The Journal of Supercomputing (2021) 77:10463–10487
https://doi.org/10.1007/s11227-021-03688-6

BCGAN: A CGAN-based over-sampling model using the boundary class for data balancing

Minjae Son[1] · Seungwon Jung[2] · Seungmin Jung[2] · Eenjun Hwang[2]

Accepted: 13 February 2021 / Published online: 5 March 2021

Abstract

A class imbalance problem occurs when a dataset is decomposed into one majority class and one minority class. This problem is critical in the machine learning domains because it induces bias in training machine learning models. One popular method to solve this problem is using a sampling technique to balance the class distribution by either under-sampling the majority class or over-sampling the minority class. So far, diverse over-sampling techniques have suffered from overfitting and noisy data generation problems. In this paper, we propose an over-sampling scheme based on the borderline class and conditional generative adversarial network (CGAN). More specifically, we define a borderline class based on the minority class data near the majority class. Then, we generate data for the borderline class using the CGAN for data balancing. To demonstrate the performance of the proposed scheme, we conducted various experiments on diverse imbalanced datasets. We report some of the results.

Keywords Imbalanced data · Conditional generative adversarial network (CGAN) · Borderline minority class · Over-sampling

Minjae Son and Seungwon Jung contributed equally to this work.

✉ Eenjun Hwang
ehwang04@korea.ac.kr

Minjae Son
smj5668@kyowon.co.kr

Seungwon Jung
jsw161@korea.ac.kr

Seungmin Jung
jmkstcom@korea.ac.kr

1 Kyowon, Seoul 04539, South Korea

2 School of Electrical Engineering, Korea University, Seoul 02841, South Korea

1 Introduction

Classification is the task of identifying the class to which the given input data belong [1]. So far, a variety of machine learning algorithms have been proposed and exploited as classification models, and their classification performance has consistently improved. Moreover, with the advent of deep learning algorithms, such as multilayer perceptrons and convolutional neural networks, classification accuracy has increased dramatically [2, 3].

However, to ensure such classification accuracy, it is essential to have a dataset with a balanced class distribution for model training. When the dataset is imbalanced, machine learning algorithms cannot properly learn the data in a minority class because the small portion of data in the minority class cannot fully represent various features of the class [4]. Furthermore, machine learning algorithms tend to be biased toward data in the majority class, which is actually the majority of the data [5]. Thus, in testing the trained models, their classification results are most likely to be in the majority class even if minority class data are given as input. This is called a class imbalance problem.

The class imbalance problem can be easily observed in real-world applications. For instance, suppose that we have data on patients who have undergone a cancer screening test. Because the number of negative patients is far greater than the number of positive patients, the data for the negative patients are generally much larger than the data for the positive patients. If we train machine learning algorithms using this dataset, the machine learning algorithms would be biased toward the negative patients and hence cannot properly identify the positive patients [6]. Similar situations occur in other fields, such as financial crime detection [7], customer demand detection [8], vehicle detection [9], and automatic extraction of definitions from documents [10].

One popular method to alleviate the class imbalance problem is to use a data sampling technique. This approach aims to balance the distribution of class data by adjusting the number of data in either class. Depending on the class being adjusted, it is divided into under-sampling and over-sampling techniques [11].

Under-sampling removes data in the majority class until the data size of the majority class is equal to that of the minority class. Popular schemes for under-sampling include random under-sampling and EasyEnsemble [12]. Random under-sampling removes data in the majority class randomly. In contrast, EasyEnsemble creates several balanced data subsets by randomly selecting data from the majority class and merging them with the minority class data. Then, this method constructs an ensemble of classification models generated for each subset. Under-sampling schemes improve classification performance, but they suffer from information loss problems due to data removal [13].

Unlike under-sampling, over-sampling generates data for the minority class to balance the distribution of the class data. Popular over-sampling schemes include random over-sampling (ROS), synthetic minority over-sampling technique (SMOTE) [14], adaptive synthetic sampling (ADASYN) [15], and borderline-SMOTE (B-SMOTE) [16]. However, these over-sampling schemes also have their own limitations. ROS makes classification models overfitted to the training data because it repeatedly generates the same data. On the other hand, the SMOTE-based schemes such as ADASYN and B-SMOTE often cannot generate minority class data effectively when neighbors of most minority class data are the majority class data [17].

Recently, as the generative adversarial network (GAN) [18] has emerged as a solution to the data shortage problem, various GAN-based schemes have been proposed to resolve the class imbalance problem [19]. The GAN is a deep learning-based generative model that estimates the probability distribution of the original data and generates realistic data. Due to its superior data generation performance, GANs have been used in diverse fields such as image generation [20], text generation [21], and video generation [22], and they also have been employed for over-sampling to generate realistic minority class data. GAN-based over-sampling methods have successfully overcome the limitations of the traditional over-sampling methods because these can generate minority class data based on the trained data distribution. The conditional generative adversarial network (CGAN) [23], which is an improved version of the GAN, has enhanced the classification performance further [24]. Unlike the GAN, the CGAN requires a condition and uses the condition when generating data. Thus, in model training, the CGAN learns all data in the training set because it can identify classes by the given class condition, while the GAN learns only the minority class data. This difference improves the quality of the data generation.

In this paper, we propose a novel over-sampling scheme based on the boundary conditional generative adversarial network (BCGAN) to achieve better classification accuracy compared to other over-sampling schemes. The BCGAN is a modified version of the CGAN and consists of two steps. In the first step, a borderline minority class is defined based on the minority class data near the decision boundary between the majority and minority class data. In the second step, the BCGAN learns about majority, minority, and borderline minority class data. For over-sampling, we generate data for the borderline minority class. Several studies have already adopted the approach to use data near the decision boundary and have demonstrated its effectiveness. However, to the best of our knowledge, combining the decision boundary and the CGAN for data generation has not yet been reported. To demonstrate the effectiveness of our scheme, we conducted several experiments and compared its performance with other over-sampling schemes. We report some of the results.

The contributions of the paper are summarized as follows:

- We propose an over-sampling scheme based on CGAN. Unlike GAN, CGAN can fully exploit all the data we have regardless of classes, which improves the quality of the generated minority class data.
- We combine a GAN-based approach with a decision boundary concept. To the best of our knowledge, this paper is the first work that considers both GAN and the decision boundary concept.
- We demonstrate the superiority of the proposed scheme through extensive experiments for 12 imbalanced datasets. In addition, we compare BCGAN-based over-sampling scheme with traditional over-sampling methods and visualize the distributions of the generated data.

The rest of this paper is organized as follows. In Sect. 2, we introduce several studies on over-sampling. In Sect. 3, we describe the details of the BCGAN and BCGAN-based over-sampling scheme. In Sect. 4, we describe some experiments that we performed for several datasets and present the experimental results. Lastly, in Sect. 5, we conclude the paper.

2 Related works

Various over-sampling schemes have been proposed to deal with how to expand minority classes. For instance, Chawla et al. [14] proposed an over-sampling method called SMOTE, which selects sample data from a minority class, finds its neighbors using the *k*-nearest neighbors (KNN) algorithm, and generates new data based on the linear combination of the selected sample and its neighbors. These steps are repeated until the number of minority class data are equal to the number of majority class data. Based on SMOTE, Haibo et al. [15] presented ADASYN, which uses the density of the minority class data. When selecting sample data from the minority class, ADASYN tends to choose the data surrounded by the majority class data. Han et al. [16] proposed a variation of SMOTE, called B-SMOTE. Like ADASYN, it has a different method for selecting sample data than SMOTE. It derives a decision boundary between the minority and majority class data and selects the minority class data near the decision boundary. Similarly, Wang [25] applied SMOTE only to the minority class support vectors derived from the training results of the biased support vector machine (SVM) [26]. Such schemes became popular solutions to the class imbalance problem because of their reasonable performance. However, they sometimes generate majority class data rather than minority class data when neighbors of most minority class data are the majority class data.

In contrast, Jo and Japowicz [27] proposed a clustering-based method. They performed *k*-means clustering on the majority and minority class data and applied the ROS to all the clusters until the total number of data in both classes was equal to the product of the number of clusters and the number of data in the largest class. In addition, Macia et al. [28] proposed a method of using a minimum spanning tree. They constructed a minimum spanning tree containing all of the data and randomly generated the data from a uniform distribution. They determined whether the generated data belonged to the minority class using the minimum spanning tree.

Over-sampling schemes can be combined with under-sampling schemes. For instance, Batista et al. [29] proposed a method that uses SMOTE for over-sampling and then performs under-sampling, such as Wilson's edited nearest neighbor (ENN) [30] and Tomek links [31]. This contributes to removing data that could be noise when training classification models regardless of classes, hence reducing classification errors. Further, Liu et al. [32] presented an ensemble-based method based on both over-sampling and under-sampling. They over-sampled the minority class data using SMOTE and under-sampled the majority class data to create subsets of majority class data. Then, they trained several classification models using the subsets and the over-sampled data and constructed an ensemble of the classification models.

Recently, GANs have been employed to solve the class imbalance problem. The GAN is a deep learning-based generative model that estimates the probability distribution of the original data using neural networks [18] and has demonstrated its superior performance of data generation [20, 21]. Using the GAN to generate data, Fiore et al. [19] demonstrated the possibility of the GAN as an over-sampling method by conducting experiments with financial data. Xie et al. [33] achieved further improvement in the classification performance of the GAN-based approach by adjusting the weight of each class in the discriminator loss. Zhou et al. [34] improved the classification performance through preprocessing, which transforms features into Gaussian distributions in the GAN training. In addition, Mariani et al. [35] proposed a GAN-based over-sampling scheme that initializes the discriminator using a pretrained autoencoder. Furthermore, Douzas and Bacao [24] improved the classification accuracy of imbalanced data using the CGAN, which is an extended version of the GAN. Due to the different structure of these models, CGAN uses whole data, while GAN uses only minority class data. Table 1 summarizes some of the related works. The decision boundary column in the table indicates whether the approach considers the decision boundary between majority and minority class when generating data. Generating minority class data near the decision boundary helps classification models find the decision boundary easily [16, 26], which has improved the classification performance. In this paper, we define a borderline class using the minority class data adjacent to the majority class data and intensively generate the borderline class data using the CGAN to improve the classification performance.

Table 1 Summary of the introduced works

Category	Approach	Author	Methodology	Decision boundary
Over-sampling	KNN	Chawla et al. [14]	SMOTE (KNN and Linear combination)	X
		Haibo et al. [15]	ADASYN (KNN and SMOTE)	O
		Han et al. [16]	B-SMOTE (KNN and SMOTE)	O
		Wang [25]	SMOTE and SVM	O
	GAN	Fiore et al. [19]	GAN	X
		Xie et al. [33]	GAN	X
		Zhou et al. [34]	GAN and Feature transform	X
		Mariani et al. [35]	GAN and Autoencoder	X
		Douzas and Bacao [24]	CGAN	X
	Etc	Jo and Japowicz [27]	*k*-means	X
		Macia et al. [28]	Minimum spanning tree	O
Hybrid sampling	–	Batista et al. [29]	SMOTE, ENN, and Tomek link	O
		Liu et al. [32]	SMOTE and Ensemble	X

3 Proposed scheme

In this section, we describe the details of our proposed scheme. We first introduce the CGAN and then describe how to define a borderline minority class. Afterward, we present the overall steps for over-sampling using the CGAN and borderline minority class.

3.1 Conditional generative adversarial network

The GAN is a deep learning-based generative model that consists of two networks, a generator G and a discriminator D. The discriminator determines whether the given data are real or fake and returns a value of 1 if real data are given and 0 otherwise. The generator generates realistic data so that the discriminator regards the generated data as real data. Owing to these adversarial purposes, they compete during training, and the objective function of the GAN, $V(D, G)$, in Eq. (1) describes these purposes mathematically. In the equation, p_{data} represents the distribution of the real data, and x is the data sample drawn from p_{data}. Similarly, p_z represents the noise distribution, and z is a noise sample from p_z:

$$V(D, G) = E_{x \sim p_{\text{data}}}\left[\log D(x)\right] + E_{z \sim p_z(z)}\left[\log(1 - D(G(Z)))\right] \tag{1}$$

From the perspective of the discriminator, $V(D, G)$ should be maximized so that $D(x)$ and $D(G(z))$ become 1 and 0, respectively. For the generator, $V(D, G)$ should be minimized to set $D(G(z))$ to 1. Thus, GANs are described as if the generator and the discriminator play a minimax game. The two individual networks update their parameters alternately corresponding to their purposes, and at the end of the training, the generator can generate realistic data.

The CGAN, which is an extended version of the GAN, has a similar structure and learning process. The only difference between them is that the generator and discriminator of the CGAN consider the given conditions. Eq. (2) presents the objective function of the CGAN, $V_{\text{CGAN}}(D, G)$. This function is similar to the objective function of the GAN, but condition y is added, affecting the output of both the generator and discriminator. Thus, the generated data are controlled by y. Owing to this property, the CGAN can be trained with y as class labels and be used as an over-sampling scheme by generating the minority class data under the condition that y is a minority class:

$$\begin{aligned} V_{\text{CGAN}}(D, G) = {} & E_{x,y \sim p_{\text{data}}}\left[\log D(x|y)\right] \\ & + E_{z \sim p_z(z), y \sim p_y}\left[\log(1 - D(G(z|y)))\right] \end{aligned} \tag{2}$$

3.2 Borderline minority class

For the more effective generation of data, we defined a borderline minority class using minority class data near the decision boundary. Many machine learning algorithms for classification have aimed to determine a decision boundary between classes to improve classification accuracy. In the same context, several previous studies for over-sampling have taken a similar approach of using the decision boundary. They found an estimated decision boundary and exploited the data near the decision boundary in generating minority class data. They reported a significant improvement in classification performance [16, 26].

Motivated by these studies, we also searched for minority class data near the decision boundary, replicated them, and labeled them with a new class called the borderline minority class. In selecting the borderline minority class data, we used the borderline sample selection method of B-SMOTE, as shown in Algorithm 1. For each data sample in the minority class a_i, we derived its k-nearest data samples $f(a_i)$ from all data using the KNN algorithm.

In $f(a_i)$, we count the number of data samples that belong to the majority class. If this number is greater than or equal to $k/2$ and less than k, we regard a_i as the borderline minority class data. Otherwise, we consider that a_i is far from a decision boundary or is fully surrounded by the majority class data. As a result, a set of the borderline minority class data contains minority class data near the decision boundary. By training a CGAN using the minority, majority, and borderline minority classes, we can obtain the BCGAN.

Algorithm 1 Borderline Minority Class Selection

```
$A = \{a_1, a_2, a_3, \cdots, a_n\}$: Set of minority class data
$B = \{b_1, b_2, b_3, \cdots, b_m\}$: Set of majority class data
$C = \emptyset$: Set of borderline minority class data
$f(a_i)$: Set of $k$-nearest neighbors of $a_i$
$N(S)$: The number of data in set $S$
for $i$=1 to $N(A)$ do
    if $N(f(a_i) \cap B) \geq \frac{k}{2}$ and $N(f(a_i) \cap B) < k$
        $C = C \cup \{a_i\}$
    end if
end for
```

3.3 Over-sampling using the boundary conditional generative adversarial network

Next, we describe the overall steps for BCGAN-based over-sampling. First, we determined the borderline minority class data using the steps in Sect. 3.2. Then, we obtained the BCGAN by training a CGAN using the majority, minority, and borderline minority classes. Figure 1 illustrates the overall flow for training BCGAN. The steps are as follows: (1) Randomly selected noise z and class

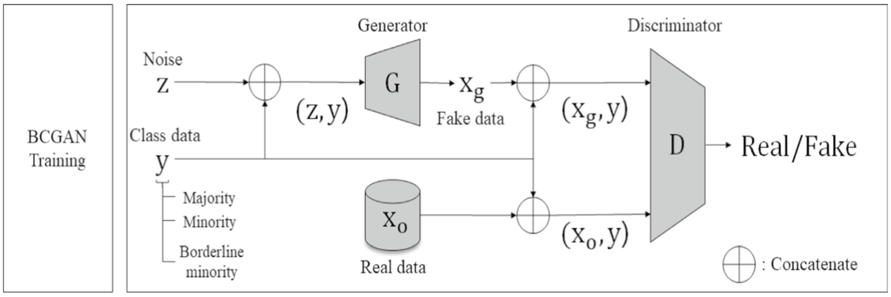


Fig. 1 Schematic of boundary conditional generative adversarial network (BCGAN) training

conditions y from the Gaussian distribution were concatenated and then given to the generator. (2) The generator generated fake data X_g based on the input (z, y). (3) The data X_g and y were concatenated. (4) Real data X_o, whose class is y, were selected from a dataset and concatenated with y. (5) Both (X_g, y) and (X_o, y) were given to the discriminator. (6) The discriminator distinguished between X_g and X_o based on y. (7) According to the objective function of the CGAN, the generator and discriminator updated the parameters in turn. (8) We repeated steps 1 to 7 until the end of the training.

Once the BCGAN is trained, then over-sampling can be carried out using the generator of the BCGAN. Figure 2 lists the steps for this. (1) A noise z was selected randomly from the Gaussian distribution, and class condition y was set as a boundary minority class. (2) The variables z and y were concatenated and then input into the generator. (3) The generator generated fake data X_g based on the input (z, y). (4) Steps 1 to 3 were repeated until the number of generated data reached the difference in the data numbers in each class. (5) The generated data merged with the minority class data. When we obtained a balanced dataset after this over-sampling process, we used this dataset to train classification models.

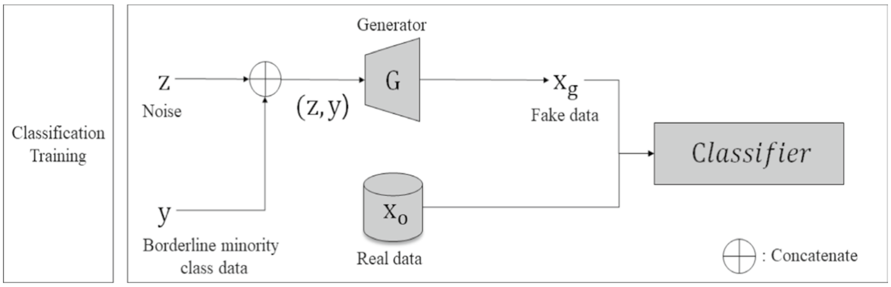


Fig. 2 Classification of the boundary conditional generative adversarial network (BCGAN)

Table 2 Statistics of ımbalanced datasets

Dataset	No of features	Features	No of data items	No of majority class data	No of minority class data	Classification task
Breast [36]	89	Clump thickness, bland chromatin, uniformity of cell size, etc.	683	444	239	Breast cancer?
WDBC [37]	30	Cell radius, compactness, concavity, etc.	569	357	212	Breast cancer?
Wine [38]	18	Fixed acidity, volatile acidity, pH, alcohol, etc.	6497	4898	1599	Red wine?
Letter [39]	16	Width, height, positions, average number of edges, etc.	20,000	19,211	789	Is the letter ‘A’?
Surgery [40]	24	Pain before surgery, hemoptysis before surgery, smoking, etc.	470	400	70	Survive for a year?
Yeast [41]	8	Responses to several methods, accession number in protein database, etc.	1484	1055	429	Localization site of yeast is nuclear?
CMC [42]	21	Wife’s age and education, husband’s education, number of children, etc.	1473	1140	333	Prefer long-term contraceptive method?
Card [43]	29	Encoded variables, amount of payment, and time	284,807	284,315	492	Fraud transaction?
Email [44]	9	Amount spent on the site, type of promotional e-mail, etc.	64,000	54,606	9394	Visit the site after receiving a promotional email?
Tel [45]	20	Age, gender, contact date, etc.	41,188	36,548	4640	Purchased any item?
Bio [46]	154	Examination results	59,795	59,782	13	Any compound activation reactions?
Pay [47]	23	Gender, marital status, age, etc.	30,000	23,364	6636	Fulfill obligations?

4 Experimental results

4.1 Conditional generative adversarial network

To evaluate the performance of our BCGAN-based over-sampling scheme, we collected 12 imbalanced datasets from diverse domains and conducted diverse classification tasks depending on the dataset domain. Table 2 lists some details of the collected datasets, which include the major features and the number of data features, data items, major class data, minor class data, and classification tasks.

4.2 Experimental setup

For the performance evaluation of our BCGAN-based over-sampling scheme, we conducted the following three experiments: (1) performance improvement in classification using the BCGAN-based over-sampling scheme, (2) comparison with other over-sampling schemes, and (3) distribution of the data generated by the BCGAN.

All experiments were done in Python 3.5 with several libraries, including scikit-learn 0.19.1, TensorFlow 1.7, and imblearn 0.4.3. Most hyperparameters used in the over-sampling schemes were set empirically. In the case of SMOTE, B-SMOTE, and ADASYN, we set $k=5$. In the case of the GAN, CGAN, and BCGAN, the numbers of hidden layers in the generator and discriminator were two (the number of nodes in each layer: 15–15), and the activation function was a rectified linear unit [48]. The activation function in the output layer was sigmoid function, and the optimizer used was Adam [49] with a learning rate of 0.001. Further, the batch size was set to 100. In the BCGAN, we set k to 5 to determine the borderline minority class data. The hyperparameters of the classification models were fixed during all experiments for fair comparison as follows. For the SVM, the regularization parameter was set to 1.0, and the kernel was the radial basis function. In the random forest (RF) method, the number of trees was 100, and the Gini index was used as an index of impurity. The setting of the multilayer perceptron (MLP) model was the same as the BCGAN, except that the MLP model had three hidden layers (the number of nodes in each layer: 25–10–5). Any hyperparameters not mentioned here followed the default setting of the scikit-learn library.

4.3 Data augmentation using the boundary conditional generative adversarial network

In the first experiment, we demonstrated the improvement in the classification performance due to the BCGAN-based over-sampling scheme. To do that, we first divided each dataset into training and testing sets at a ratio of 7:3 and augmented the training set using the BCGAN-based over-sampling scheme. Using the original and augmented training sets, we trained popular classification models, such as the SVM, RF [50], and MLP [51], and measured their area under the curve (AUC) for the testing set. Then, we repeated this process ten times to obtain an average AUC. The

AUC is the area under the line describing the dependence of a true positive rate on a false positive rate. A higher AUC indicates better performance of a classification model. If the AUC of a classification model is close to 0.5, it is considered to have performance similar to selecting a class at random. The AUC is a popular metric for evaluating the performance of classification models for imbalanced data [6]. We measured the AUC with and without the BCGAN-based over-sampling scheme and represented the degree of improvement by the ratio.

Figure 3 illustrates the experimental results. In the figure, the *x*-axis represents the classification models per dataset, and the *y*-axis represents the improvement in the AUC in percentage. All the classification models demonstrated improvement in the AUC when they were trained using the augmented data from the BCGAN-based over-sampling scheme. For example, the BCGAN achieved an average performance improvement of over 30% in all classification models for the surgery and bio datasets; particularly, RF achieved about a 40% performance gain. For the yeast and CMC datasets, our over-sampling scheme achieved relatively significant improvement in classification performance but showed relatively marginal performance improvement in the RF. Our scheme exhibited the least improvement for the pay dataset, which was lower than 2%. The improvements in other datasets such as the breast and wine datasets were also not great. However, considering the high AUC of the classification models based on the original datasets, this level of improvement is meaningful.

4.4 Comparison with other over-sampling schemes

In the second experiment, we compared the BCGAN-based over-sampling scheme with other over-sampling schemes, including ROS, SMOTE, B-SMOTE, ADASYN, GAN-based over-sampling (GAN), and CGAN-based over-sampling (CGAN). This experiment was conducted under the same conditions as the first experiment except for the improvement calculation. In addition, we conducted the Wilcoxon signed-rank test [52] to validate the improvement in our over-sampling scheme.

Table 3 reveals the AUC comparison of seven over-sampling schemes with a baseline. In the table, “Base” means the AUC of the classification models constructed

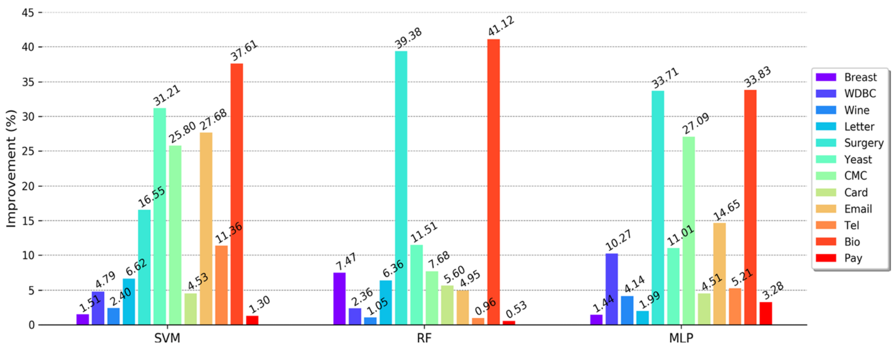


Fig. 3 Area under the curve (AUC) improvement in classification models

Table 3 Performance comparison of over-sampling schemes

Dataset	Model	Base	ROS	SMOTE	B-SMOTE	ADASYN	GAN	CGAN	BCGAN
Breast	SVM	0.9847	0.9943	0.9943	0.9938	0.9943	0.9987	0.9992	**0.9996**
	RF	0.9299	0.9810	0.9830	0.9795	0.9798	**0.9995**	0.9993	0.9993
	MLP	0.9847	0.9799	0.9819	0.9807	0.9803	0.9988	0.9982	**0.9989**
WDBC	SVM	0.9511	0.9501	0.9509	0.9429	0.9497	0.9962	0.9967	**0.9967**
	RF	0.9749	0.9670	0.9655	0.9692	0.9585	0.9978	0.9969	**0.9979**
	MLP	0.9068	0.9855	0.9860	0.9828	0.9775	0.9965	0.9995	**0.9998**
Wine	SVM	0.9739	0.9888	0.9869	0.9831	0.9866	0.9856	0.9957	**0.9973**
	RF	0.9863	0.9915	0.9910	0.9898	0.9905	0.9919	0.9949	**0.9966**
	MLP	0.9565	0.9947	0.9945	0.9937	0.9935	0.9848	0.9959	**0.9961**
Letter	SVM	0.9150	0.9550	0.9573	0.9416	0.9448	0.8713	0.9340	**0.9756**
	RF	0.9380	0.9564	0.9558	0.9538	0.9551	0.9793	0.9974	**0.9977**
	MLP	0.9779	0.9661	0.9728	0.9710	0.9749	0.9882	0.9965	**0.9973**
Surgery	SVM	0.5000	0.6300	**0.6469**	0.6228	0.6051	0.5861	0.5768	0.5827
	RF	0.4821	0.5271	0.5309	0.5128	0.5211	**0.6820**	0.6454	0.6720
	MLP	0.5000	0.5418	0.5941	0.5991	0.6233	0.5311	0.6221	**0.6685**
Yeast	SVM	0.5508	0.6779	0.6798	0.6732	0.6745	0.6532	0.6957	**0.7226**
	RF	0.6923	0.7112	0.7050	0.7050	0.7103	0.7711	**0.7728**	0.7720
	MLP	0.7021	0.7099	0.7125	0.7067	0.7012	0.7369	0.7662	**0.7794**
CMC	SVM	0.5531	0.6369	0.6414	0.6405	0.6425	0.6401	0.6836	**0.6959**
	RF	0.6135	0.6229	0.6243	0.6134	0.6071	0.6532	0.6389	**0.6606**
	MLP	0.5561	0.6400	0.6321	0.6363	0.6313	0.6908	0.6959	**0.7067**
Card	SVM	0.9121	0.9510	**0.9543**	0.9528	0.9517	0.9415	0.9534	0.9534
	RF	0.8532	0.8592	0.8671	0.8491	0.8571	0.8570	0.8780	**0.9010**
	MLP	0.9329	0.9591	0.9613	0.9592	0.9643	0.9650	0.9710	**0.9750**
Email	SVM	0.4953	0.6312	0.6313	**0.6362**	0.6334	0.6292	0.6321	0.6324
	RF	0.5662	0.5623	0.5592	0.5621	0.5582	0.5631	0.5881	**0.5942**
	MLP	0.5644	0.6261	0.6154	0.6340	0.6313	0.6323	0.6372	**0.6471**
Tel	SVM	0.6972	0.7843	0.7844	**0.7854**	0.7823	0.6312	0.7791	0.7764
	RF	0.7710	0.7712	0.7633	0.7661	0.7671	0.7711	0.7731	**0.7784**
	MLP	0.7531	0.7790	0.7812	0.7811	0.7812	0.7793	0.7892	**0.7923**
Bio	SVM	0.5212	0.5113	0.5114	**0.7311**	0.5513	0.5632	0.7011	0.7172
	RF	0.5231	0.7341	**0.7421**	0.7354	0.7322	0.7312	0.7320	0.7382
	MLP	0.5920	0.6963	0.7172	0.5431	0.7251	0.7231	0.7812	**0.7923**
Pay	SVM	0.7151	0.7290	0.7310	**0.7371**	0.7360	0.7191	0.7212	0.7244
	RF	0.7512	0.7491	0.5560	0.5213	0.5181	0.7062	0.7450	**0.7552**
	MLP	0.7582	0.7554	0.7652	0.7611	0.7693	0.7663	0.7752	**0.7831**

Bold values indicate the best AUC value among those of the comparison models

using the original data. The calculated standard deviations can be found in the tables of Appendix 2. Figures 4 and 5 graphically present the AUC comparison of seven over-sampling schemes for the datasets. Overall, the BCGAN outperforms other

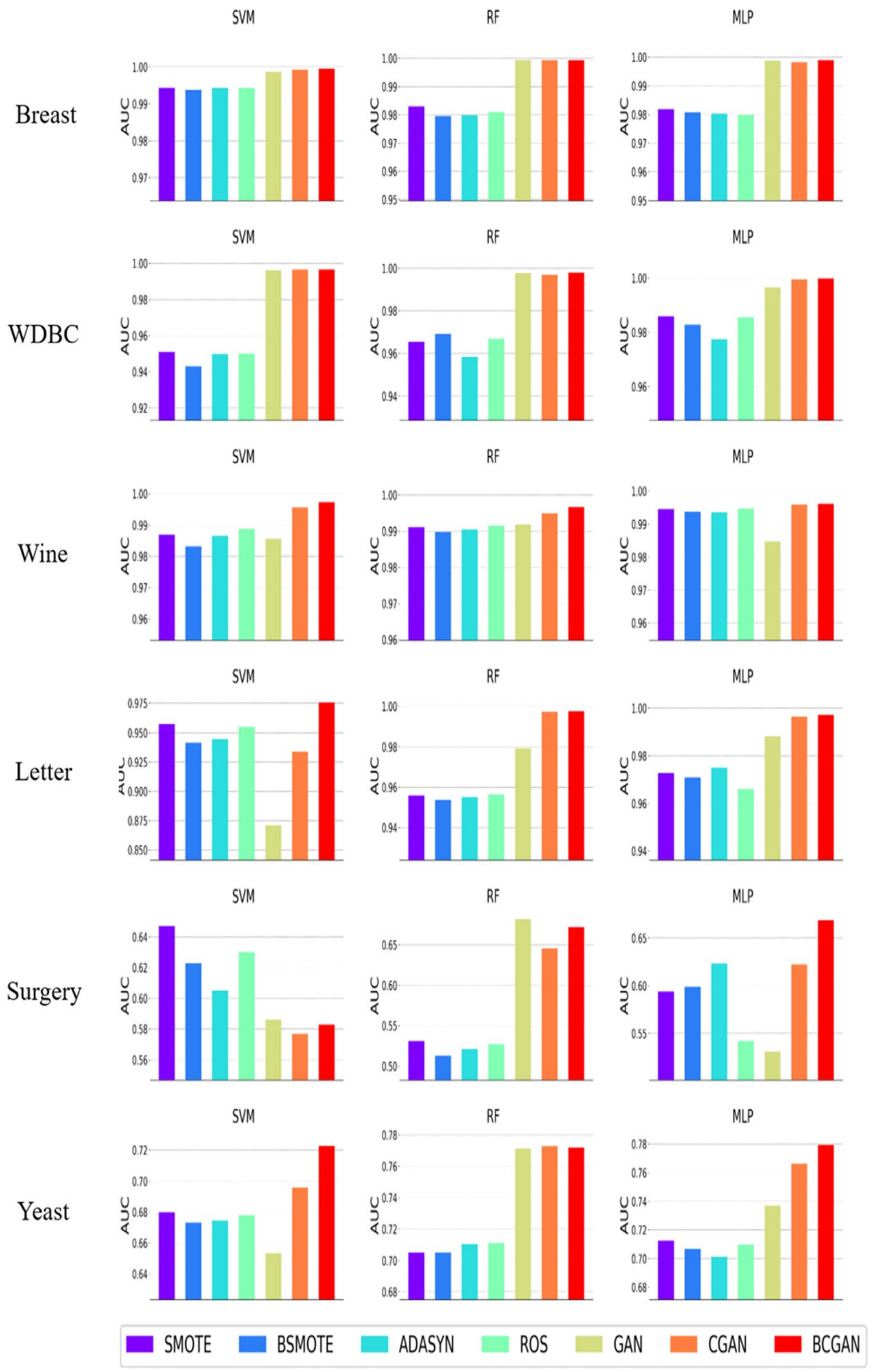


Fig. 4 Area under the curve (AUC) comparison of over-sampling schemes (part 1)

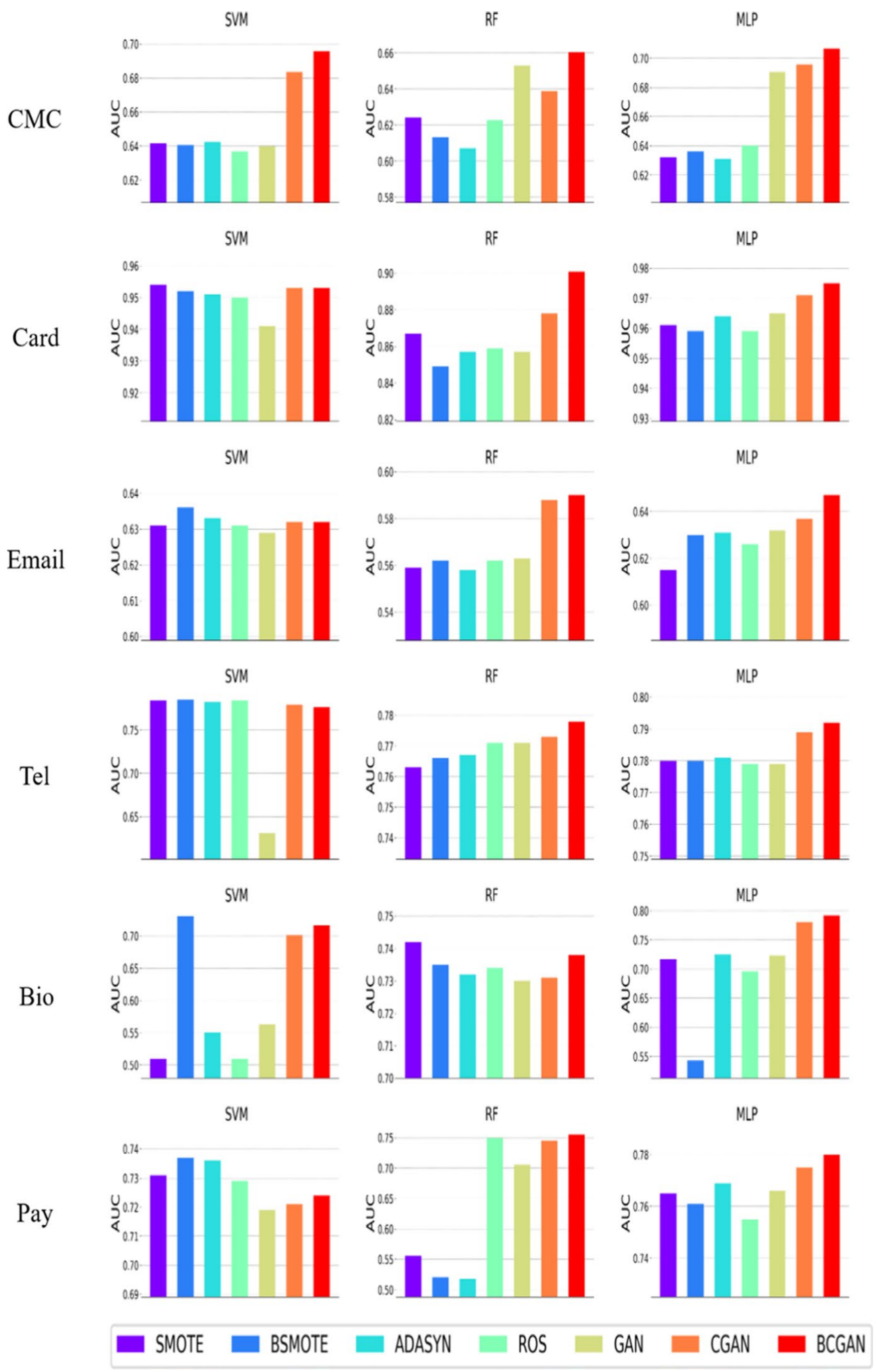


Fig. 5 Area under the curve (AUC) comparison of over-sampling schemes (part 2)

over-sampling schemes. In particular, the BCGAN achieved the best performance on the WDBC, wine, letter, and CMC datasets in all the classification models. On the breast and yeast datasets, the BCGAN could not completely outperform other over-sampling schemes. More specifically, the BCGAN exhibited better performance than other over-sampling schemes for the SVM and MLP, and the second-best performance for RF. The BCGAN drastically improved the AUC of the RF and MLP in the card, email, tel., bio, and pay datasets. For the SVM, the BCGAN was ranked below SMOTE or B-SMOTE. However, the differences between the BCGAN and these schemes were marginal, considering the differences in other datasets and other classification models. Lastly, in the surgery dataset, the BCGAN demonstrated inferior performance for the SVM, while it achieved the best AUC for the MLP. To summarize, the BCGAN could be an outstanding over-sampling scheme compared to other over-sampling schemes regardless of the classification models to be used. In particular, when the MLP is used as a classification model, significant performance improvement in the classification can be expected.

Further, we performed the Wilcoxon signed-rank test to verify that the BCGAN-based over-sampling is the best among the over-sampling schemes considered in this paper. The Wilcoxon signed-rank test examines the null hypothesis that no difference exists between the two given cases [53]. If the *p* value is below the significance level, the null hypothesis is rejected, indicating that a difference exists between the cases. In this experiment, the significance level was set to 0.05, and the results of the Wilcoxon signed-rank test are listed in Table 4. The *p* value in all cases is below the significance level. B-SMOTE has the largest *p* value, but this value is much smaller than 0.05. This proves that the BCGAN is superior to other conventional over-sampling schemes.

4.5 Comparison of original and generated data

In the third experiment, we investigated the distribution of the generated data using the principal component analysis (PCA) method [54]. If the BCGAN is trained well, it should be able to generate the minority class data near the majority class. However, it is not easy to visualize the data distribution precisely because the number of features in each dataset is high dimensional (i.e., greater than three). Thus, we

Table 4 Wilcoxon signed-rank test results

Over-sampling schemes	*p* value (<.05)
Random over-sampling (ROS)	1.885×10^{-7}
Synthetic minority over-sampling technique (SMOTE)	3.557×10^{-7}
Adaptive synthetic sampling (ADASYN)	9.172×10^{-7}
Borderline-SMOTE (B-SMOTE)	1.656×10^{-6}
Generative adversarial network (GAN)	8.930×10^{-7}
Conditional generative adversarial network (CGAN)	1.620×10^{-6}

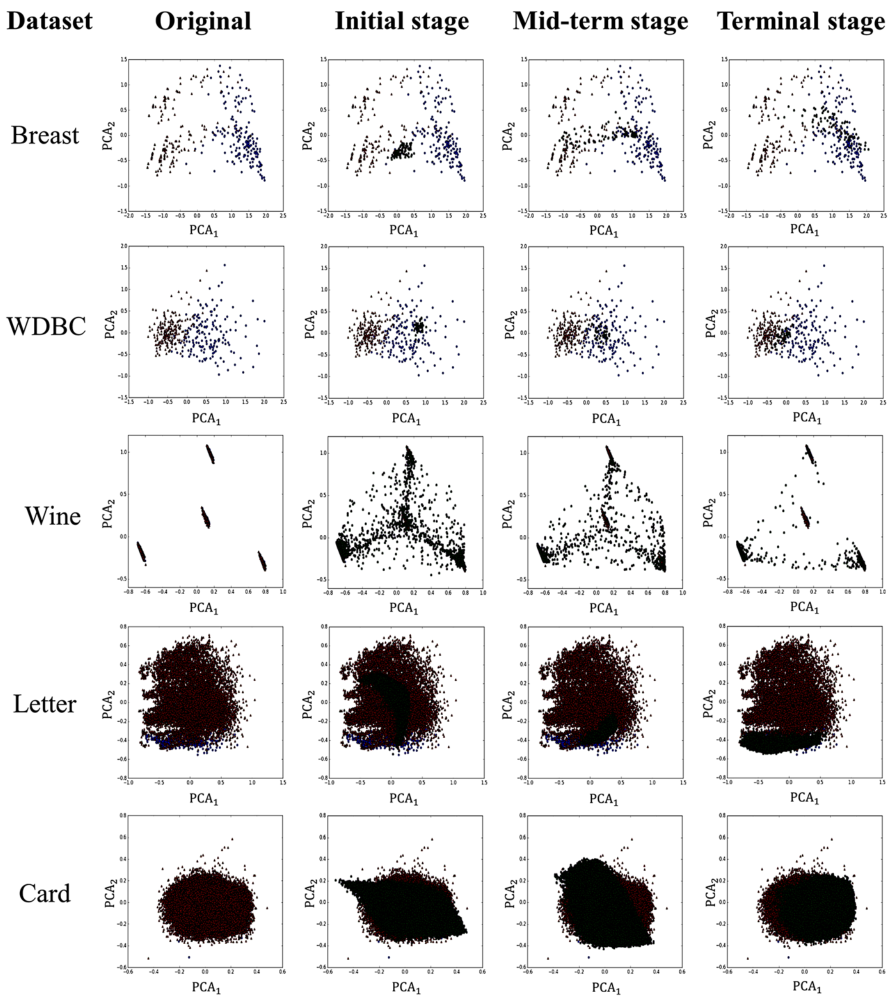


Fig. 6 Distribution of data by the BCGAN for datasets with a base area under the curve (AUC) > 0.9

reduced the feature dimension using the PCA and illustrated the distribution of the dimension-reduced data.

Figures 6 and 7 depict the distribution of the data generated by the BCGAN. To set the training process step by step, we divided the entire training process into three stages: the initial, mid-term, and terminal stages. As mentioned, we used the PCA for visualization and represented the data distribution using 2D space comprising the first and second principal components. In the figures, PCA_1 and PCA_2 on the *x*- and *y*-axes represent the first and second principal components, respectively. For comparison, we performed the PCA on the original and generated data. In the figure, the blue circles, red triangles, and green circles represent the minority class data, majority class, and borderline minority class generated by the BCGAN, respectively.

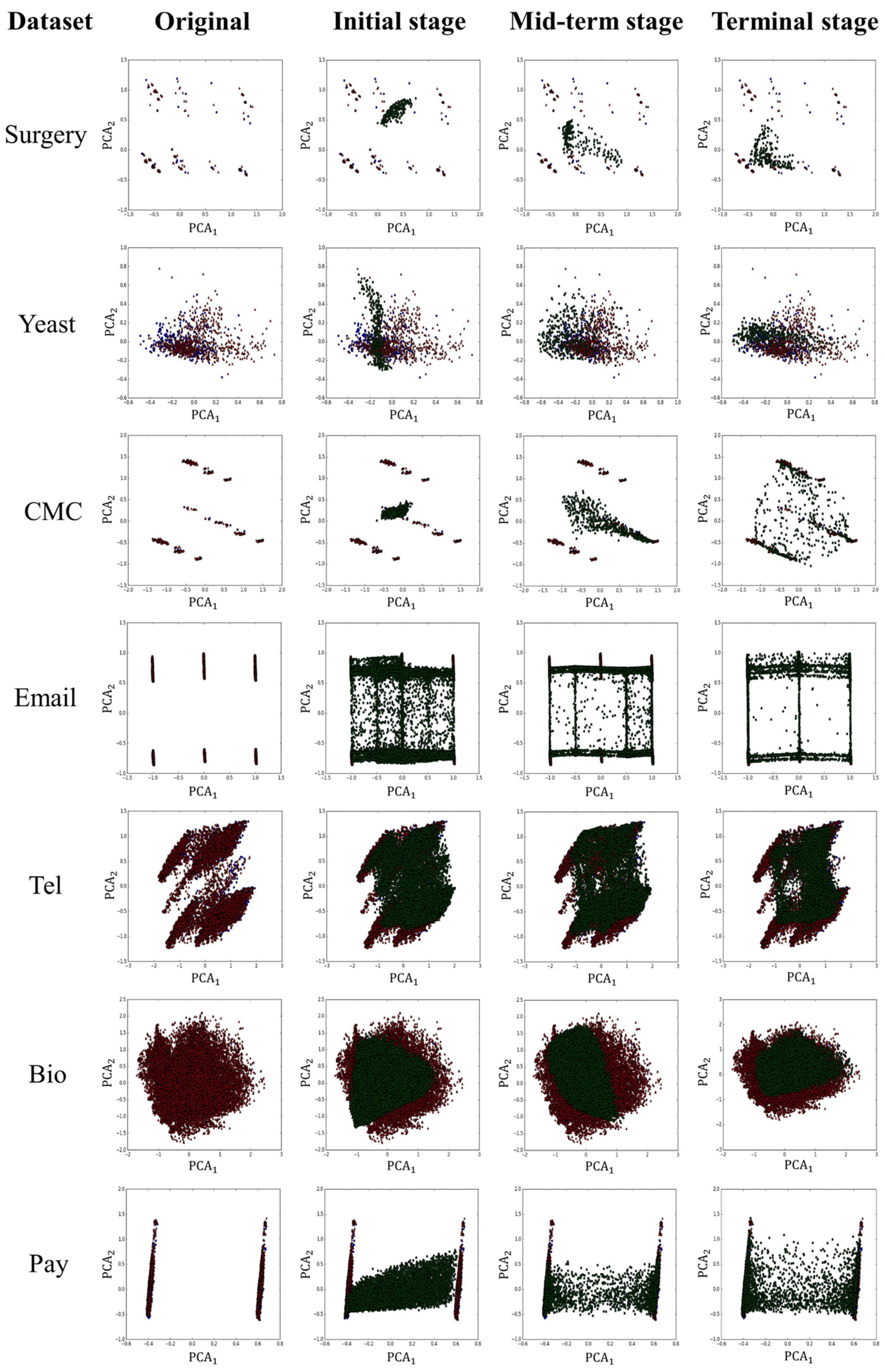


Fig. 7 Distribution of data by the BCGAN for the other datasets

Figure 6 displays the distribution of datasets with base AUCs larger than 0.9. As the training progresses, the distribution of the generated data became increasingly similar to that of the minority class data. In particular, the BCGAN generated minority class data near the majority class data, which was the goal of the BCGAN that generates data near the decision boundaries.

Meanwhile, Fig. 7 illustrates the distribution of datasets whose base AUCs were less than 0.9. Overall, the distribution of the generated data is similar to that of the minority class data. However, compared to the previous datasets, the generated data follow the distribution of the original data less accurately. The reason for this is that the size of the minority class is too small for the BCGAN to learn its characteristics or that the minority class data do not have sufficient features to discriminate between the two classes.

5 Conclusion

In this paper, we proposed a novel over-sampling scheme using BCGAN to solve the class imbalance problem in the classification tasks. The BCGAN generated minority class data along the decision boundary between the majority and minority classes to improve the classification performance. To demonstrate the effectiveness of our proposed scheme, we conducted experiments including data generation for various imbalanced datasets and compared the performance of popular classification models. The experimental results demonstrated that the BCGAN-based over-sampling scheme could effectively generate minority class data and have popular classification models achieve an improved classification accuracy up to about 40% using such augmented data. In addition, we compared the BCGAN-based over-sampling with other over-sampling methods. We showed that the BCGAN-based over-sampling over-sampling scheme achieved statistically significant improvement in performance by conducting the Wilcoxon signed-rank test.

In future work, we will research to determine the optimized *k* values for each data point in the algorithm to find the borderline minority class. In addition, we will consider more datasets from diverse domains to increase the versatility of our scheme.

Appendix 1: Correlation matrices of datasets

For better understanding of the datasets used, we present the correlation matrices of each dataset. Figure 8 describes the obtained correlation matrices with heatmap.

In the figure, each cell indicates the correlation between features in x-axis and y-axis by color. The closer the color is to dark blue, the higher the correlation value.

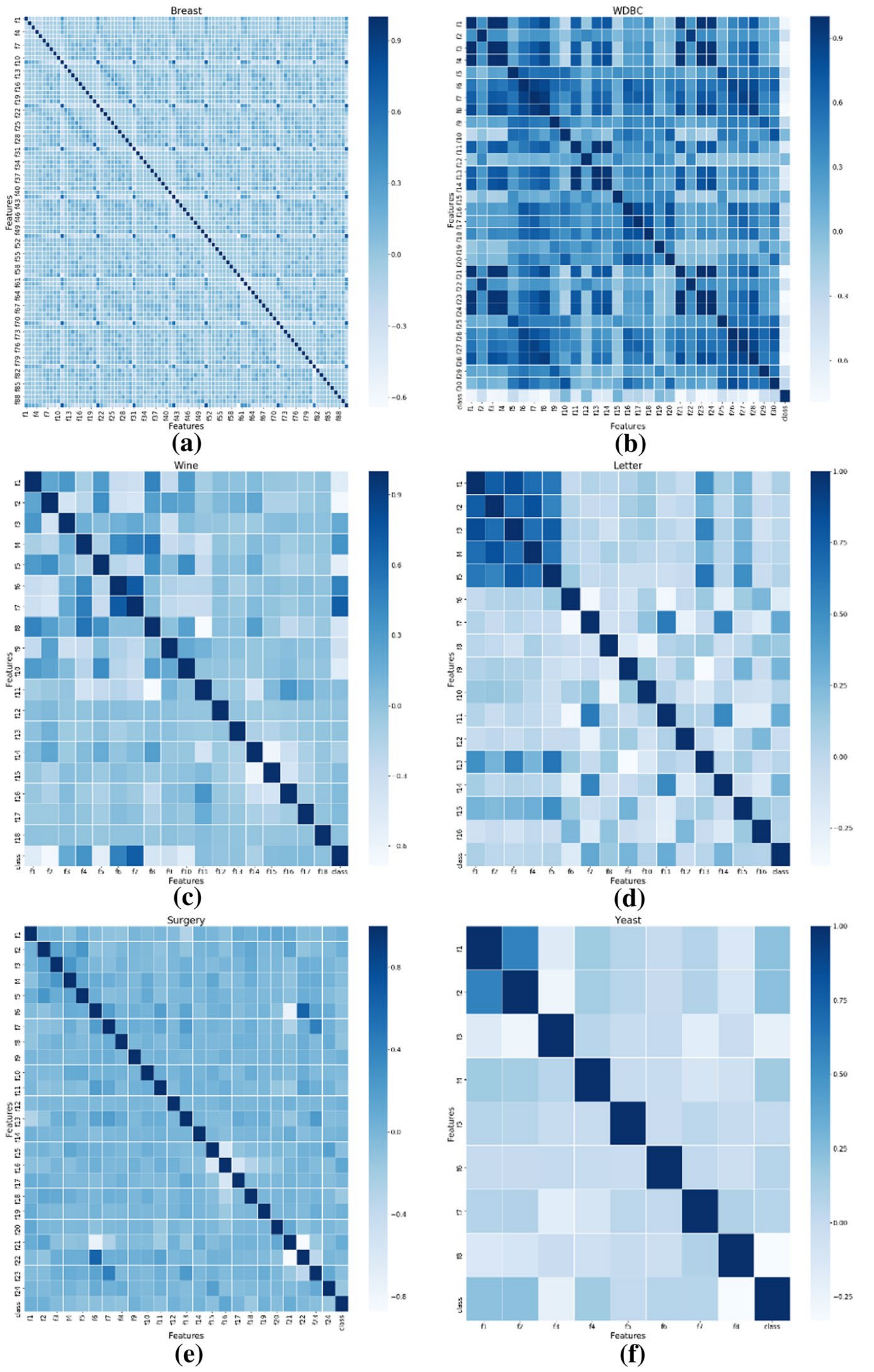
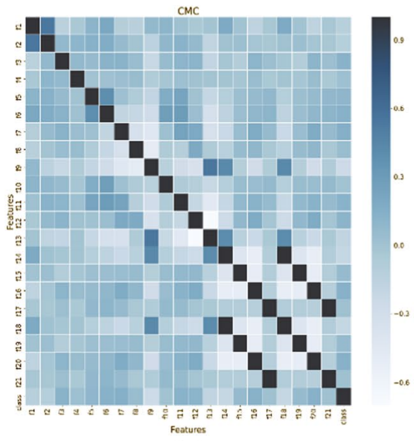
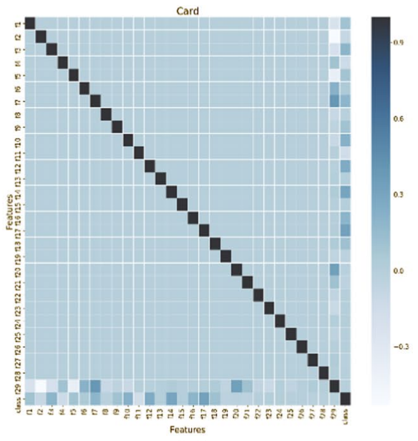


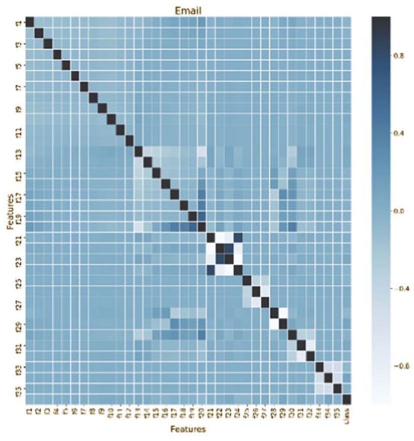
Fig. 8 Correlation matrices of all datasets: **a** Breast, **b** WDBC, **c** Wine, **d** Letter, **e** Surgery, **f** Yeast, **g** CMC, **h** Card, **i** Email, **j** Tel, **k** Bio, **l** Pay (color figure online)



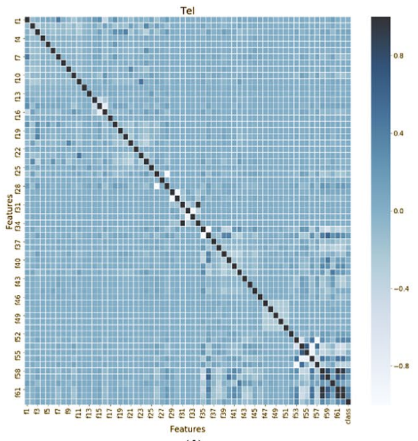
(g)



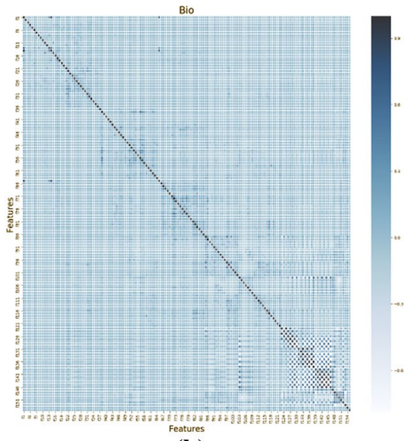
(h)



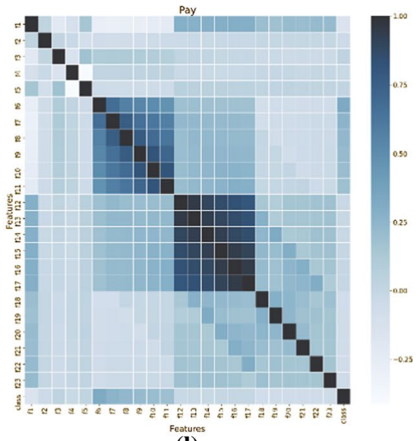
(i)



(j)



(k)



(l)

Fig. 8 (continued)

Table 5 Performance comparison of over-sampling methods and classification models with six datasets

Dataset	Model	Base	ROS	SMOTE	B-SMOTE	ADASYN	GAN	CGAN	BCGAN
Breast	SVM	0.9847	0.9943 ± 0.0000	0.9943 ± 0.0000	0.9938 ± 0.0012	0.9943 ± 0.0000	0.9987 ± 0.0001	0.9992 ± 0.000	**0.9996 ± 0.0001**
	RF	0.9299	0.9810 ± 0.0090	0.9830 ± 0.0059	0.9795 ± 0.0075	0.9798 ± 0.0078	**0.9995 ± 0.0004**	0.9993 ± 0.0009	0.9993 ± 0.0002
	MLP	0.9847	0.9799 ± 0.0050	0.9819 ± 0.0053	0.9807 ± 0.0053	0.9803 ± 0.0052	0.9988 ± 0.0003	0.9982 ± 0.0002	**0.9989 ± 0.0002**
WDBC	SVM	0.9511	0.9501 ± 0.0067	0.9509 ± 0.0069	0.9429 ± 0.0078	0.9497 ± 0.0056	0.9962 ± 0.0016	0.9967 ± 0.0020	**0.9967 ± 0.0015**
	RF	0.9749	0.9670 ± 0.0079	0.9655 ± 0.0134	0.9692 ± 0.0070	0.9585 ± 0.0143	0.9978 ± 0.0006	0.9969 ± 0.0010	**0.9979 ± 0.0004**
	MLP	0.9068	0.9855 ± 0.0072	0.9860 ± 0.0038	0.9828 ± 0.0031	0.9775 ± 0.0092	0.9965 ± 0.0010	0.9995 ± 0.0001	**0.9998 ± 0.0001**
Wine	SVM	0.9739	0.9888 ± 0.0006	0.9869 ± 0.0003	0.9831 ± 0.0004	0.9866 ± 0.0015	0.9856 ± 0.0006	0.9957 ± 0.0005	**0.9973 ± 0.0007**
	RF	0.9863	0.9915 ± 0.0021	0.99910 ± 0.0014	0.9898 ± 0.0028	0.9905 ± 0.0021	0.9919 ± 0.0008	0.9949 ± 0.0005	**0.9966 ± 0.0006**
	MLP	0.9565	0.9947 ± 0.0017	0.9945 ± 0.0011	0.9937 ± 0.0029	0.9935 ± 0.0019	0.9848 ± 0.0030	0.9959 ± 0.0039	**0.9961 ± 0.0034**
Letter	SVM	0.5000	0.9550 ± 0.0037	0.9573 ± 0.0028	0.9416 ± 0.0008	0.9448 ± 0.0003	0.8713 ± 0.0499	0.9340 ± 0.0120	**0.9756 ± 0.0067**
	RF	0.4821	0.9564 ± 0.0022	0.9558 ± 0.0048	0.9538 ± 0.0053	0.9551 ± 0.0030	0.9793 ± 0.0199	0.9974 ± 0.0016	**0.9977 ± 0.0012**
	MLP	0.5000	0.9661 ± 0.0080	0.9728 ± 0.0083	0.9710 ± 0.0141	0.9749 ± 0.0063	0.9882 ± 0.0089	0.9965 ± 0.0014	**0.9973 ± 0.0016**
Surgery	SVM	0.5000	0.6300 ± 0.0345	**0.6469 ± 0.0171**	0.6228 ± 0.0381	0.6051 ± 0.0276	0.5861 ± 0.0419	0.5768 ± 0.0397	0.5827 ± 0.0254
	RF	0.4821	0.5271 ± 0.0288	0.5309 ± 0.0458	0.5128 ± 0.0287	0.5211 ± 0.0291	**0.6820 ± 0.0483**	0.6454 ± 0.0299	0.6720 ± 0.0191
	MLP	0.5000	0.5418 ± 0.0540	0.5941 ± 0.0556	0.5991 ± 0.0402	0.6233 ± 0.0355	0.5311 ± 0.0449	0.6221 ± 0.0530	**0.6685 ± 0.0604**
Yeast	SVM	0.5508	0.6779 ± 0.0087	0.6798 ± 0.0049	0.6732 ± 0.0055	0.6745 ± 0.0058	0.6532 ± 0.0586	0.6957 ± 0.0374	**0.7226 ± 0.0385**
	RF	0.6923	0.7112 ± 0.0156	0.7050 ± 0.0131	0.7050 ± 0.0177	0.7103 ± 0.0159	0.7711 ± 0.0040	**0.7728 ± 0.0041**	0.7720 ± 0.0055
	MLP	0.7021	0.7099 ± 0.0056	0.7125 ± 0.0097	0.7067 ± 0.0051	0.7012 ± 0.0113	0.7369 ± 0.0070	0.7662 ± 0.0187	**0.7794 ± 0.0043**

Bold values indicate the best AUC value among those of the comparison models

Table 6 Performance comparison of over-sampling methods and classification models with the others

Dataset	Model	Base	ROS	SMOTE	B-SMOTE	ADASYN	GAN	CGAN	BCGAN
CMC	SVM	0.5531	0.6369 ± 0.0049	0.6414 ± 0.0009	0.6405 ± 0.0011	0.6425 ± 0.0024	0.6401 ± 0.0159	0.6836 ± 0.0270	**0.6959 ± 0.0235**
	RF	0.6135	0.6229 ± 0.0114	0.6243 ± 0.0155	0.6134 ± 0.0130	0.6071 ± 0.0167	0.6532 ± 0.0063	0.6389 ± 0.0092	**0.6606 ± 0.0044**
	MLP	0.5561	0.6400 ± 0.0189	0.6321 ± 0.0157	0.6363 ± 0.0304	0.6313 ± 0.0184	0.6908 ± 0.0139	0.6959 ± 0.0110	**0.7067 ± 0.0119**
Card	SVM	0.9121	0.9510 ± 0.0010	**0.9543 ± 0.0001**	0.9528 ± 0.0001	0.9517 ± 0.0001	0.9415 ± 0.0002	0.9534 ± 0.0001	0.9534 ± 0.0001
	RF	0.8532	0.8592 ± 0.0013	0.8671 ± 0.0032	0.8491 ± 0.0020	0.8571 ± 0.0003	0.8570 ± 0.0023	0.8780 ± 0.0049	**0.9010 ± 0.0037**
	MLP	0.9329	0.9591 ± 0.0020	0.9613 ± 0.0032	0.9592 ± 0.0044	0.9643 ± 0.0021	0.9650 ± 0.0025	0.9710 ± 0.0037	**0.9750 ± 0.0027**
Email	SVM	0.4953	0.6312 ± 0.0445	0.6313 ± 0.0459	**0.6362 ± 0.0429**	0.6334 ± 0.0398	0.6292 ± 0.0547	0.6321 ± 0.0384	0.6324 ± 0.0375
	RF	0.5662	0.5623 ± 0.0285	0.5592 ± 0.0239	0.5621 ± 0.0301	0.5582 ± 0.0293	0.5631 ± 0.0409	0.5881 ± 0.0443	**0.5942 ± 0.0429**
	MLP	0.5644	0.6261 ± 0.0314	0.6154 ± 0.0398	0.6340 ± 0.0387	0.6313 ± 0.0428	0.6323 ± 0.0394	0.6372 ± 0.0420	**0.6471 ± 0.0359**
Tel	SVM	0.6972	0.7843 ± 0.0481	0.7844 ± 0.0325	**0.7854 ± 0.0309**	0.7823 ± 0.0315	0.6312 ± 0.0572	0.7791 ± 0.0429	0.7764 ± 0.0372
	RF	0.7710	0.7712 ± 0.0408	0.7633 ± 0.0284	0.7661 ± 0.0399	0.7671 ± 0.0319	0.7711 ± 0.0355	0.7731 ± 0.0322	**0.7784 ± 0.0319**
	MLP	0.7531	0.7790 ± 0.0325	0.7812 ± 0.0400	0.7811 ± 0.0387	0.7812 ± 0.0398	0.7793 ± 0.0453	0.7892 ± 0.0411	**0.7923 ± 0.0384**
Bio.fig	SVM	0.5212	0.5113 ± 0.0455	0.5114 ± 0.0400	**0.7311 ± 0.0533**	0.5513 ± 0.0498	0.5632 ± 0.0426	0.7011 ± 0.0479	0.7172 ± 0.0427
	RF	0.5231	0.7341 ± 0.0354	**0.7421 ± 0.0384**	0.7354 ± 0.0289	0.7322 ± 0.0387	0.7312 ± 0.0336	0.7320 ± 0.0301	0.7382 ± 0.0325
	MLP	0.5920	0.6963 ± 0.0410	0.7172 ± 0.0299	0.5431 ± 0.0424	0.7251 ± 0.0436	0.7231 ± 0.0420	0.7812 ± 0.0398	**0.7923 ± 0.0402**
Pay	SVM	0.7151	0.7290 ± 0.0281	0.7310 ± 0.0294	0.7371 ± 0.0216	0.7360 ± 0.0312	0.7191 ± 0.0373	0.7212 ± 0.0316	0.7244 ± 0.0317
	RF	0.7512	0.7491 ± 0.0288	0.5560 ± 0.0440	0.5213 ± 0.0298	0.5181 ± 0.0237	0.7062 ± 0.0447	0.7450 ± 0.0328	**0.7552 ± 0.0285**
	MLP	0.7582	0.7554 ± 0.0359	0.7652 ± 0.0378	0.7611 ± 0.0241	0.7693 ± 0.0311	0.7663 ± 0.0426	0.7752 ± 0.0290	**0.7831 ± 0.0298**

Bold values indicate the best AUC value among those of the comparison models

Appendix 2: Detailed results of Sect. 4.4

We present the more detailed results of the experiment in Sect. 4.4. Tables 5 and 6 show both averages and standard deviations of the obtained AUC. In each cell, the left value is the average AUC, and the right value is the standard deviation. We omit the standard deviations of the "Base" case because we did not conduct over-sampling in this case.

Acknowledgements This research was supported in part by Energy Cloud R&D Program (Grant Number: 2019M3F2A1073184) through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT and in part by Government-wide R&D Fund project for infectious disease research (GFID), Republic of Korea (Grant Number: HG19C0682).

References

1. Gong Y, Jia L (2019) Research on SVM environment performance of parallel computing based on large data set of machine learning. J Supercomput 75(9):5966–5983. https://doi.org/10.1007/s11227-019-02894-7
2. Garea AS, Heras DB, Argüello F (2019) Caffe CNN-based classification of hyperspectral images on GPU. J Supercomput 75(3):1065–1077. https://doi.org/10.1007/s11227-018-2300-2
3. Adewole KS, Han T, Wu W, Song H, Sangaiah AK (2020) Twitter spam account detection based on clustering and classification methods. J Supercomput 76(7):4802–4837. https://doi.org/10.1007/s11227-018-2641-x
4. Hasanin T, Khoshgoftaar TM, & Leevy JL (2019, July) A comparison of performance metrics with severely imbalanced network security big data. In: Proceedings of 2019 IEEE 20th international conference on information reuse and integration for data science (IRI). Los Angeles, CA, USA, pp 83–88. https://doi.org/10.1109/IRI.2019.00026.
5. O'Brien R, Ishwaran H (2019) A random forests quantile classifier for class imbalanced data. Pattern Recognit 90:232–249. https://doi.org/10.1016/j.patcog.2019.01.036
6. Haixiang G, Yijing L, Shang J, Mingyun G, Yuanyue H, Bing G (2017) Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl 73:220–239. https://doi.org/10.1016/j.eswa.2016.12.035
7. Stolfo SJ, Fan W, Lee W, Prodromidis A, Chan PK (2000, February) Cost-based modeling for fraud and intrusion detection: Results from the JAM project. In: Proceedings of the DARPA information survivability conference and exposition, DISCEX 2000. South Carolina, USA, pp 130–144. https://doi.org/10.1109/DISCEX.2000.821515
8. Ling CX, Li C (1998, August) Data mining for direct marketing: problems and solutions. In: Proceedings of the 4th international conference on knowledge discovery and data mining. AAAI Press, New York, NY, pp 73–79
9. Yang MY, Liao W, Li X, Cao Y, Rosenhahn B (2019) Vehicle detection in aerial images. Photogramm Eng Remote Sens 85(4):297–304. https://doi.org/10.14358/PERS.85.4.297
10. Del Gaudio R, Batista G, Branco A (2014) Coping with highly imbalanced datasets: a case study with definition extraction in a multilingual setting. Nat Lang Eng 20(3):327–359. https://doi.org/10.1017/S1351324912000381
11. Van Hulse J, Khoshgoftaar TM, Napolitano A (2007, June) Experimental perspectives on learning from imbalanced data. In: Proceedings of the 24th international conference on machine learning, Corvallis, Oregon, USA, pp 935–942
12. Liu XY, Wu J, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man Cybern B Cybern 39(2):539–550. https://doi.org/10.1109/TSMCB.2008.2007853
13. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: Synthetic minority over-sampling technique. J Artif Intell Res 16:321–357. https://doi.org/10.1613/jair.953

14. Liu T, Zhu X, Pedrycz W, Li Z (2020) A design of information granule-based under-sampling method in imbalanced data classification. Soft Comput. https://doi.org/10.1007/s00500-020-05023-2
15. He H, Bai Y, Garcia EA, Li S (2008, June) ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: Proceedings of the international joint conference on neural networks, Hong Kong, China, pp 1322–1328 https://doi.org/10.1109/IJCNN.2008.4633969
16. Han H, Wang WY, Mao BH (2005, August) Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In: Proceedings of the 2005 international conference on intelligent computing (ICIC'05), Lecture notes in computer science, Hefei, China, pp 878–887 https://doi.org/10.1007/11538059_91
17. Xie W, Liang G, Dong Z, Tan B, Zhang B (2019) An improved oversampling algorithm based on the samples selection strategy for classifying imbalanced data. Math Probl Eng 2019:3526539. https://doi.org/10.1155/2019/3526539
18. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Bengio Y (2014, December) Generative adversarial nets. In: Proceedings of 27th international conference on neural information processing systems, Montreal, Quebec, Canada, pp 2672–2680. https://doi.org/10.3156/jsoft.29.5_177_2
19. Fiore U, De Santis A, Perla F, Zanetti P, Palmieri F (2019) Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. Inf Sci 479:448–455. https://doi.org/10.1016/j.ins.2017.12.030
20. Liu J, Gu C, Wang J, Youn G, Kim JU (2019) Multi-scale multi-class conditional generative adversarial network for handwritten character generation. J Supercomput 75(4):1922–1940. https://doi.org/10.1007/s11227-017-2218-0
21. Guo J, Lu S, Cai H, Zhang W, Yu Y, Wang J (2018, February) Long text generation via adversarial training with leaked information. In: Proceedings of the 32nd AAAI conference on artificial intelligence, New Orleans, Louisiana, USA, AAAI, pp 1–27
22. Vondrick C, Pirsiavash H, Torralba A (2016, December) Generating videos with scene dynamics. In: Proceedings of 30th international conference on neural information processing system. Barcelona, Spain, pp 613–621
23. Mirza M, Osindero S (2014) Conditional generative adversarial nets. arXiv preprint http://arxiv.org/abs/1411.1784.
24. Douzas G, Bacao F (2018) Effective data generation for imbalanced learning using conditional generative adversarial networks. Expert Syst Appl 91:464–471. https://doi.org/10.1016/j.eswa.2017.09.030
25. Wang H Y (2008, June) Combination approach of SMOTE and biased-SVM for imbalanced datasets. In: Proceedings of the international joint conference on neural networks, Hong Kong, China, pp 228–231. https://doi.org/10.1109/IJCNN.2008.4633794
26. Hoi C H, Chan C H, Huang K, Lyu M R, King I (2004, July) Biased support vector machine for relevance feedback in image retrieval. In: Proceedings of the international joint conference on neural networks, Budapest, Hungary, pp 3189–3194. https://doi.org/10.1109/ijcnn.2004.1381186
27. Jo T, Japkowicz N (2004) Class imbalances versus small disjuncts. ACM SIGKDD Explor Newsl 6(1):40–49. https://doi.org/10.1145/1007730.1007737
28. Macia N, Bernadó-Mansilla E, Orriols-Puig A (2008, December) Preliminary approach on synthetic data sets generation based on class separability measure. In: Proceedings of the 19th international conference on pattern recognition, Tampa, USA, pp 1–4. https://doi.org/10.1109/icpr.2008.4761770.
29. Batista GEAPA, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 6(1):20–29. https://doi.org/10.1145/1007730.1007735
30. Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern 3:408–421. https://doi.org/10.1109/TSMC.1972.4309137
31. Tomek I (1976) Two modifications of CNN. IEEE Trans Syst Man Cybern 6:769–772. https://doi.org/10.1109/TSMC.1976.4309452
32. Liu Y, An A, Huang X (2006, April) Boosting prediction accuracy on imbalanced datasets with SVM ensembles. In: Proceedings of the tenth Pacific-Asia conference on knowledge discovery and data mining. Singapore, pp 107–118
33. Xie X, Xiong J, Lu L, Gui G, Yang J, Fan S, Li H (2020) Generative adversarial network-based credit card fraud detection. In: Liang Q, Liu X, Na Z, Wang W, Mu J, Zhang B (eds)

Communications, Signal Processing, and Systems. Springer, Singapore, pp 1007–1014. https://doi.org/10.1007/978-981-13-6508-9_122
34. Zhou Z, Zhang B, Lv Y, Shi T, Chang F (2019) Data augment in imbalanced learning based on generative adversarial networks. In: Gedeon T, Wong K, Lee M (eds) Neural Information Processing. Springer, Cham, pp 21–30. https://doi.org/10.1007/978-3-030-36808-1_3
35. Mariani G, Scheidegger F, Istrate R, Bekas C, Malossi C (2018) Bagan: data augmentation with balancing GAN. *arXiv preprint* http://arxiv.org/abs/1803.09655.
36. Bennett KP, Mangasarian OL (1992) Robust linear programming discrimination of two linearly inseparable sets. Optim Methods Softw 1(1):23–34. https://doi.org/10.1080/10556789208805504
37. Wolberg WH, Street WN, Mangasarian OL (1995) Image analysis and machine learning applied to breast cancer diagnosis and prognosis. Anal Quant Cytol Histol 17(2):77–87
38. Cortez P, Cerdeira A, Almeida F, Matos T, Reis J (2009) Modeling wine preferences by data mining from physicochemical properties. Decis Support Syst 47(4):547–553. https://doi.org/10.1016/j.dss.2009.05.016
39. Frey PW, Slate DJ (1991) Letter recognition using Holland-style adaptive classifiers. Mach Learn 6(2):161–182. https://doi.org/10.1023/A:1022606404104
40. Zięba M, Tomczak JM, Lubicz M, Świątek J (2014) Boosted SVM for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients. Appl Soft Comput J 14:99–108. https://doi.org/10.1016/j.asoc.2013.07.016
41. Horton P, Nakai K (1996, June) A probabilistic classification system for predicting the cellular localization sites of proteins. In: Proceedings of the fourth international conference on intelligent systems for molecular biology, St. Louis, USA. AAAI Press, pp 109–115
42. Lim TS, Loh WY, Shih YS (2000) Comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Mach Learn 40(3):203–228. https://doi.org/10.1023/A:1007608224229
43. Dal Pozzolo A (2015) Adaptive machine learning for credit card fraud detection. PhD thesis. https://doi.org/10.14419/ijet.v7i2.9356.
44. Hillstrom K (2017) Kevin Hillstrom: minethatdata project pricing http://www.minethatdata.com/ Accessed 8 January 2020.
45. Moro S, Cortez P, Rita P (2014) A data-driven approach to predict the success of bank telemarketing. Decision Support Syst 62:22–31. https://doi.org/10.1016/j.dss.2014.03.001
46. Schierz AC (2009) Virtual screening of bioassay data. J Cheminformatics 1(1):21. https://doi.org/10.1186/1758-2946-1-21
47. Yeh IC, Lien CH (2009) The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. Expert Syst Appl 36(2):2473–2480. https://doi.org/10.1016/j.eswa.2007.12.020
48. Nair V, Hinton G E (2010, June) Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th international conference on machine learning, Haifa, Israel, pp 807–814
49. Kingma D P, Ba J (2014) Adam: a method for stochastic optimization. *arXiv preprint* http://arxiv.org/abs/1412.6980.
50. Liaw A, Wiener M (2002) Classification and regression by random forest. R News 2(3):18–22
51. Haykin S (2010) Neural networks and learning machines, 3rd edn. Macmillan, New York
52. Corder GW, Foreman DI (2009) Nonparametric statistics for non-statisticians, 1st edn. John Wiley & Sons, Hoboken, New Jersey, USA
53. Li P, Li J, Chen Y, Pei Y, Fu G, Xie H (2020) Classification and recognition of computed tomography images using image reconstruction and information fusion methods. J Supercomput. https://doi.org/10.1007/s11227-020-03367-y
54. Abdi H, Williams LJ (2010) Principal component analysis. Wiley Interdisciplin Rev Comput Stat 2(4):433–459. https://doi.org/10.1002/wics.101