive performance in certain datasets, they fail to match BODE's overall balance between accuracy and consistency. The margin-maximizing nature of SVM particularly benefits from BODE's strategy of generating synthetic samples near informative decision boundaries while avoiding noisy or overlapping regions. These results confirm that BODE not only performs well across diverse data distributions but also adapts effectively to classifiers with fundamentally different decision-making mechanisms.

Across all three classifiers—k-NN, Decision Tree, and SVM—BODE consistently demonstrates superior performance. It achieves the highest average AUC and the greatest number of dataset-level wins in each setting: 28/44, 33/44, and 26/44, respectively. This highlights BODE's robustness and adaptability across different classification paradigms.

By combining boundary-aware instance selection, evolutionary sample generation, and density-based filtering, BODE generates high-quality synthetic samples near informative boundaries while minimizing noise. Unlike conventional or purely evolutionary methods, it balances diversity with structural reliability. These results confirm BODE's effectiveness as a general-purpose oversampling method for imbalanced data problems.

Table 2. The mean of the AUC results with the kNN classifier of all methods

Dataset Name	Original		SMOTE		S-TL		S-ENN		Border1		Border2		Safelevel		ADASYN		S-RSB		DEBOHID		BODE	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
ecoli0137vs26	0.8500	0.2236	0.8154	0.2066	0.8118	0.2040	0.8136	0.2039	0.8318	0.2113	0.8391	0.2171	0.8172	0.2064	0.8191	0.2074	**0.8581**	0.2069	0.8227	0.2042	0.8391	0.2172
shuttle0vs4	**0.9960**	0.0089	**0.9960**	0.0089	**0.9960**	0.0089	**0.9960**	0.0089	**0.9960**	0.0089	**0.9960**	0.0089	0.9951	0.0085	0.9951	0.0085	**0.9960**	0.0089	**0.9960**	0.0089	**0.9960**	0.0089
yeastB1vs7	0.5167	0.0373	0.7156	0.0948	0.6943	0.0698	0.7040	0.0553	0.6417	0.1355	0.6545	0.1146	0.6683	0.0682	0.7121	0.0635	0.7086	0.1150	0.7408	0.1338	0.7050	0.0768
shuttle2vs4	0.6000	0.2236	0.9960	0.0089	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	0.9673	0.0374	0.9960	0.0089	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
glass016vs2	0.5552	0.0763	0.6686	0.0924	0.6369	0.1310	0.6655	0.1356	0.6893	0.1372	0.6760	0.0540	0.6848	0.1500	0.6493	0.0744	0.6940	0.1342	0.6810	0.0992	**0.7198**	0.1567
glass016vs5	0.7386	0.1821	0.9686	0.0156	0.9157	0.1246	0.9186	0.1186	0.9771	0.0078	0.8714	0.1310	0.9457	0.0120	0.9600	0.0256	0.9243	0.1140	0.9243	0.1140	**0.9786**	0.0189
pageblocks13vs4	0.8076	0.0998	0.9383	0.0435	0.9250	0.0735	0.9416	0.0372	0.9351	0.0514	0.9351	0.0542	0.9047	0.0306	0.9360	0.0426	0.9106	0.0436	0.9317	0.0446	**0.9475**	0.0563
yeast05679vs4	0.6667	0.0879	0.8096	0.0607	0.8033	0.0636	0.8138	0.0597	0.8028	0.0684	0.8049	0.1017	0.7974	0.0631	0.8065	0.0554	0.7996	0.0706	0.8313	0.0453	**0.8349**	0.0572
yeast1289vs7	0.4995	0.0012	0.6322	0.1270	0.6923	0.0973	**0.6972**	0.0899	0.6107	0.1022	0.5481	0.0323	0.6256	0.0996	0.6617	0.1127	0.6557	0.1047	0.6404	0.0759	0.6073	0.0876
yeast1458vs7	0.4992	0.0017	0.6771	0.0744	**0.6824**	0.0786	0.6354	0.0658	0.5487	0.0927	0.5032	0.0424	0.6279	0.0682	0.6233	0.1105	0.6430	0.0394	0.6983	0.0658	0.6670	0.0975
yeast2vs4	0.8195	0.0452	0.8898	0.0449	0.9055	0.0149	0.8974	0.0553	0.8915	0.0397	0.8725	0.0497	0.8736	0.0230	0.8969	0.0372	0.8709	0.0538	0.8985	0.0306	**0.9225**	0.0459
Ecoli4	0.8484	0.1366	0.8965	0.0711	0.8965	0.0779	0.8997	0.0722	0.8905	0.0532	0.8905	0.0577	0.8870	0.0749	0.8870	0.0696	0.9215	0.0795	0.9060	0.0756	**0.9425**	0.0700
Yeast4	0.5741	0.0508	0.7959	0.0818	0.8238	0.0497	0.8087	0.0920	0.7401	0.0946	0.7554	0.0716	**0.8224**	0.0494	0.8069	0.0916	0.8069	0.0654	0.8252	0.0825	0.8112	0.0910
Vowel0	0.9772	0.0299	0.9978	0.0023	0.9994	0.0012	0.9994	0.0012	0.9994	0.0012	0.9994	0.0012	0.9783	0.0240	0.9911	0.0042	0.9961	0.0042	0.9994	0.0012	**1.0000**	0.0000
Yeast2vs8	0.7739	0.1033	**0.8295**	0.1159	0.8176	0.1172	0.8241	0.1175	0.7502	0.1134	0.7653	0.1094	0.7970	0.1604	0.8372	0.1015	0.8198	0.1258	0.7872	0.1432	0.8271	0.1332
Glass4	0.7808	0.1421	0.9001	0.0938	0.9001	0.0986	0.9051	0.1011	0.8917	0.1127	0.8942	0.1144	0.8704	0.1158	0.9052	0.1088	0.9151	0.1000	0.9226	0.1050	**0.9363**	0.1084
Glass5	0.6951	0.2080	0.9537	0.0338	0.9585	0.0281	0.9537	0.0200	0.8780	0.2125	0.9256	0.1268	0.9268	0.0414	0.9463	0.0318	**0.9659**	0.0291	0.8756	0.1212	0.8902	0.2187
Glass2	0.4848	0.0166	0.7948	0.0538	0.7562	0.1384	0.7688	0.1236	0.7001	0.0778	0.6756	0.0734	0.7437	0.0752	**0.8029**	0.1067	0.7763	0.1079	0.7174	0.0963	0.8021	0.0661
Yeast5	0.8497	0.0627	0.9663	0.0240	0.9649	0.0232	0.9663	0.0241	0.9510	0.0290	0.9500	0.0307	0.9757	0.0055	0.9653	0.0236	0.9656	0.0236	0.9674	0.0261	**0.9782**	0.0238
Yeast6	0.7387	0.1097	0.8736	0.0738	0.8705	0.0748	0.8705	0.0759	0.8566	0.1053	0.8719	0.1083	**0.8749**	0.0744	0.8708	0.0765	0.8698	0.0786	0.8367	0.0758	0.8708	0.0653
abalone19	0.5000	0.0000	0.5865	0.0702	0.5982	0.0350	0.5817	0.0697	0.6129	0.1051	0.5458	0.0579	0.5939	0.0727	0.5855	0.0707	0.5637	0.1063	0.5734	0.0942	**0.6780**	0.1199
abalone918	0.5681	0.0606	0.7720	0.1107	0.7675	0.1124	0.7703	0.1271	0.7355	0.1403	0.6616	0.1363	0.7663	0.1008	0.7800	0.1155	0.8212	0.1101	0.7815	0.1171	**0.8574**	0.0919
cleveland0vs4	0.4937	0.0086	0.5935	0.2184	0.5621	0.1891	0.5845	0.2056	0.6089	0.2395	0.6247	0.2318	0.5986	0.2418	0.5373	0.1542	0.5499	0.1602	**0.6916**	0.1912	0.5592	0.1836
ecoli01vs235	0.8300	0.0671	0.8964	0.0475	0.8468	0.1016	0.8941	0.0354	0.8386	0.0707	0.8386	0.0712	0.8450	0.0529	0.8645	0.0811	0.8223	0.1158	**0.9036**	0.0829	0.9001	0.1017
ecoli01vs5	0.9000	0.1046	0.9045	0.0743	0.8977	0.0705	0.9000	0.0743	0.8932	0.1037	0.8886	0.1014	0.8909	0.0756	0.8614	0.1011	0.9045	0.0790	0.9159	0.0721	**0.9170**	0.0709
ecoli0146vs5	0.8981	0.1023	0.9038	0.0978	0.9000	0.1012	0.9058	0.1025	0.8942	0.1008	0.8923	0.1014	0.8962	0.1039	0.8904	0.0977	0.9019	0.0967	0.9173	0.1083	**0.9187**	0.1094
ecoli0147vs2356	0.8467	0.0298	0.8712	0.0235	0.8696	0.0261	0.8760	0.0603	0.8821	0.0630	0.9020	0.0477	0.8329	0.0582	0.8598	0.0206	0.8612	0.0499	**0.9122**	0.0572	0.8957	0.0492
ecoli0147vs56	0.8384	0.1509	0.8875	0.0457	0.8711	0.0356	0.8728	0.0452	0.9037	0.0407	**0.9069**	0.0389	0.8825	0.0388	0.8744	0.0478	0.8776	0.0413	0.8923	0.0429	**0.9273**	0.0560
ecoli0234vs5	0.8944	0.1449	0.8975	0.1135	0.8975	0.1088	0.9031	0.1098	0.8890	0.1400	0.8890	0.1400	0.8920	0.1135	0.8561	0.1478	0.9031	0.1141	0.9113	0.1088	**0.9105**	0.1198
ecoli0267vs35	0.7875	0.0599	0.8679	0.0595	0.8654	0.0579	0.8654	0.0537	0.8501	0.1106	0.8452	0.1111	0.8079	0.1169	0.8531	0.0583	0.8778	0.0756	0.8728	0.0976	**0.9276**	0.0608
ecoli034vs5	0.8750	0.1250	0.9028	0.1215	0.8944	0.1189	0.9028	0.1133	0.8944	0.1105	0.8917	0.1073	0.8750	0.1035	0.8583	0.1109	0.8972	0.1165	0.9139	0.1184	**0.9153**	0.1176
ecoli0346vs5	0.8750	0.0884	0.9061	0.0647	0.8980	0.0683	0.9088	0.0670	**0.9223**	0.0711	0.9196	0.0736	0.8845	0.0526	0.8318	0.0873	0.9088	0.0724	0.9142	0.0676	0.9209	0.0724
ecoli0347vs56	0.8757	0.1337	0.8768	0.1243	0.8768	0.1241	0.8833	0.1210	0.9006	0.1370	0.9006	0.1364	0.8725	0.1175	0.8618	0.1237	0.8769	0.1283	0.9055	0.1271	**0.9232**	0.1380
ecoli046vs5	0.9000	0.1046	0.9060	0.0972	0.9032	0.0945	0.9061	0.1052	0.8919	0.0952	0.8891	0.0960	0.8868	0.0970	0.8788	0.0970	0.9060	0.1083	0.9142	0.1034	**0.9209**	0.1123
ecoli067vs35	0.8350	0.1799	0.8975	0.1257	0.8600	0.1210	0.8900	0.1285	0.8300	0.1671	0.8475	0.1664	0.8100	0.1701	0.8800	0.1220	0.8875	0.1259	0.8825	0.1653	**0.9135**	0.1406
ecoli067vs5	0.8475	0.0548	0.8525	0.0511	0.8575	0.0641	0.8375	0.0606	0.9000	0.0696	0.8975	0.0736	0.8550	0.0338	0.8675	0.0429	0.8675	0.0665	0.8925	0.0699	**0.9056**	0.0631
glass0146vs2	0.5118	0.0635	**0.7807**	0.1095	0.7367	0.0557	0.7166	0.0967	0.6809	0.1697	0.6883	0.0640	0.7464	0.0855	0.7087	0.1009	0.7246	0.0955	0.7326	0.1059	0.7422	0.0986
glass015vs2	0.5269	0.0794	0.7427	0.1188	0.7298	0.1439	0.7608	0.1432	0.6462	0.1913	0.6202	0.1343	0.7220	0.1469	0.7427	0.1271	0.7382	0.1258	0.7153	0.1100	**0.7691**	0.1226
glass04vs5	0.8500	0.1369	0.9445	0.0558	0.9507	0.0521	0.9570	0.0520	0.9816	0.0278	0.9570	0.0520	0.9154	0.0792	0.9445	0.0558	0.9511	0.0558	0.9820	0.0165	**1.0000**	0.0000
glass06vs5	0.7450	0.1771	0.9750	0.0250	0.9850	0.0137	0.9850	0.0137	0.9350	0.1181	0.8900	0.1399	0.9597	0.0221	0.9697	0.0208	0.9297	0.1150	0.9800	0.0209	**0.9875**	0.0125
led7digit02456789vs1	0.5393	0.0360	0.6386	0.1086	0.6491	0.1057	**0.8576**	0.0389	0.5881	0.1266	0.5856	0.1271	0.5494	0.0516	0.6361	0.1080	0.6230	0.1165	0.6621	0.0944	0.6671	0.0823
yeast0359vs78	0.6390	0.0660	**0.7671**	0.0584	0.7406	0.0402	0.7303	0.0393	0.7076	0.0878	0.7007	0.0865	0.7273	0.0671	0.7505	0.0732	0.7283	0.0431	0.7425	0.0801	0.7517	0.0363
yeast0256vs3789	0.7624	0.0553	0.7781	0.0481	0.7759	0.0480	0.7944	0.0492	0.7643	0.0368	0.7787	0.0302	0.7698	0.0643	0.7798	0.0438	0.7787	0.0684	0.7894	0.0603	**0.7963**	0.0570
yeast02579vs368	0.9023	0.0260	0.9133	0.0196	0.9016	0.0209	0.8916	0.0310	0.8871	0.0259	0.8971	0.0325	0.8933	0.0273	0.8834	0.0260	0.9044	0.0121	0.8947	0.0224	**0.9202**	0.0267
Average	0.7389		0.8449		0.8383		0.8467		0.8232		0.8156		0.8240		0.8324		0.8387		0.8476		**0.8613**	
Winner / Total	1/44		4/44		3/44		4/44		3/44		3/44		1/44		1/44		4/44		5/44		**28/44**	

Table 3. The mean of the AUC results with the DT classifier of all methods

Dataset Name	Original		SMOTE		S-TL		S-ENN		Border1		Border2		Safelevel		ADASYN		S-RSB		DEBOHID		BODE	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
ecoli0137vs26	0.8427	0.2202	0.6818	0.2097	0.7709	0.2104	0.7781	0.2085	0.8427	0.2201	0.7409	0.2501	0.7154	0.2424	0.5800	0.1318	0.6336	0.2197	0.7390	0.2413	**0.8391**	0.2172
shuttle0vs4	**1.0000**	0.0000	0.9997	0.0007	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	0.9991	0.0013	0.9997	0.0007	0.9997	0.0007	0.9997	0.0007	0.9960	0.0089
yeastB1vs7	**0.7100**	0.0598	0.5797	0.1179	0.6123	0.0395	0.6844	0.1094	0.6220	0.0738	0.6101	0.0952	0.6459	0.1101	0.6572	0.1111	0.6681	0.0908	0.6383	0.0837	0.7050	0.0768
shuttle2vs4	0.9500	0.1118	0.9918	0.0112	0.9960	0.0089	**1.0000**	0.0000	0.9500	0.1118	**1.0000**	0.0000	0.9298	0.1148	0.9960	0.0089	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
glass016vs2	0.5548	0.0870	0.6226	0.0625	0.6195	0.1147	0.6421	0.0849	0.5290	0.1101	0.6017	0.1073	0.5819	0.1423	0.6367	0.0834	**0.7345**	0.1074	0.6071	0.1146	0.7198	0.1567
glass016vs5	0.8329	0.2299	0.8300	0.2274	0.9329	0.1267	0.8629	0.2588	0.8386	0.2341	0.9386	0.1216	0.8129	0.2366	0.8686	0.1607	0.9214	0.1050	0.8443	0.2229	**0.9786**	0.0189
pageblocks13vs4	0.9955	0.0062	0.9475	0.0528	0.9755	0.0372	0.9565	0.0472	**0.9978**	0.0050	0.9600	0.0501	0.9653	0.0444	0.9354	0.0769	0.9630	0.0443	**0.9978**	0.0050	**0.9978**	0.0047
yeast05679vs4	0.6540	0.1137	0.8094	0.0649	0.7585	0.0782	0.7751	0.0911	0.7019	0.1039	0.6821	0.0965	0.7572	0.0493	0.7104	0.1095	0.7441	0.0664	0.7304	0.0918	**0.8349**	0.0572
yeast1289vs7	0.6353	0.1157	0.6118	0.1115	0.6348	0.1083	0.5870	0.1092	0.5793	0.0459	0.6096	0.0756	0.5856	0.0393	0.6274	0.0998	0.6026	0.0652	**0.6389**	0.0693	0.6073	0.0876
yeast1458vs7	0.5259	0.0515	0.5025	0.0489	0.5117	0.0660	0.4949	0.0760	0.5448	0.0515	0.5061	0.0401	0.5769	0.0963	0.5783	0.1019	0.5518	0.1001	0.6087	0.1307	**0.6670**	0.0975
yeast2vs4	0.8475	0.0782	0.8428	0.0201	0.8652	0.0550	0.9016	0.0415	0.8353	0.0746	0.8324	0.0693	0.8759	0.0333	0.8369	0.0696	0.8876	0.0447	0.8347	0.0298	**0.9225**	0.0459
Ecoli4	0.8624	0.1484	0.8608	0.0806	0.8592	0.1174	0.8278	0.1031	0.8389	0.0971	0.8203	0.0717	0.7997	0.1318	0.8263	0.1374	0.8810	0.1380	0.8389	0.0945	**0.9425**	0.0700
Yeast4	0.6484	0.0943	0.6965	0.0482	0.7558	0.0767	0.6944	0.0648	0.6754	0.0689	0.7030	0.0710	0.7162	0.1070	0.6712	0.0331	0.6975	0.1122	0.7138	0.1019	**0.8112**	0.0910
Vowel0	0.9422	0.0513	0.9444	0.0358	0.9727	0.0158	0.9633	0.0430	0.9039	0.0888	0.9533	0.0245	0.9483	0.0441	0.9655	0.0280	0.9589	0.0218	0.9561	0.0449	**1.0000**	0.0000
Yeast2vs8	0.7696	0.1046	0.7545	0.1322	0.7773	0.1287	0.8066	0.1393	0.7402	0.0924	0.7870	0.1077	0.7796	0.1700	0.7100	0.1638	0.7730	0.1438	0.7762	0.1376	**0.8271**	0.1332
Glass4	0.8567	0.1852	0.8984	0.0954	0.8818	0.1107	**0.9392**	0.0878	0.8450	0.1570	0.9067	0.1138	0.9051	0.1011	0.8460	0.1670	0.8917	0.1159	0.9350	0.1036	0.9363	0.1084
Glass5	0.8427	0.2180	0.9183	0.1042	0.9110	0.0990	0.9207	0.1035	0.8451	0.2230	0.7854	0.2611	0.8037	0.2097	**0.9280**	0.1015	0.8659	0.2199	0.8976	0.2223	0.8902	0.2187
Glass2	0.5376	0.1418	0.6904	0.0508	0.6060	0.1111	0.7132	0.1720	0.6052	0.1301	0.6058	0.1100	0.6429	0.1250	0.6965	0.1092	**0.8075**	0.0666	0.7390	0.0351	0.8021	0.0661
Yeast5	0.8201	0.0802	0.8521	0.0370	0.8847	0.0272	0.9076	0.0682	0.8337	0.0481	0.8309	0.0546	0.9281	0.0071	0.8885	0.0619	0.8632	0.0268	0.9021	0.0465	**0.9782**	0.0238
Yeast6	0.6823	0.1109	0.7974	0.1380	0.7936	0.1360	0.8089	0.1203	0.7506	0.1079	0.7649	0.1271	0.8242	0.1415	0.7705	0.1307	0.7953	0.0798	0.7633	0.1136	0.8708	0.0653
abalone19	0.4978	0.0020	0.5513	0.0682	0.5308	0.0738	0.5341	0.0746	0.4888	0.0091	0.5087	0.0302	0.5153	0.0408	0.5358	0.0744	0.5283	0.0428	0.5938	0.0431	**0.6780**	0.1199
abalone918	0.6911	0.1521	0.8039	0.1052	0.7340	0.1129	0.7151	0.1001	0.6837	0.1687	0.7274	0.1249	0.7623	0.1253	0.7202	0.1408	0.7411	0.1400	0.7508	0.1189	**0.8574**	0.0919
cleveland0vs4	0.7888	0.1178	0.7198	0.0674	0.7355	0.0695	0.6833	0.0759	**0.8906**	0.1506	0.7416	0.0506	0.8282	0.0980	0.6561	0.1256	0.7489	0.1535	0.7139	0.0764	0.5592	0.1836
ecoli01vs235	0.8114	0.1359	0.7709	0.1019	0.8959	0.0704	0.8173	0.1216	0.8136	0.1613	0.8632	0.0950	0.7932	0.1072	0.7936	0.0562	0.8045	0.0641	0.8155	0.1431	**0.9001**	0.1017
ecoli01vs5	0.8636	0.1512	0.8250	0.1021	0.8159	0.0998	0.8795	0.0915	0.8636	0.1139	0.8614	0.1115	0.8750	0.1134	0.7682	0.1780	0.8727	0.0517	0.8864	0.0927	**0.9170**	0.0709
ecoli0146vs5	0.7308	0.1967	0.7981	0.1544	0.8481	0.1645	0.8692	0.1473	0.7904	0.1427	0.8308	0.1021	0.7846	0.1363	0.8731	0.1198	0.8558	0.1207	0.8750	0.1534	**0.9187**	0.1094
ecoli0147vs2356	0.8219	0.1151	0.8174	0.0934	0.8642	0.0578	0.8674	0.0651	0.8071	0.0512	0.8236	0.1525	0.8146	0.0822	0.8293	0.1345	0.8475	0.0668	0.8524	0.0741	**0.8957**	0.0492
ecoli0147vs56	0.7886	0.1277	0.8324	0.0654	0.7993	0.1255	0.8122	0.0440	0.8654	0.0858	0.8854	0.0789	0.7923	0.0905	0.8393	0.0921	0.8392	0.0703	0.8837	0.0653	**0.9273**	0.0560
ecoli0234vs5	0.7806	0.0624	0.8834	0.1147	0.8892	0.1177	0.8862	0.1098	0.8584	0.1456	0.8195	0.1104	0.8202	0.1222	0.8724	0.1055	0.8562	0.1613	0.8501	0.1262	**0.9105**	0.1198
ecoli0267vs35	0.7952	0.1090	0.8577	0.0918	0.7903	0.1283	0.8254	0.1190	0.8078	0.1134	0.8078	0.1165	0.7879	0.1150	0.7754	0.1159	0.8080	0.1132	0.8304	0.1096	**0.9276**	0.0608
ecoli034vs5	0.8056	0.1562	0.8278	0.1245	0.8667	0.1405	0.8500	0.1308	0.7889	0.1387	0.8111	0.1429	0.8500	0.1143	0.8667	0.1004	0.8972	0.1193	0.8611	0.0997	**0.9153**	0.1176
ecoli0346vs5	0.8392	0.1103	0.8676	0.0400	0.8703	0.0448	0.8730	0.0376	0.8446	0.1072	0.8419	0.1356	0.8568	0.0720	0.8345	0.0994	0.9041	0.0433	0.8507	0.0802	**0.9209**	0.0724
ecoli0347vs56	0.7692	0.0772	0.8476	0.1544	0.8611	0.1636	0.8675	0.1527	0.8470	0.1480	0.8449	0.1583	0.8325	0.1489	0.8454	0.1115	0.9077	0.0520	0.8834	0.0819	**0.9232**	0.1380
ecoli046vs5	0.8141	0.1134	0.8119	0.1526	0.8508	0.1151	0.8591	0.0984	0.8336	0.1333	0.8586	0.1290	0.8592	0.1569	0.8924	0.0917	0.8592	0.0863	0.8782	0.0960	**0.9209**	0.1123
ecoli067vs35	0.8550	0.2181	0.8175	0.1535	0.8125	0.1589	0.8300	0.1619	0.7850	0.1791	0.8275	0.1726	0.8250	0.1635	0.8100	0.1638	0.8225	0.1662	0.8625	0.1556	**0.9135**	0.1406
ecoli067vs5	0.7700	0.2082	0.8775	0.0681	0.8375	0.0935	0.8900	0.0389	0.8025	0.1857	0.8600	0.0807	0.8650	0.0698	0.8650	0.0907	0.8300	0.1095	0.8275	0.1210	**0.9056**	0.0631
glass0146vs2	0.6120	0.1301	0.7539	0.1298	0.7685	0.1266	0.6757	0.0980	0.5793	0.1159	0.5492	0.0786	0.6751	0.1282	0.7346	0.0326	0.7137	0.0888	**0.8137**	0.1057	0.7422	0.0986
glass015vs2	0.5914	0.1430	0.6309	0.2545	0.6796	0.2202	0.7207	0.1659	0.6933	0.0960	0.6419	0.1742	0.6519	0.1706	0.7022	0.1723	0.7304	0.1323	0.6785	0.1960	**0.7691**	0.1226
glass04vs5	0.9941	0.0132	0.9401	0.0462	0.9761	0.0247	0.9577	0.0268	0.9941	0.0132	0.9938	0.0140	0.9632	0.0336	0.9574	0.0353	0.9463	0.0237	0.9941	0.0132	**1.0000**	0.0000
glass06vs5	0.9350	0.1055	0.9550	0.0671	0.9647	0.0518	0.9597	0.0517	0.9450	0.1095	0.9897	0.0141	0.9287	0.0354	0.9545	0.0372	0.9800	0.0274	**0.9950**	0.0112	**0.9875**	0.0125
led7digit02456789vs1	0.8788	0.0740	0.8869	0.0969	0.8808	0.0985	0.8372	0.0810	0.8955	0.0878	0.8931	0.0847	0.8871	0.0443	0.8958	0.0819	0.8689	0.0812	**0.9088**	0.0725	0.6671	0.0823
yeast0359vs78	0.6804	0.0428	0.5998	0.0760	0.6479	0.0539	0.6365	0.0756	0.6072	0.0399	0.6638	0.1051	0.7268	0.0884	0.6009	0.0225	0.6410	0.0461	0.6534	0.0714	**0.7517**	0.0363
yeast0256vs3789	0.7483	0.0280	0.7252	0.0735	0.7402	0.0548	0.7718	0.0262	0.7159	0.0470	0.7056	0.0473	0.7644	0.0563	0.7394	0.0503	0.7193	0.0504	0.7619	0.0731	**0.7963**	0.0570
yeast02579vs368	0.8715	0.0286	0.8963	0.0421	0.9090	0.0410	0.9168	0.0372	0.8696	0.0190	0.8782	0.0438	0.8910	0.0242	0.8918	0.0460	0.8854	0.0322	0.8724	0.0398	**0.9202**	0.0267
Average	0.7783		0.7984		0.8111		0.8136		0.7852		0.7924		0.7987		0.7951		0.8147		0.8194		**0.8625**	
Winner / Total	2/44		0/44		1/44		3/44		3/44		2/44		0/44		1/44		3/44		6/44		**33/44**	

Table 4. The mean of the AUC results with the SVM classifier of all methods

Dataset Name	Original		SMOTE		S-TL		S-ENN		Border1		Border2		Safelevel		ADASYN		S-RSB		DEBOHID		BODE	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
ecoli0137vs26	0.8500	0.2236	0.7935	0.1959	0.7935	0.1959	0.7971	0.2007	0.8263	0.2118	0.8244	0.2159	0.8327	0.2164	0.7917	0.1949	0.8398	0.2056	0.8172	0.1993	**0.9654**	0.0723
shuttle0vs4	**1.0000**	0.0000	**1.0000**	0.0000	0.9960	0.0089	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	0.9991	0.0013	0.9994	0.0008	0.9997	0.0007	**1.0000**	0.0000	**1.0000**	0.0000
yeastB1vs7	0.5000	0.0000	0.7636	0.0825	0.7543	0.0799	0.7419	0.0479	0.6718	0.1150	0.6707	0.1165	0.7733	0.0670	0.7651	0.0690	0.7605	0.0587	0.7384	0.0482	**0.7797**	0.0489
shuttle2vs4	**1.0000**	0.0000	0.9793	0.0361	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	0.9470	0.0556	0.9795	0.0292	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
glass016vs2	0.5000	0.0000	0.5171	0.1412	0.5407	0.1310	0.5352	0.1023	0.6281	0.1235	0.5979	0.0616	0.5138	0.1022	0.5221	0.1280	**0.6293**	0.0724	0.6157	0.1116	0.5650	0.0993
glass016vs5	0.4971	0.0064	0.9486	0.0163	0.9486	0.0217	0.9514	0.0163	**0.9800**	0.0128	0.9629	0.0278	0.9314	0.0156	0.9514	0.0163	0.9486	0.0128	0.9686	0.0156	0.9171	0.0626
pageblocks13vs4	0.4728	0.1333	0.4090	0.1235	0.4840	0.2333	0.2948	0.1355	0.4604	0.1287	0.4606	0.0641	0.3069	0.1815	0.3565	0.0655	0.3934	0.2353	0.6004	0.1689	**0.6071**	0.0983
yeast05679vs4	0.5000	0.0000	0.7865	0.0830	0.7912	0.0612	0.7854	0.0822	0.7943	0.0847	0.7953	0.0862	0.7944	0.0639	0.7829	0.0592	0.7844	0.0803	0.7838	0.0918	**0.8412**	0.0594
yeast1289vs7	0.5000	0.0000	0.7189	0.0528	0.7263	0.0390	0.6910	0.0378	0.6762	0.0682	0.6757	0.0690	0.7064	0.0532	0.7123	0.0496	0.6946	0.0269	0.6921	0.0343	**0.7583**	0.0722
yeast1458vs7	0.5000	0.0000	0.6343	0.0519	0.6146	0.0482	0.6357	0.0500	0.6308	0.0662	0.6081	0.0743	0.6328	0.0556	0.6146	0.0433	0.6388	0.0524	0.6333	0.0797	**0.6348**	0.0961
yeast2vs4	0.6691	0.1163	0.8964	0.0331	0.8953	0.0349	0.8907	0.0326	0.8896	0.0293	0.8874	0.0280	0.8885	0.0351	0.8677	0.0253	0.8864	0.0350	0.8805	0.0193	**0.9104**	0.0272
Ecoli4	0.5750	0.0685	0.9716	0.0119	0.9402	0.0505	0.9668	0.0103	0.9342	0.0616	0.9295	0.0612	0.9541	0.0206	0.9102	0.0549	0.9620	0.0153	0.9576	0.0526	**0.9921**	0.0001
Yeast4	0.5000	0.0000	0.8434	0.0217	0.8265	0.0348	0.8338	0.0336	0.8251	0.0300	0.8223	0.0310	0.8286	0.0314	0.8212	0.0355	0.8131	0.0328	0.8104	0.0294	**0.8724**	0.0330
Vowel0	0.8950	0.0645	0.9699	0.0077	0.9700	0.0095	0.9705	0.0093	0.9200	0.0367	0.9244	0.0560	0.9505	0.0150	0.9616	0.0170	0.9649	0.0080	0.9683	0.0274	**0.9828**	0.0146
Yeast2vs8	0.7739	0.1033	0.7664	0.0960	0.7653	0.0948	0.7664	0.0960	0.7065	0.1244	0.7141	0.1159	0.7739	0.1033	0.7394	0.0523	0.7631	0.0971	0.7718	0.1023	**0.8532**	0.0620
Glass4	0.5592	0.0993	0.9101	0.1041	0.8977	0.1047	0.9027	0.1076	**0.9226**	0.1042	0.9176	0.1013	0.8704	0.1091	0.9002	0.1123	0.9002	0.1060	0.9101	0.1052	0.8871	0.1705
Glass5	0.5000	0.0000	0.9366	0.0218	0.9366	0.0200	0.9439	0.0185	**0.9732**	0.0316	0.9561	0.0222	0.9366	0.0200	0.9463	0.0204	0.9415	0.0235	0.9683	0.0329	0.9000	0.0802
Glass2	0.5000	0.0000	0.6155	0.1537	0.6777	0.0396	0.6309	0.1197	0.6050	0.2247	0.5880	0.2053	0.6546	0.0812	**0.6803**	0.0239	0.6212	0.1475	0.6085	0.1486	0.6241	0.1046
Yeast5	0.5000	0.0000	0.9635	0.0066	0.9622	0.0055	0.9642	0.0066	0.9667	0.0033	0.9660	0.0038	0.9608	0.0044	0.9608	0.0068	0.9625	0.0062	**0.9635**	0.0032	0.9842	0.0048
Yeast6	0.5000	0.0000	0.8730	0.0694	0.8723	0.0694	0.8737	0.0694	0.8791	0.0928	0.8777	0.0922	0.8730	0.0706	0.8628	0.0739	0.8870	0.0574	0.8820	0.0702	**0.9180**	0.0826
abalone19	0.5000	0.0000	0.7453	0.0889	0.7601	0.0679	0.7623	0.0659	0.6914	0.1140	0.6963	0.1528	0.7821	0.0451	0.7786	0.0642	0.7826	0.0869	**0.7894**	0.0607	0.7719	0.0778
abalone918	0.5000	0.0000	0.8581	0.0429	0.8545	0.0419	0.8493	0.0481	**0.8833**	0.0728	0.8760	0.0769	0.8174	0.0391	0.8530	0.0414	0.8727	0.0360	0.8761	0.0444	0.7814	0.0552
cleveland0vs4	0.7478	0.2172	0.9167	0.0557	0.9167	0.0568	0.9166	0.0655	0.7947	0.1655	0.8426	0.1102	0.8950	0.0652	0.9314	0.0279	0.8735	0.0600	0.9260	0.0645	**0.9769**	0.0227
ecoli01vs235	0.8359	0.1670	0.8777	0.0981	0.8732	0.0931	0.8845	0.1032	0.8468	0.1525	0.8673	0.1092	0.8555	0.0873	0.8255	0.1379	0.8732	0.0931	0.8918	0.1027	**0.9161**	0.0937
ecoli01vs5	0.8364	0.1112	0.8591	0.0993	0.8341	0.1239	0.8614	0.0976	0.8864	0.1057	0.8568	0.1257	0.8614	0.1033	0.8068	0.1288	0.8364	0.1267	0.8750	0.1038	**0.9170**	0.1046
ecoli0146vs5	0.8635	0.1516	0.8885	0.0956	0.8712	0.0939	0.8519	0.0914	0.8808	0.1595	0.8769	0.1533	0.8769	0.0960	0.8423	0.0614	0.8827	0.1014	**0.8981**	0.1083	0.8731	0.1360
ecoli0147vs2356	0.8267	0.0538	0.8812	0.0710	0.8645	0.1066	0.8661	0.0554	0.8555	0.0470	0.8539	0.0428	0.8796	0.0774	0.8228	0.0509	0.8929	0.0740	0.8658	0.0681	**0.9154**	0.0621
ecoli0147vs56	0.8719	0.0863	0.8812	0.0229	0.8730	0.0704	0.8530	0.0498	0.8821	0.0712	0.9021	0.0793	0.9012	0.0376	0.8054	0.0760	0.8779	0.0513	0.8775	0.0608	**0.9276**	0.0777
ecoli0234vs5	0.8667	0.0933	0.8891	0.1135	0.8810	0.1133	0.8946	0.1109	0.8806	0.1032	**0.9029**	0.1134	0.8920	0.1154	0.8204	0.1016	0.8865	0.1108	0.9002	0.1112	0.8729	0.1708
ecoli0267vs35	0.8526	0.1062	0.8304	0.1198	0.8108	0.1403	0.8604	0.1075	0.8353	0.1020	0.8155	0.1074	0.8507	0.1347	0.8137	0.0904	**0.8883**	0.0928	0.8379	0.1127	0.8644	0.1467
ecoli034vs5	0.8611	0.1361	0.8611	0.1593	0.8639	0.1606	0.8639	0.1588	**0.9333**	0.0794	0.9000	0.1160	0.8611	0.1614	0.8111	0.1109	0.8611	0.1593	0.8722	0.1650	0.9014	0.1103
ecoli0346vs5	0.8696	0.0838	0.8899	0.0650	0.8926	0.0616	0.8676	0.0324	0.8588	0.0947	0.8588	0.0947	0.8899	0.0549	0.7993	0.0727	0.8872	0.0674	**0.9088**	0.0724	0.8608	0.1327
ecoli0347vs56	0.8935	0.0746	0.8947	0.0691	0.8883	0.0719	0.8925	0.0752	0.9007	0.0805	0.8985	0.0846	0.9040	0.0820	0.8651	0.0741	0.9019	0.0843	0.9099	0.0777	**0.9215**	0.1371
ecoli046vs5	0.8696	0.0887	0.8896	0.1160	0.8843	0.1111	0.8897	0.1143	0.8892	0.1088	0.8809	0.1103	0.8951	0.1158	0.8291	0.0886	0.8788	0.1077	**0.9032**	0.1170	0.8988	0.1270
ecoli067vs35	0.8525	0.2160	0.8425	0.2032	0.8650	0.1555	0.8525	0.2045	0.8350	0.2094	0.8000	0.2008	0.8400	0.2057	0.8000	0.1970	0.8325	0.1538	**0.8700**	0.1624	0.8463	0.2455
ecoli067vs5	0.8425	0.1357	0.8525	0.0675	0.8350	0.0757	0.8400	0.0693	0.8825	0.0535	0.8475	0.0768	0.8650	0.0912	0.7800	0.0942	0.8475	0.0681	0.8775	0.0548	**0.8825**	0.1084
glass0146vs2	0.5000	0.0000	0.6067	0.0504	0.6237	0.0759	0.6227	0.0571	0.6277	0.0588	**0.6785**	0.0687	0.6013	0.0511	0.6174	0.0473	0.6306	0.0881	0.6052	0.0591	0.6459	0.0819
glass015vs2	0.5000	0.0000	0.5126	0.0685	0.5094	0.1494	0.5320	0.0304	0.4927	0.1287	0.5261	0.1147	0.5191	0.0664	0.4868	0.1023	0.5132	0.1298	0.5000	0.0861	**0.5610**	0.0912
glass04vs5	0.8500	0.1369	0.9445	0.0518	0.9449	0.0465	0.9449	0.0465	**0.9816**	0.0278	0.9570	0.0360	0.9570	0.0415	0.9507	0.0472	0.9691	0.0383	**0.9816**	0.0278	0.9504	0.0569
glass06vs5	0.6500	0.1369	0.9387	0.0444	0.9437	0.0439	0.9387	0.0444	**0.9689**	0.0468	0.9087	0.1159	0.9387	0.0444	0.9387	0.0444	0.9387	0.0444	0.9589	0.0437	0.8684	0.0457
led7digit02456789vs1	0.9056	0.0794	0.8863	0.0579	0.8969	0.0469	0.8760	0.0523	0.8789	0.0444	0.8770	0.0765	0.8883	0.0576	0.8745	0.0729	0.8839	0.0825	0.8748	0.0689	**0.9605**	0.0304
yeast0359vs78	0.6067	0.0396	0.7484	0.0366	0.7254	0.0395	0.7417	0.0454	0.7482	0.0411	0.7450	0.0474	0.7506	0.0352	0.7025	0.0217	0.7462	0.0383	0.7439	0.0434	**0.7947**	0.0501
yeast0256vs3789	0.5486	0.0367	0.7957	0.0421	0.7913	0.0503	0.8007	0.0505	0.7929	0.0563	0.7923	0.0536	0.7940	0.0533	0.7717	0.0397	0.7951	0.0519	0.7907	0.0413	**0.8287**	0.0761
yeast02579vs368	0.8006	0.0693	0.9007	0.0362	0.9041	0.0320	0.9107	0.0268	0.9107	0.0276	0.9063	0.0268	0.9007	0.0364	0.8638	0.0359	0.9057	0.0302	0.9057	0.0300	**0.9290**	0.0365
Average	0.6942		0.8293		0.8296		0.8261		0.8279		0.8238		0.8260		0.8072		0.8329		0.8412		**0.8559**	
Winner / Total	2/44		1/44		1/44		2/44		9/44		4/44		0/44		1/44		3/44		9/44		**26/44**	

6. CONCLUSION

This study introduced BODE, a novel oversampling method that integrates boundary-aware instance selection with differential evolution and density-based validation. By focusing on borderline minority samples and ensuring that synthetic instances are generated within safe regions, BODE effectively improves classifier performance in imbalanced classification tasks.

Extensive experiments on 44 benchmark datasets using three different classifiers demonstrated that BODE consistently outperforms ten widely used oversampling techniques in terms of AUC. The method achieved the highest average performance and win rates across k-NN, Decision Tree, and SVM classifiers.

These results validate the strength of combining evolutionary diversity with structural awareness. Given its adaptability and robustness, BODE holds promise as a general-purpose solution for real-world imbalanced learning problems. Future work may focus on extending the approach to multi-class settings or integrating it with ensemble-based classifiers.

While the results demonstrate the strong performance and robustness of BODE, this study has certain limitations that present opportunities for future work. In particular, only the AUC metric was considered, and experiments were restricted to binary classification tasks on KEEL datasets. Future research could include evaluating BODE on multi-class and multi-label datasets, exploring additional performance metrics such as F1-score and G-mean, and investigating adaptive parameter optimization to further enhance its effectiveness.

REFERENCES

1. Leevy, J.L., Khoshgoftaar, T.M., Bauder, R.A. and Seliya, N., “A survey on addressing high-class imbalance in big data”, Journal of Big Data, Vol. 5, Pages 42, 2018.

2. Kononenko, I., “Machine learning for medical diagnosis: history, state of the art and perspective”, Artificial Intelligence in Medicine, Vol. 23, Pages 89–109, 2001.

3. Huang, S.Y., Lin, C.-C., Chiu, A.-A. and Yen, D.C., “Fraud detection using fraud triangle risk factors”, Information Systems Frontiers, Vol. 19, Pages 1343–1356, 2017.

4. Jiang, Y., Cukic, B. and Ma, Y., “Techniques for evaluating fault prediction models”, Empirical Software Engineering, Vol. 13, Pages 561–595, 2008.

5. Kovács, G., “SMOTE-variants: A Python implementation of 85 minority oversampling techniques”, Neurocomputing, Vol. 366, Pages 352–354, 2019.

6. Korkmaz, S., Şahman, M.A., Cinar, A.C. and Kaya, E., “Boosting the oversampling methods based on differential evolution strategies for imbalanced learning”, Applied Soft Computing, Vol. 112, Pages 107787, 2021.

7. Korkmaz, S., “Hybridization of DEBOHID with ENN algorithm for highly imbalanced datasets”, Engineering Science and Technology, an International Journal, Vol. 63, Pages 101976, 2025.

8. Wang, X., Li, Y., Zhang, J., Zhang, B. and Gong, H., “An oversampling method based on differential evolution and natural neighbors”, Applied Soft Computing, Vol. 149, Pages 110952, 2023.

9. Kaya, E., Korkmaz, S., Sahman, M.A. and Cinar, A.C., “DEBOHID: A differential evolution based oversampling approach for highly imbalanced datasets”, Expert Systems with Applications, Vol. 169, Pages 114482, 2021.

10. Bunkhumpornpat, C., Sinapiromsaran, K. and Lursinsap, C., “Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for handling the class imbalanced problem”, Proceedings, Pages 475–482, 2009.

11. He, H., Bai, Y., Garcia, E.A. and Li, S., “ADASYN: Adaptive synthetic sampling approach for imbalanced learning”, Proceedings of the 2008 IEEE International Joint Conference on Neural Networks, Pages 1322–1328, 2008.

12. Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P., “SMOTE: Synthetic Minority Over-sampling Technique”, Journal of Artificial Intelligence Research, Vol. 16, Pages 321–357, 2002.

13. Hairani, H., Anggrawan, A. and Priyanto, D., "Improvement performance of the Random Forest method on unbalanced diabetes data classification using SMOTE-Tomek Link", International Journal on Informatics Visualization, Vol. 7, Pages 258, 2023.

14. Nishat, M.M., Faisal, F., Ratul, I.J., Al-Monsur, A., Ar-Rafi, A.M., Nasrullah, S.M., Reza, M.T. and Khan, M.R.H., "A comprehensive investigation of the performances of different machine learning classifiers with SMOTE-ENN oversampling technique and hyperparameter optimization for imbalanced heart failure dataset", Scientific Programming, Vol. 2022, Pages 1–17, 2022.

15. Shen, Y., Wu, J., Ma, M., Du, X., Wu, H., Fei, X. and Niu, D., "Improved differential evolution algorithm based on cooperative multi-population", Engineering Applications of Artificial Intelligence, Vol. 133, Pages 108149, 2024.

16. Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y. and Xu, J., "An improved differential evolution algorithm and its application in optimization problem", Soft Computing, Vol. 25, Pages 5277–5298, 2021.

17. Salgotra, R. and Gandomi, A.H., "A novel multi-hybrid differential evolution algorithm for optimization of frame structures", Scientific Reports, Vol. 14, Pages 4877, 2024.

18. Chen, Y.-C., "A tutorial on kernel density estimation and recent advances", Biostatistics & Epidemiology, Vol. 1, Pages 161–187, 2017.

19. Alcalá-Fdez, J., Sánchez, L., García, S., del Jesus, M.J., Ventura, S., Garrell, J.M., Otero, J., Romero, C., Bacardit, J., Rivas, V.M., Fernández, J.C. and Herrera, F., "KEEL: A software tool to assess evolutionary algorithms for data mining problems", Soft Computing, Vol. 13, Pages 307–318, 2009.

20. Chandra, M.A. and Bedi, S.S., "Survey on SVM and their application in image classification", International Journal of Information Technology, Vol. 13, Pages 1–11, 2021.

21. Myles, A.J., Feudale, R.N., Liu, Y., Woody, N.A. and Brown, S.D., "An introduction to decision tree modeling", Journal of Chemometrics, Vol. 18, Pages 275–285, 2004.

22. Guo, G., Wang, H., Bell, D., Bi, Y. and Greer, K., "KNN model-based approach in classification", Proceedings, Pages 986–996, 2003.

23. Huang, J. and Ling, C.X., "Using AUC and accuracy in evaluating learning algorithms", IEEE Transactions on Knowledge and Data Engineering, Vol. 17, Pages 299–310, 2005.