Distance-based Probabilistic Data Augmentation for Synthetic Minority Oversampling

JOEL GOODMAN, US Naval Research Laboratory
SHARHAM SARKANI and THOMAS MAZZUCHI, The George Washington University

Class imbalance can adversely affect the performance of machine learning for prediction and classification. One approach to address the class imbalance problem is synthetic minority oversampling. Oversampling approaches can be broadly categorized as either being structural or statistical in nature. Structural approaches generally have the advantage of identifying and oversampling those minority data points that best facilitate class separation, while statistical approaches model the underlying distribution from which the minority samples can be drawn. In this article, we formulate a distance-based approach that generates samples by both modeling the underlying minority class distribution and by geometrically considering those borderline samples entangled in the majority class. We demonstrate the efficacy of our approach operating on the Class-Imbalance data set from UCI by comparing its mean accuracy, AUC and F_1-score performance against both statistical and structural synthetic minority oversampling methods.

CCS Concepts: • **Information systems** → **Physical data models**; *Expert systems*; • **Mathematics of computing** → *Probabilistic representations*; **Probabilistic algorithms**; **Information theory;**

Additional Key Words and Phrases: Machine learning, class imbalance, data augmentation

ACM Reference format:
Joel Goodman, Sharham Sarkani, and Thomas Mazzuchi. 2022. Distance-based Probabilistic Data Augmentation for Synthetic Minority Oversampling. *ACM/IMS Trans. Data Sci.* 2, 4, Article 40 (May 2022), 18 pages.
https://doi.org/10.1145/3510834

1 INTRODUCTION

Imbalanced class distributions in machine learning can adversely effect the performance of classification and/or prediction in applications such as fraud prevention [14], intrusion detection [34], identification of rare diseases [33], and medical diagnosis [26]. Class imbalance occurs when one or more classes that form the *minority* group contains significantly fewer samples than the classes that make up the *majority* group [23]. When a class imbalance does exist, it is not uncommon for a machine learning algorithm trained on this data set to correctly categorize the majority class with near-perfect accuracy, while incorrectly categorizing the minority class [32]. Class imbalance is often addressed in the *clean* (or *scrub*) phase of a data science project [27, 43].

Authors' addresses: J. Goodman, US Naval Research Laboratory, 4555 Overlook Ave, Washington, DC 20375; email: joel.goodman@nrl.navy.mil; S. Sarkani and T. Mazzuchi, The George Washington University, 2121 I St NW, Washington, DC 20052; emails: {sarkani, mazzu}@gwu.edu.

2577-3224/2022/05-ART40 $15.00
https://doi.org/10.1145/3510834

The most common methods to address class imbalance are cost-sensitive learning [9, 11, 44], algorithmic-level approaches [19, 30, 50], data resampling [6, 13, 36], and hybrid techniques composed of some combination of the aforementioned methods [25, 28, 41, 49]. Cost-sensitive learning assigns a different weight to misclassification of minority samples than those from the majority class, and can operate at the resampling or algorithmic level. For example, in deep learning architectures, it's possible to separately weight the cross-entropy loss function used in backpropagation based on whether the training samples originated from the minority or majority class [17]. Algorithmic-level methods modify the classification algorithm to take into account the underlying class distribution. For instance, in [1] a random forest is used as the classifier, where the number of constituent classifiers is increased to represent the minority class in the ensemble to improve classification performance. Perhaps the most common and popular approach to address class imbalance is data resampling that includes both over- and undersampling techniques to equipoise the proportion of samples across all classes. For data sets where there are relatively few minority samples, or the majority class is not well represented, undersampling may remove important points in feature space that adversely effect downstream classification performance [3]. Oversampling methods are the alternative to undersampling and can be further subdivided into random oversampling and synthetic minority sample generation. Random oversampling replicates a select minority sample set to balance the distribution to any desired level. However, random oversampling may increase the likelihood of overfitting by making facsimiles of existing minority samples. So, although machine learning algorithms will make rules or construct decision boundaries that appear accurate during training, the performance of such sampling leads to poor classification performance on unseen samples during test and/or validation [24].

1.1 Structural Synthetic Minority Oversampling

An alternative to random resampling is synthetic oversampling, which represents the most commonly employed approach to address class imbalance [18]. One of the most popular and well cited synthetic minority oversampling techniques is SMOTE [7], which in the Python PyPi library has spawned 85 variants alone [31]. SMOTE generates synthetic minority samples along with the line segments to its k nearest neighbors based on the desired level of oversampling. A potential shortcoming of SMOTE is that it does not consider the possibility of generating unwanted samples along with the line segments that cross the majority class region [39]. There are a number of follow-ons to SMOTE, including Borderline-SMOTE [22], Safe-Level-SMOTE [5], and **Adaptive Synthetic sampling** (**ADASYN**) [20], which exemplify some of the most popular variants. Each of these methods are designed to either strengthen minority sample generation near the majority class boundary (Borderline SMOTE and ADASYN), or avoid inadvertently generating spurious samples in the majority class region (Safe-Level-SMOTE). Perhaps one of the highest performing recent SMOTE variants is **Majority Weighted Minority Oversampling Technique** (**MWMOTE**) [2]. MWMOTE iteratively identifies those minority samples closest to the border with the majority class. It then weights the samples by importance based on the following criteria: minority samples that are closer to the border of the majority class are more important than those further away, minority samples in sparse clusters are more important than those in dense ones, and samples near dense majority clusters are more important than those near sparse majority samples. Once a minority set S_{imin} is identified, samples are generated in a fashion that is nearly identical to that of SMOTE, with the distinct difference being that samples which have a higher weight have more samples generated around them. Although each of the algorithms has a stochastic component to them, these types of algorithms are structural in nature. That is, structural synthetic minority techniques generate samples proportionally based on some combination of their relative proximity to the majority class and the relative sample density in each region.

1.2 Statistical Synthetic Minority Oversampling

Another approach to generate synthetic minority samples, which is statistical in nature, is to try and model the underlying distribution of the minority class. Once the underlying distribution is determined, minority samples can be generated by drawing samples from this distribution. Some high-performing representative statistical approaches for generating synthetic minority samples include **rapidly converging Gibbs sampling** (**RACOG**) and **wrapper RACOG** (**wRACOG**) [10], **PDF oversampling** (**PDFOS**) [16] and **random walk oversampling** (**RWO**) [48]. The challenge associated with modeling an unknown distribution is the potentially high-dimensional feature space and sparse number of samples in the minority set. RACOG addresses this challenge by estimating the multidimensional distribution as the product of second-order conditional and marginal distributions using Chow-Liu trees to minimize the **Kullback–Leibler** (**KL**) divergence [4] with the true multidimensional distribution. Gibbs sampling using multiple Markov chains is then used to rapidly generate synthetic minority samples. An extension of the RACOG is wRACOG. The wrapper in wRACOG is a user defined machine learning algorithm. Once the standard deviation of the prediction and/or classification of the true positive rate of this algorithm reaches a predefined threshold, minority sample generation terminates. Using wRAGOG, it is no longer necessary to fix the number of iterations in Gibbs sampling needed to discard samples to avoid autocorrelation or to reach a stationary distribution. Another example of statistical synthetic minority oversampling is PDFOS, which directly tries to estimate the multidimensional distribution based on kernel density estimation of the minority class. However, it does so be making the simplifying assumption that a Gaussian multivariate distribution which best fits the minority data is an archetypical representative model. Here, synthetic minority oversampling is executed by taking the product of the (Cholesky) square root of the minority class covariance matrix with samples from a multidimensional standard normal distribution, and then adding this to the minority samples. This algorithm is also tightly coupled to radial basis kernel classification. RWO is a somewhat simpler statistical approach to synthetic minority sample generation. RWO independently estimates the mean and variance of each of the (one dimensional - or 1D) features. It then generates a set of Gaussian random variables whose mean and variance are the same as each of the minority features, and adds these to the minority points to generate minority samples. The objective of RWO is to preserve the mean and variance of each feature in the minority sample generation process. However, unlike RACOG or PDFOS, no attempt is made to model the potential statistical dependence between features of the underlying distribution in the sample generation process.

1.3 Distance-based Probabilistic Synthetic Minority Oversampling

The two preceding subsections outlined two distinct approaches to synthetic minority sample generation. The first, structural synthetic minority oversampling, uses the proximity and relative sample density with respect to the majority class as a part of the sample generation process. The second, statistical synthetic minority oversampling, models the underlying distribution of the minority sample set with varying degrees of fidelity, and then samples from this distribution to generate synthetic data. In this article, we present a **Distance-based Probabilistic Data Augmentation** (**DPDA**) technique that both models the underlying distribution and assesses the relative degree of entanglement of minority candidate samples in the majority class region. DPDA leverages an information-theoretic result relating the KL-divergence to the ratio of ***k* nearest neighbor** (***kNN***) distances [4] of the minority and synthetic samples. The statistical distribution of kNN distances is modeled, and from this candidate samples are generated. Synthetically generated candidates are chosen based on their statistical similarity to the distribution of minority class samples and their relative position with respect to the majority class. Furthermore, the KL-divergence is recursively

Table 1. Symbolic Notation Definitions

Symbol	Definition
$X = [X_1, \ldots, X_N]$	The set of minority class samples, where $X_i \in \mathbb{R}^D$ is a point in D-dimensional feature space
$\hat{X} = [\hat{X}_1, \ldots, \hat{X}_S]$	The set of synthetic minority class samples generated by DPDA, where $\hat{X}_i \in \mathbb{R}^D$
$\hat{X}_{1:s} = [\hat{X}_1, \ldots, \hat{X}_s]$	Subset of synthetic minority class samples generated by DPDA
$d_{kNN}(\hat{X}_i, \hat{X}_{\forall j \neq i})$	The $k-$Nearest Neighbor (Euclidean) distance of $\hat{X}_i$ w.r.t. all *other* members of the synthetic minority set
$d_{kNN}(\hat{X}_i, X)$	The $k-$Nearest Neighbor (Euclidean) distance of $\hat{X}_i$ w.r.t. all members of the minority set
$p(d_{kNN}(X))$	The estimated probability density of kNN distances in minority data set
$B(\hat{X}_i, \epsilon(i))$	Euclidean ball centered at $\hat{X}_i$ with radius $\epsilon(i)$

minimized by replacing the synthetically generated samples that have the highest negative impact on the divergence. Extensions to the algorithm are formulated to adaptively choose k in kNN and reduce bias in the KL-divergence estimation process when there is low minority sample support. DPDA represents a novel hybrid technique that combines both a structural and statistical approach to data augmentation.

The rest of this article is organized as follows. In Section 2, we derive the DPDA algorithm and present a detailed description of the steps involved in its execution. In Section 3, we compare the performance of DPDA against both structural and statistical approaches to data augmentation, and in Section 4, we conclude with a brief summary.

2 TECHNICAL APPROACH

DPDA is a multistep algorithm that starts by first generating synthetic samples that approximately follow the same distribution as the minority samples based on their "distance" from both one another and from the samples in the minority class. Once the candidate synthetic samples are generated, a test of significance is used to reject or accept samples based on the relative position of the candidates w.r.t. to both the majority and minority class. It then iteratively replaces a subset of the accepted synthetic samples by measuring the divergence between the minority and synthetic class distributions. In the succeeding paragraphs, we describe the DPDA algorithm where the symbolic notation that follows is defined in Table 1.

The DPDA algorithm construction leverages the following theorem:

Theorem 1. *Suppose $g(X)$ and $f(\hat{X})$ represent two multidimensional probability distributions associated with the minority class and synthetically generated minority samples, respectively, with Kullback-Leibler divergence measure $D_{KL}(f||g)$. Then the estimated KL-divergence*

$$\hat{D}_{KL}(f||g)) = \frac{D}{S}\sum_{i=1}^{S}\log\frac{d_{kNN}(\hat{X}_i, X)}{d_{kNN}(\hat{X}_i, \hat{X}_{\forall j\neq i})} + \log\frac{N}{S-1}, \tag{1}$$

is a consistent and unbiased divergence estimate such that

$$\lim_{N,S\to\infty}\mathbb{E}\left[\left(\hat{D}_{KL}(f||g) - D_{KL}(f||g)\right)^2\right] = 0, \tag{2}$$

where $\mathbb{E}(\cdot)$ represents the statistical expectation operator.

Proof. See [46] □

Equation (1), which is an application of the general result in [46], states that a consistent estimate of the divergence between the true minority class multivariate distribution and a synthetically generated one is directly related to kNN distances. Although not constructive, this does offer guidance in a generative approach for instantiating synthetic minority samples. In the limit, the choice of k in Equation (1) is immaterial, however, in practical applications with finite sample support, the choice of k will have an impact on the quality of the estimate. We consider both a fixed k equal to one ($k = 1$) to instantiate candidate samples and a data-dependent choice of k to measure statistical divergence in the process of generating synthetic minority samples. The following subsections describe in detail the sample generation process.

2.1 Fitting a $1NN$ Minority Sample Distance Distribution

To generate samples that roughly follow the same pattern of distances in feature space that minimizes $\hat{D}(f||g)$ in Equation (1), we first need to determine the probability distribution $p(d_{kNN}(X))$. This will be used in the next two sections to generate the initial candidate synthetic minority samples for $k = 1$, as well as a test of significance to accept or reject candidate synthetic samples. To determine $p(d_{1NN}(X))$, the N $d_{1NN}(X)$ Euclidean distances from the minority class samples are calculated from which we fit the distribution $p(d_{1NN}(X))$. There are a number of approaches to fitting a distribution, including **Expectation-Maximization** (**EM**) [12], Dirichlet Process Mixtures [38], and Kernel Density Estimation [37], each of which has its merits and associated computational considerations. However, once the distribution is fit, it is computationally expedient that this distribution facilitates a straightforward sampling strategy. For the results that follow, we chose an EM-based **Gaussian Mixture Model** (**GMM**)

$$p(d_{1NN}(X)) = \sum_{i=1}^{M} \pi_i \mathcal{N}\left(d_{1NN}(X); \mu_i, \sigma_i^2\right), \tag{3}$$

with a BIC optimized [15] number of mixing components M, and an EM optimized categorical probability π_i, mean μ_i and variance σ_i [45]. The form of Equation (3) facilitates a simple sampling strategy, first drawing a number from 1 to M with categorical probability π_i, and then sampling from the associated Gaussian.

2.2 Generating Synthetic Minority Sample Candidates

Using Equation (3), it is now possible to sample from this distribution to generate synthetic samples $\hat{X}$ whose $1NN$ distribution roughly mirrors that of the minority samples X. To that end, synthetic candidates are generated by *pivoting* over each of the minority samples X_i by adding an offset

$$\hat{X}_c = X_i + r_s \frac{\rho_c}{||\rho_c||_2}, \text{ for } c = 1, 2, \ldots, O \cdot N_C, \tag{4}$$

with

$$r_s \sim p(d_{1NN}(X)) \text{ and } \rho_c \sim \mathcal{N}(0, \mathrm{I}_D), \tag{5}$$

sampled from the fitted $1NN$ distribution in Equation (3) and a D-dimensional Gaussian distribution, respectively. In Equation (5), I_D represents the D-dimensional identity matrix and $||\cdot||_2$ the ℓ_2 norm. For each minority sample X_i, $O \cdot N_C$ synthetic samples are generated, where $O = \lceil S/N \rceil$ is the oversampling factor. From these samples, the O candidates whose kNN log-ratio in Equation (1) for $k = 1$ comes closest to 0 are selected, while the other $O \cdot (N_C - 1)$ are discarded, i.e.,

$$\hat{X}_s = \underset{\hat{X}_c \notin \hat{X}_{1:s-1}}{\arg\min} \log \left| \frac{d_{1NN}(\hat{X}_c, X)}{d_{1NN}(\hat{X}_c, \hat{X}_{1:s-1})} \right|, \tag{6}$$

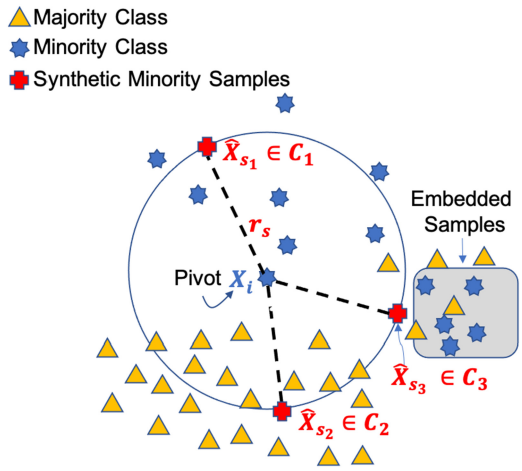


Fig. 1. Various outcomes from synthetic minority sample generation.

for $s = 1, 2, \ldots, O$. The candidates selected using Equation (6) will closely follow samples pulled from a distribution to minimize Equation (1), although in the steps that follow there is further refinement of the minimization of the divergence. A geometric interpretation of Equations (4)–(6) is that samples are drawn from the surface of a hypersphere with radius r_s at the angles $\hat{} = [\theta_1, \theta_2, \ldots, \theta_{D-1}]$ that are independent and uniformly distributed over $[0, 2\pi)^{D-1}$ [35], and whose origin is the pivot point, $X_i \in \mathbb{R}^D$.

2.3 Candidate Acceptance and Rejection

The combination of a randomly sampled offset with spherical radius r_s at angle $\hat{}$ from the pivot could potentially place a synthetic sample $\hat{X}_s$ into the region of space principally populated by the majority class, even though the pivot X_i's nearest neighbor is from the minority class, or vice versa. These situations are illustrated in the toy 2D example of Figure 1. Referring to this figure, consider the three cases—C_1, C_2, and C_3—where a synthetic sample is generated at distance r_s from the pivot. For case C_1, the (1D) angular offset $\hat{}$ places the minority sample $\hat{X}_{s_1}$ squarely in the minority class region. A minority class region is one in which there are no majority class samples using a kNN classifier, for $k > 1$. We will return to the choice of k in the next few sections. A synthetically generated minority sample $\hat{X}_{s_2}$ belongs to C_2 when its k nearest neighbors are all from the majority class, and $\hat{X}_{s_3}$ belongs to C_3 when its k nearest neighbors are split across both the minority and majority classes. Using the observation that sample distance is congruous with a measure of similarity, a one-sided test of significance is used to accept or reject candidate samples, where the null hypothesis is that the candidate sample's $1NN$ distance is consistent with the distances from the underlying $1NN$ distribution $p(d_{1NN}(X))$. Defining

$$p(d_{1NN}(X) \leq \beta_u) = 1 - \alpha, \tag{7}$$

where α is the significance of the test and β_u is the value from which the tail of the distribution integrates to α, then a candidate synthetic sample is accepted with binary probability $P_A \in \{0, 1\}$, such that

$$P_A = \begin{cases} 1 & \text{if } C_1 \vee d_{1NN}(\hat{X}_s, X_{\forall j \neq i}) \leq \beta_u \\ 0 & \text{otherwise} \end{cases}. \tag{8}$$

The intuition behind Equation (8) is that if a candidate is associated with C_1, then this sample reflects a "safe" candidate that is statistically similar to the other clearly differentiable minority samples. If a synthetically generated sample belongs to C_2 or C_3, its acceptance is conditioned on its $1NN$ distance not being statistically different than the largest $1NN$ distances of the samples from the original minority class. For example, $\hat{X}_{s_2}$ in Figure 1 has a low probability of acceptance given that its $1NN$ distance (with the pivot removed from the set X) is likely to be greater than most if not all of $d_{1NN}(X)$. This prevents spurious *noise-like* samples from being generated. On the other hand, $\hat{X}_{s_3}$ has a $1NN$ distance similar to those of existing minority samples belonging to C_3, so there is a good chance that this candidate will be accepted.

2.4 KL-Divergence and Iterative Sample Replacement

With a complete set of synthetic samples, it is now possible to both measure how closely the distribution of synthetic samples matches the distributions of the original minority class and iteratively replace synthetic samples to more closely align the two distributions. Indeed, and without loss of generality, consider the set of ratios in Equation (1) with values that have been sorted in descending order such that

$$R(\hat{X}_1) \geq R(\hat{X}_2) \geq \cdots \geq R(\hat{X}_S) \in \mathcal{R}, \tag{9}$$

where

$$R(\hat{X}_i) = \log \left| \frac{d_{kNN}(\hat{X}_i, X)}{d_{kNN}(\hat{X}_i, \hat{X}_{\forall j \neq i})} \right|. \tag{10}$$

The set of samples $[\hat{X}_1, \ldots, \hat{X}_n]$ in Equation (9)

$$\mathcal{R}_n = [R(\hat{X}_1), R(\hat{X}_2), \ldots, R(\hat{X}_n)], \tag{11}$$

represent the n ratios in Equation (1) with the largest values, with $|\mathcal{R}_n| = n$. The samples $\hat{X}_i$ that are a function of $R(\hat{X}_i) \in \mathcal{R}_n$ are chosen for replacement, repeating the steps starting from Equation (4). The choice of the cardinality n is nominally set to 10% of the total number S of synthetic samples generated, and iterations terminate once the change in the KL-divergence is small or a max number of iterations is reached.

2.5 Selection of k and Bias Reduction

The divergence measure in Equation (1) suffers from bias when there is low sample support, and here we briefly discuss two approaches to reduce that bias. One approach is a modification of Equation (1) in which $R(\hat{X}_i)$ in Equation (10) is replaced with

$$R(\hat{X}_i) = \log \frac{d_{k_i NN}(\hat{X}_i, X)}{d_{l_i NN}(\hat{X}_i, \hat{X}_{\forall j \neq i})} + \frac{1}{DS}(\psi(k_i) - \psi(l_i)), \tag{12}$$

where k_i and l_i are data-adaptive choices for the number of nearest neighbors and $\psi(\cdot)$ is the digamma function [42]. Both k_i and l_i are chosen by counting the number of samples from the sets $\hat{X}$ and X, respectively, contained in the ball $B(\hat{X}_i, \epsilon(i))$, where $\epsilon(i)$ is defined as

$$\epsilon(i) = \max\{\min_{\forall j \neq i}(\hat{X}_i - \hat{X}_j), \min_{\forall j}(\hat{X}_i - X_j))\}, \tag{13}$$

with a proof outlined in [46]. An alternative to Equation (12) is to continue using $R(\hat{X}_i)$ as defined in Equation (10) and to choose a larger fixed value of k, e.g., k=4, by cross-validation. Both bias reduction techniques in the context of the results are discussed in Section 3.

ALGORITHM 1: Generate Synthetic Minority Sample Set $\hat{X}$

Require: Input k for kNN, Number of Synthetic Samples S, Candidates Per Point Factor N_C, α, and *MaxIterations*
Calculate $\rightarrow d_{1NN}(X)$;
Fit distribution $\rightarrow p(d_{1NN}(X))$; Equation (3)
Calculate oversampling factor $\rightarrow O = \lceil S/N \rceil$;
Select all minority samples as *pivots*, X_i for $i = 1$ to N;
while iteration < *MaxIterations* **do**
  for i = 1 to number of *pivots* **do**
    Draw $N_C \cdot O$ samples from $p(d_{1NN}(X))$ and $\mathcal{N}(0, \mathrm{I}_D)$
    Multiply $N_C \cdot O$ samples point-by-point; Equation (5)
    Add $N_C \cdot O$ samples to pivot X_i; Equation (4)
    Select O samples by optimizing divergence; Equation (6)
  end for
  Determine kNN neighbors of candidates $\hat{X}$;
  Form cases C_i and accept/reject candidates; Equation (8)
  Compute KL distance; Equations (10), (11)
  Pick n highest ratios and associated *pivots*;
end while
Return $\hat{X}_i$ for $i = 1$ to S

2.6 Extensions

It is instructive to consider two modifications to Equation (8), the first being that even the synthetic samples belonging to C_1 are subject to a test of significance, and the second, that the distribution of $1NN$ distances be conditioned on the case to which they belong. That is, Equation (8) is modified such that

$$P_A = \begin{cases} 1 & \text{if } d_{1NN}(\hat{X}_s, X_{\forall j \neq i} | C_i) \leq \beta_u, \\ 0 & \text{otherwise} \end{cases}, \tag{14}$$

where $d_{1NN}(\hat{X}_s, X_{\forall j \neq i} | C_i)$ for $i \in \{1, 2, 3\}$ is the $1NN$ distance of the candidates to the minority class samples excluding the pivot, conditioned on the case to which the candidate belongs. The idea behind Equation (14) is twofold. First, as opposed to admitting all C_1 candidates, those whose $1NN$ distance is statistically greater than most of its neighbors are excluded, thereby putting more emphasis on the border samples, $\{C_2, C_3\}$. Second, the probability of $1NN$ distances is now conditioned on the particular case that the candidate belongs to. This reflects the prior belief that candidates entangled in the minority class may follow a somewhat different distribution than those candidates only embedded in minority class regions. The approach to placing more emphasis on the border samples, which is reflected in many other synthetic minority sampling approaches [21], can also be realized by simply adopting an adaptive choice to the significance of the test. That is, in Equation (14) α is replaced with α_i, with each α_i associated with case C_i. By increasing the value of α_1 with respect to α_2 and α_3, it is likely a higher proportion of border samples will be present in final synthetic set of samples.

2.7 DPDA Algorithm Summary

The DPDA pseudocode is summarized in Algorithm 1. The parameters needed to seed the algorithm include the value of k to determine the k nearest neighbors, the number S of synthetic samples to generate, a factor associated with oversampling N_C, and a maximum number of

iterations indicating when to stop replacing samples. The first step in DPDA is to calculate the $1NN$ distances from the original minority class and fit a distribution using these distances. The $N_C \times O$ number of candidate samples per pivot point are generated by sampling from a unit hypersphere around the pivot whose radius is pulled from the $1NN$ distribution of minority samples. From these, O candidates are selected after minimizing the divergence between the synthetic and minority sample distributions. Once the candidates are generated, the region of space (in terms of cases C_i) based on their kNN neighbors in the full original data set are identified. From this, a test of significance based on their $1NN$ distance to the original minority samples (sans pivot) conditioned on the Case C_i for $i \in \{1, 2, 3\}$ is used to accept or reject candidate samples.

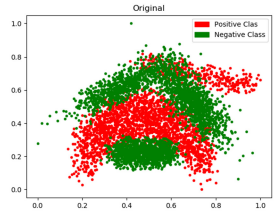
The choice of α forms the basis of the hypothesis test in Equation (8) to determine if the synthetically generated sample should be accepted based on its distance to other minority samples. The larger the value of α, the smaller the value of the distance β_u needed to pass the test. Because this test only applies when the synthetically generated sample falls into the majority border region, the higher the value of α, the less likely it is that a spurious noise-like sample is accepted. Conversely, larger values of α lead to lower diversity. We have found in practice that a significance level of $\alpha = 0.05$ is a good comprise between sample diversity and spurious sample generation, and this is the value that we used to generate the results presented in Section 3.

We briefly note that in practice the oversampling factor O is chosen to account for the fact that some samples will be rejected. Using the accepted candidates, the pivots associated with n candidates samples which have the highest negative impact on the KL-divergence are then used to resample a smaller fraction of candidates. This process continues until the maximum number of iterations is reached. Empirically, we have found that selecting n to be 10% of the size of the minority set, i.e., $n = 0.1 \times S$, works well in practice as it is a good balance between time-to-solution and classification performance.

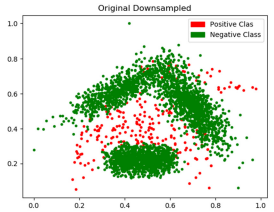
Extensions to Algorithm 1 are as follows. First, Equations (3) and (8) can be modified (e.g., see Equation (14)) to explicitly condition candidate acceptance on the case C_i that it is associated with, and to choose a test of significance α_i specifically tailored to that case (see Section 2.6). Further, the choice of k when measuring divergence in Equations (8) and (11) can be adaptively determined on a sample-by-sample basis using Equation (12) to reduce bias when there is low sample support. Finally, the maximum number of iterations can be replaced by measuring machine learning performance, that is, after generating minority samples, the cross-validated performance of the classification or prediction algorithm can be measured and when a desired level of performance is reached and/or the change in performance from iteration-to-iteration is small, the process is terminated.

3 EXPERIMENTAL RESULTS

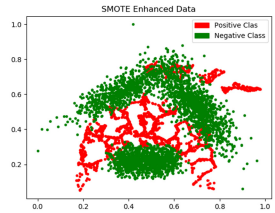
To test the efficacy of DPDA, we made use of two data sets. The first is the so-called banana data set from Google's Kaggle standard classification library [40]. The banana data is two-dimensional and highly balanced with two categories, a *positive* class and *negative* class. The purpose of using this data set was principally for visualization purposes, given that the low dimensional (2D) nature of the feature space enabled us to conceptualize how the various minority oversampling methods generated synthetic samples. To artificially instantiate a minority class, we randomly undersampled the *positive* class so that it had roughly ten times fewer samples than the *negative* class. To generate synthetic samples, we used the R package "*imbalance*" whose library consisted of the following minority oversampling routines: {RACOG, PDFOS, MWMWOTE, and RWO}, which represents state of the art probabilistic approaches to synthetic minority oversampling as described in Section 1. The DPDA algorithm was developed in Python version 3.7.8 and made use of the



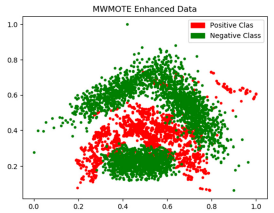
(a) Original



(b) Original Downsampled

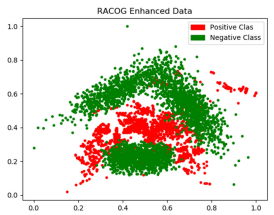


(c) SMOTE Oversampling

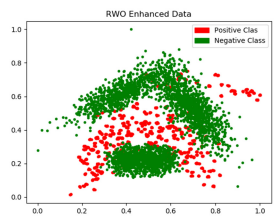


(d) MWMOTE Oversampling

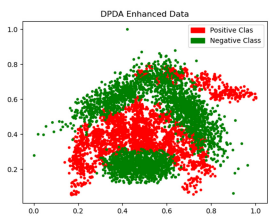
(e) PDFOS Oversampling



(f) RACOG Oversampling



(g) RWO Oversampling



(h) DPDA Oversampling

Fig. 2. Synthetic minority oversampling of the Banana data set.

scikit-learn version 0.23 library for reporting out classification results. We also used the Python *imbalanced-learn* library for obtaining results for SMOTE.

The banana data has 5,299 two-dimensional features, of which 2,923 belonged to the *negative* class, and the remaining 2,376 were associated with the *positive* class. We randomly removed 2,126 *positive* features so that the undersampled *positive* class consisted of 250 two-dimensional samples.

The 2,126 remaining *positive* class samples, as well as a random selection of 250 *negative* class samples, were then set aside and used for validation. Unlike the routines in the R *imbalance* package, the SMOTE implementation in the *imbalanced-learn* library was only capable of oversampling

Table 2. Comparative kNN Classification Performance on Banana Data Set Using Synthetically Generated Data

Banana Data Set			
Augmentation	Mean Accuracy	Precision/Recall	AUC
Baseline	0.6180	P: 0.6108/R: 0.7249	0.8529
SMOTE	0.7888	0.6649/0.8717	0.9258
DPDA	**0.8574**	0.7094/0.9115	**0.9503**
RACOG	0.7938	0.6635/0.8690	0.9247
PDFOS	0.8252	0.6730/0.8781	0.9329
MWMOTE	0.8210	0.6814/0.8894	0.9338
RWO	0.7955	0.6649/0.8717	0.9252

Those values highlighted in bold indicate highest group performance, and those in red second highest. Note that unlike the oversampling approaches, the Baseline did not use use any synthetically generated data.

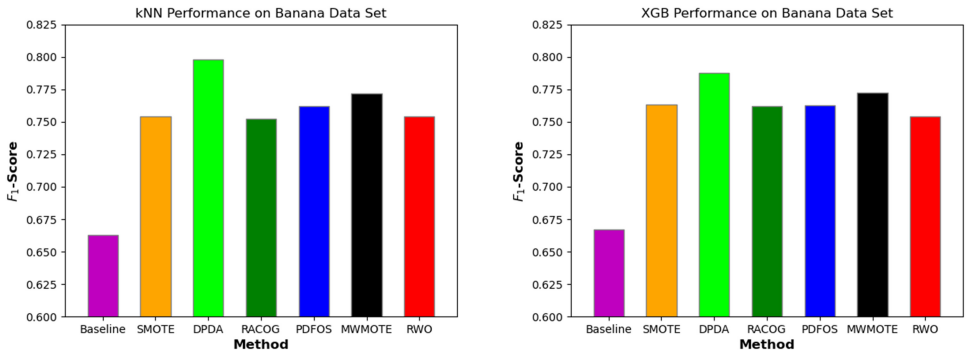
Table 3. Comparative XGBoost Classification Performance on Banana Data Set Using Augmented Data

Banana Data Set			
Augmentation	Mean Accuracy	Precision/Recall	AUC
Baseline	0.6358	P: 0.6070/R: 0.7390	0.8682
SMOTE	0.8110	0.6740/0.8803	0.9417
DPDA	**0.8653**	0.7096/0.8842	**0.9577**
RACOG	0.8102	0.6730/0.8781	0.9411
PDFOS	0.8127	0.6740/0.8777	0.9465
MWMOTE	0.8394	0.6889/0.8785	0.9573
RWO	0.8344	0.6649/0.8717	0.9400

Those values highlighted in bold indicate highest group performance, and those in red second highest Note that unlike the oversampling approaches, the Baseline did not use any synthetically generated data.

the minority set until it achieved parity with the number of samples in the majority class. Accordingly, 2,423 *positive* class synthetic samples were generated by each of the oversampling routines. A visual comparison of the synthetic minority samples generated is plotted in Figure 2. Figure 2(a) represents the original data set, and Figure 2(b) the downsampled data set after removing all but 250 *positive* class samples. Figure 2(c)–(h) represents the synthetically generated data using each of the approaches from the R and Python class imbalance libraries, as well as DPDA. Qualitatively, it appears that each of the minority oversampling approaches did a reasonable job reconstituting the original *positive* class. Both SMOTE and RWO, which represent both a structural and statistical approach to oversampling, visually appear to leave the largest gaps in feature space. PDFOS covers the feature space well but does so by inserting more minority samples into the majority class region. MWMOTE, RACOG, and DPDA appear to generate samples that more closely represent the shape of the original *positive* class.

To test the efficacy of the oversampling approaches on the banana data set, we chose to use two classification algorithms, kNN with $k = 4$ and XGBoost [8]. These algorithms were chosen to represent both non-axis aligned (kNN) and axis-aligned (XGBoost) approaches to classification. The classification process consisted of both a training and testing phase, with separate data sets for each. For training, 2,423 *negative* samples from the original minority data set and 2,423 *positive*



(a) kNN F_1 Performance (b) XGB F_1 Performance

Fig. 3. Relative F1 performance on the banana dataset.

synthetically generated samples generated by each of the minority oversampling methods was used. For testing, we used the 2,126 original *positive* samples and 250 randomly selected *negative* samples from the undersampled banana data set (Figure, 2b). The metrics used to evaluate classification performance were mean accuracy, which was the number of correct classifications divided by the total number of test samples, as well the **area under the curve** (**AUC**). AUC is calculated by determining the area under the true positive (TP) vs. the false positive (FP) rate curve, and corresponds to the degree of separability between classes. We included AUC as it is a more robust measure of classification performance for imbalanced data sets [47]. Finally, for completeness, we measured the precision and recall defined as

$$\begin{aligned} Precision &= \frac{TP}{TP + FP}, \\ Recall &= \frac{TP}{TP + FN}, \end{aligned} \tag{15}$$

where FN corresponds to false negatives, and from which the harmonic mean - or F_1 score

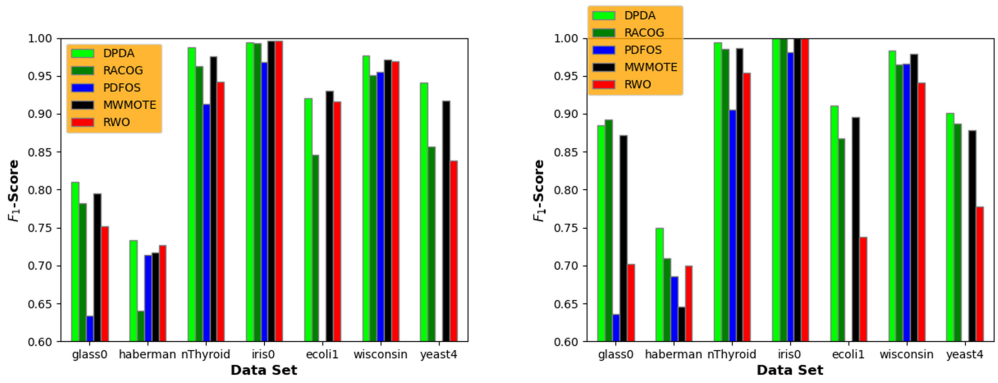
$$F_1 = \frac{2}{1/Precision + 1/Recall}, \tag{16}$$

can be calculated. Using these metrics, the results for both kNN and XGBoost classification are listed in Tables 2 and 3, respectively, as well as the F_1 performance plotted in Figure 3. For the Baseline line approach. we used the same data that was used for all the other approaches with the exception of synthetic data so that the total number of *positive* class samples used for training were the original 250 that were left after downsampling. The purpose of the Baseline was to measure the relative improvement in performance when using synthetically generated data. In all cases, DPDA had both the highest mean categorical identification and AUC for both kNN- and XGBoost-based classification (highlighted in bold), while MWMOTE and PDFOS came in a second (highlighted in red). This combined with the F_1 scores plotted in Figure 3 corroborated the visual assessment of synthetic minority sample generation for both DPDA and MWMOTE, but PDFOS was a bit of a surprise as it seemed to generate many samples outside of the minority class region.

The second data set used was the class imbalance library from the Keel repository, originating from the **University of California Irvine's** (**UCI's**) Machine Learning Data Repository [29]. The Class Imbalance library consisted of seven data sets:{*glass0, haberman, newthyroid1, iris0, ecoli1, wisconsin, yeast4*}, each of which contained a minority class. The number of dimensions (features), size of the data set and the number of minority class samples are listed in Table 4. For each of the

Table 4. Parameters of the Keel Class Imbalance Data Set

Keel Data Set Files	Features	Total Samples	Minority Samples
glass0	10	214	70
haberman	4	306	81
newThyroid1	6	215	35
iris0	5	150	50
ecoli1	8	336	77
wisconsin	10	683	289
yeast4	9	1,484	59



(a) kNN F_1 Performance (b) XGB F_1 Performance

Fig. 4. Relative F_1 performance on UCI keel dataset.

data sets in the Keel Class Imbalance library, we generated an equal number of synthetic samples such that the minority and majority classes were balanced. So, for example, the *yeast4* data set was composed of 1,484 nine-dimensional samples (e.g., $\mathbb{R}^{1484\times9}$), of which, 1,433 samples were from the *negative* class and 53 samples from the *positive* class. In this case, 1,433 synthetic minority samples were generated such that there were an equal number of samples in all classes. Following the same procedure for testing the relative performance of each method as [2, 7], and [10], we used k-fold cross-validation with $k = 10$ to generate the results that are both tabulated and plotted below. Because MWMOTE is an extension of SMOTE with better performance on the classification tasks that follow, SMOTE was excluded from the following performance comparisons.

Using the 10-fold cross-validated results, the mean performance listed in Tables 5 and 6 were tabulated and F_1 performance plotted in Figure 4. In all cases, the standard deviation of the performance results across all algorithms and data sets was no greater than 0.015, and in most cases was less than 10^{-3}. A brief note regarding one the algorithms in the *imbalance* library. The PDFOS algorithm tries to find an inverse for the sample covariance matrix, and if the matrix is rank deficient (uninvertible), no synthetic data is generated. For two of the data sets, *yeast4* and *ecoli1* in the Keel library, an inverse could not be found by PDFOS. This is reflected in in Tables 5 and 6, by marking this as an **Error** condition. Although all the oversampling algorithms did reasonably well, DPDA was in the top two for both mean accuracy, F_1 score and AUC across all data sets, sans *iris0*. As detailed in Algorithm 1, the only hyperparameters needed for DPDA were the value of k

Table 5. Classification Accuracy of Synthetically Augmented Data Using kNN

Dataset	DPDA	RACOG	PDFOS	MWMOTE	RWO
Mean Accuracy					
glass0	**0.8700**	0.8300	0.6767	0.8200	0.7933
haberman	**0.7847**	0.6903	0.7641	0.7738	0.7791
newThyroid1	**0.9914**	0.9721	0.9339	0.9836	0.9571
iris0	0.9960	0.9955	0.9760	0.9965	**0.9970**
ecoli1	0.9476	0.8928	**Error**	**0.9574**	0.9370
wisconsin	**0.9865**	0.9702	0.9732	0.9835	0.9810
yeast4	**0.9637**	0.9007	**Error**	0.9451	0.8947
AUC					
glass0	**0.8025**	0.7715	0.6512	0.7908	0.7489
haberman	**0.7286**	0.6512	0.7123	0.7154	0.7243
newThyroid1	**0.9880**	0.9589	0.9062	0.9752	0.9314
iris0	0.9927	0.9925	0.9658	**0.9962**	0.9953
ecoli1	0.9063	0.8277	**Error**	**0.9162**	0.9048
wisconsin	**0.9704**	0.9398	0.9464	0.9652	0.9617
yeast4	**0.9256**	0.8416	**Error**	0.9007	0.8181
Precision					
glass0	0.8025	0.7715	0.6512	0.7908	0.7489
haberman	0.7286	0.6512	0.7123	0.7154	0.7243
newThyroid1	0.9880	0.9589	0.9062	0.9752	0.9314
iris0	0.9927	0.9925	0.9658	0.9962	0.9953
ecoli1	0.9063	0.8277	**Error**	0.9162	0.9048
wisconsin	0.9704	0.9398	0.9464	0.9652	0.9617
yeast4	0.9256	0.8416	**Error**	0.9007	0.8181
Recall					
glass0	0.8575	0.8247	0.6807	0.8203	0.7877
haberman	0.7860	0.6925	0.7616	0.7661	0.7784
newThyroid1	0.9888	0.9729	0.9309	0.9799	0.9625
iris0	0.9970	0.9954	0.9769	0.9969	0.9972
ecoli1	0.9478	0.8939	**Error**	0.9550	0.9377
wisconsin	0.9855	0.9686	0.9706	0.9825	0.9811
yeast4	0.9661	0.8912	**Error**	0.9468	0.8959

Those values highlighted in bold indicate highest group performance, and those in red second highest.

for kNN, and the number of samples to generate. In fact, if a value of k were not given, a default value of 4 is applied, which is the value used for all the results that were generated. We found that the bias reduction and adaptively choosing k in the process of measuring KL-divergence resulted in no noticeable change in performance. This may be an artifact of the data sets tested against. DPDA, MWMOTE, and RACOG all appeared to generate samples that resulted in relatively consistent results that were close to one another across all the data sets. Again, referring to Figure 2, this was not a surprise given qualitatively all three minority oversampling routines appeared to best preserve the shape of the artificially generated minority class from the banana data set.

Table 6. Classification Accuracy of Synthetically Augmented Data Using XGBoost Classification

Dataset	DPDA	RACOG	PDFOS	MWMOTE	RWO
Mean Accuracy					
glass0	0.9067	**0.9300**	0.7133	0.8933	0.7333
haberman	**0.7991**	0.7531	0.7419	0.6978	0.7625
newThyroid1	**0.9961**	0.9896	0.9307	0.9904	0.9679
iris0	**1.0000**	**1.0000**	0.9870	**1.0000**	**1.0000**
ecoli1	**0.9426**	0.9068	**Error**	0.9282	0.7878
wisconsin	**0.9902**	0.9800	0.9802	0.9855	0.9623
yeast4	**0.9367**	0.9180	**Error**	0.9229	0.8343
AUC					
glass0	0.9654	0.9639	0.8520	**0.9787**	0.8856
haberman	**0.8973**	0.8804	0.8814	0.8284	0.8758
newThyroid1	0.9989	**0.9993**	0.9770	0.9961	0.9877
iris0	**1.0000**	**1.0000**	0.9952	**1.0000**	**1.0000**
ecoli1	**0.9733**	0.9547	**Error**	0.9680	0.8867
wisconsin	**0.9940**	0.9932	0.9931	0.9930	0.9780
yeast4	**0.9748**	0.9554	**Error**	0.9745	0.9158
Precision					
glass0	0.8843	0.8699	0.6537	0.8546	0.7012
haberman	0.7468	0.7136	0.6906	0.6551	0.6986
newThyroid1	0.9933	0.9838	0.8967	0.9848	0.9458
iris0	1.0000	1.0000	0.9779	1.0000	1.0000
ecoli1	0.8974	0.8503	**Error**	0.8814	0.7328
wisconsin	0.9791	0.9534	0.9572	0.9737	0.9300
yeast4	0.8838	0.8724	**Error**	0.8593	0.7663
Recall					
glass0	0.8933	0.9405	0.7144	0.9145	0.7438
haberman	0.7959	0.7544	0.7517	0.6978	0.7612
newThyroid1	0.9951	0.9890	0.9264	0.9901	0.9687
iris0	1.0000	1.0000	0.9870	1.0000	1.0000
ecoli1	0.9374	0.9132	**Error**	0.9268	0.7860
wisconsin	0.9896	0.9826	0.9807	0.9857	0.9586
yeast4	0.9374	0.9147	**Error**	0.9254	0.8275

Those values highlighted in bold indicate highest group performance, and those in red second highest.

Finally, the computational complexity of DPDA is dominated by the *kNN* calculations used in both modeling and divergence calculations, as well as the iterations as defined by Equations (9)–(11). Because the distribution that was modeled in Equation (3) is one-dimensional, the density estimation was a relatively efficient process. Qualitatively, the structural approaches ran much more quickly than the statistical approaches using the Keel Class Imbalance data sets using a MacBook Pro with an I7 processor and 16 GB of memory, with DPDA having the penultimate runtime next to RACOG which had the noticeably longest runtime. These runtimes are plotted in Figure 5 and represent the cumulative time in log-seconds (base 10) for generating synthetic minority samples for all 7-datasets in the Keel library. It is interesting to note that even though

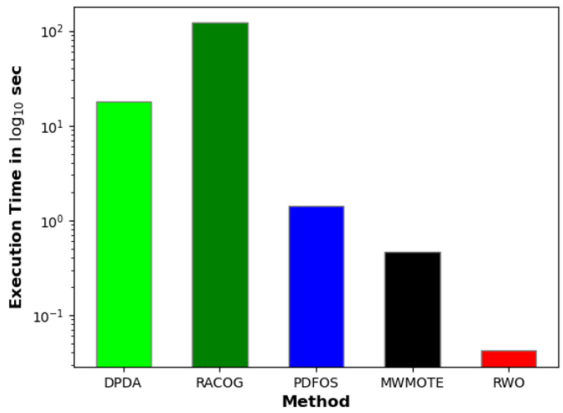


Fig. 5. Comparative runtimes of various methods on UCI keel data set.

DPDA had one of the longest execution times, it also tended to have the best overall performance and was competitive in all cases with the top performers across all of the datasets.

4 SUMMARY

Synthetic minority oversampling to address class imbalance is commonly conducted using one of two techniques, structural or statistical. Structural minority oversampling uses the proximity and relative sample density with respect to the majority class as a part of the sample generation process, while statistical approaches model the underlying distribution. This article details a DPDA technique that both, directly and indirectly, models the underlying statistical distribution while considering the geometric arrangement of the minority and majority classes. We presented both the intuition behind the generation of samples, and compared performance of DPDA against other popular and recently developed high-performance structural and statistical synthetic minority oversampling techniques using the Keel Class Imbalance library. In almost all cases, DPDA either outperformed or was a close second in classification performance using synthetic samples, offering a very competitive heterogeneous alternative to the homogeneous structural and statistical approaches to data augmentation.

REFERENCES

[1] Mohammed Bader-El-Den, Eleman Teitei, and Todd Perry. 2019. Biased random forest for dealing with the class imbalance problem. *IEEE Transactions on Neural Networks and Learning Systems* 30, 7 (2019), 2163–2172. DOI: https://doi.org/10.1109/TNNLS.2018.2878400

[2] Sukarna Barua, Monirul Islam, Xin Yao, and Kazuyuki Murase. 2014. MWMOTE–Majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering* 26, 2 (2014), 405–425. DOI: https://doi.org/10.1109/TKDE.2012.232

[3] Gustavo E. Batista, Ronaldo C. Prati, and Maria Carolina Monard. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter* 6, 1 (2004). DOI: https://doi.org/10.1145/1007730.1007735

[4] Christopher M. Bishop. 2007. *Pattern Recognition and Machine Learning*. Springer Information Science and Statistics Series.

[5] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. 2009. Safe-Level-SMOTE: Safe-Level-Synthetic minority over-sampling technique for handling the class imbalanced problem. In *Proceedings of the Advances in Knowledge Discovery and Data Mining*. Springer, Berlin, 475–482. DOI: https://doi.org/10.1007/978-3-642-01307-2_43

[6] Hong Cao, Xiao Li, David Yew-Kong Woon, and See Ng. 2013. Integrated oversampling for imbalanced time series classification. *IEEE Transactions on Knowledge and Data Engineering* 25, 12 (2013), 2809–2822. DOI: https://doi.org/10.1109/TKDE.2013.37

[7] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357. DOI: https://doi.org/10.5555/1622407.1622416

[8] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery. 785–794. DOI: https://doi.org/10.1145/2939672.2939785

[9] Fanyong Cheng, Jing Zhang, and Cuihong Wen. 2016. Cost-Sensitive large margin distribution machine for classification of imbalanced data. *Pattern Recognition Letters* 80, C (2016), 107–112. DOI: https://doi.org/10.1016/j.patrec.2016.06.009

[10] Barnan Das, Narayanan Krishnan, and Diane Cook. 2015. RACOG and wRACOG: Two probabilistic oversampling techniques. *IEEE Transactions on Knowledge and Data Engineering* 27, 1 (2015), 222–234. DOI: https://doi.org/10.1109/TKDE.2014.2324567

[11] Shounak Datta and Swagatam Das. 2015. Near-Bayesian support vector machines for imbalanced data classification with equal or unequal misclassification costs. *Pattern Recognition Letters* 70 (2015), 39–52. DOI: https://doi.org/10.1016/j.neunet.2015.06.005

[12] Arthur P. Dempster, Nan M. Laird, and Donald.B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* 39, 1 (1977), 1–38.

[13] Annarita D'Addabbo and Rosalia Maglietta. 2015. Parallel selective sampling method for imbalanced and large data classification. *Pattern Recognition Letters* 62 (2015), 61–67. DOI: https://doi.org/10.1016/j.patrec.2015.05.008

[14] Tom Fawcett and Foster Provost. 1997. Adaptive fraud detection. *Data Mining and Knowledge Discovery* 1, 3 (1997), 291–316. DOI: https://doi.org/10.1023/A:1009700419189

[15] Chris Fraley and Adrian Raftery. 1998. How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal* 41, 8 (1998), 578–588. DOI: https://doi.org/10.1093/comjnl/41.8.578

[16] Ming Gao, Xia Hong, Sheng Chen, Chris J. Harris, and Emad Khalaf. 2014. PDFOS: PDF estimation based over-sampling for imbalanced two-class problems. *Neurocomputing* 138 (2014), 248–259. DOI: https://doi.org/10.1016/j.neucom.2014.02.006

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. Retrieved from http://www.deeplearningbook.org.

[18] Anjana Gosain and Saanchi Sardana. 2017. Handling class imbalance problem using oversampling techniques: A review. In *Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics*. 79–85. DOI: https://doi.org/10.1109/ICACCI.2017.8125820

[19] Hongyu Guo and Herna L. Viktor. 2004. Learning from imbalanced data sets with boosting and data generation: The DataBoost-IM approach. *ACM SIGKDD Explorations Newsletter* 6, 1 (2004), 30–39. DOI: https://doi.org/10.1145/1007730.1007736

[20] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. 1322–1328. DOI: https://doi.org/10.1109/IJCNN.2008.4633969

[21] Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. 2017. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications* 73 (2017), 220–239. DOI: https://doi.org/10.1016/j.eswa.2016.12.035

[22] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. 2005. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the Advances in Intelligent Computing*. Springer, Berlin, 878–887. DOI: https://doi.org/10.1007/11538059_91

[23] Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering* 21, 9 (2009), 1263–1284. DOI: https://doi.org/10.1109/TKDE.2008.239

[24] Robert C. Holte, Liane E. Acker, and Bruce W. Porter. 1989. Concept learning and the problem of small disjuncts. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*. 813–818.

[25] Shengguo Hu, Yanfeng Liang, Lintao Ma, and Ying He. 2009. MSMOTE: Improving classification performance when training data is imbalanced. In *Proceedings of the 2009 2nd International Workshop on Computer Science and Engineering*. 13–17. DOI: https://doi.org/10.1109/WCSE.2009.756

[26] Anju Jain, Saroj Ratnoo, and Dinesh Kumar. 2017. Addressing class imbalance problem in medical diagnosis: A genetic algorithm approach. In *Proceedings of the 2017 International Conference on Information, Communication, Instrumentation and Control*. 1–8.

[27] Jeroen Janssens. 2014. *Data Science at the Command Line: Facing the Future with Time-Tested Tools*. O'Reilly Media.

[28] Piyasak Jeatrakul, Kok Wai Wong, and Chun Che Fung. 2010. Classification of imbalanced data by combining the complementary neural network and SMOTE algorithm. In *Proceedings of the 17th International Conference on Neural Information Processing: Models and Applications*. 152–159. DOI: https://doi.org/10.1007/978-3-642-17534-3_19

[29] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera. 2020. Supervised Classification Library. (2020). Retrieved 2020-7-11 from https://sci2s.ugr.es/keel/datasets.php.

[30] Byungju Kim and Junmo Kim. 2020. Adjusting decision boundary for class imbalanced learning. *IEEE Access* 8 (2020), 81674–81685. DOI: https://doi.org/10.1109/ACCESS.2020.2991231

[31] György Kovács. 2019. Smote-variants: A python implementation of 85 minority oversampling techniques. *Neurocomputing* 366 (2019), 352–354. DOI: https://doi.org/10.1016/j.neucom.2019.06.100

[32] Joffrey L. Leevy, Taghi M. Khoshgoftaar, Richard A. Bauder, and Naeem Seliya. 2018. A survey on addressing high-class imbalance in big data. *Springer Journal of Big Data* 42, 5 (2018), 1–30.

[33] Xuedong Li, Yue Wang, Dongwu Wang, Walter Yuan, Dezhong Peng, and Qiaozhu Me. 2019. Improving rare disease classification using imperfect knowledge graph. In *Proceedings of the BMC Medical Information and Decision Making*. DOI: https://doi.org/doi.org/10.1186/s12911-019-0938-1

[34] Joseph W. Mikhail, John M. Fossaceca, and Ronald Iammartino. 2019. A semi-boosted nested model with sensitivity-based weighted binarization for multi-domain network intrusion. *ACM Transactions on Intelligent Systems Technology* 10, 3 (2019), 1–27. DOI: https://doi.org/10.1145/3313778

[35] Robb J. Muirhead. 2005. *Aspects of Multivariate Statistical Theory*. Wiley, New York.

[36] Iman Nekooeimehr and Susana K. Lai-Yuen. 2019. Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets. *Elsevier Expert Systems with Applications* 46 (2019), 405–416. DOI: https://doi.org/10.1016/j.eswa.2015.10.031

[37] Emmanuel Parzen. 1962. On estimation of the probability density function and mode. *The Annals of Mathematical Statistics* 3, 33 (1962), 56–65.

[38] Carl E. Rasmussen. 2000. The infinite gaussian mixture model. In *Proceedings of the Advances in Neural Information Processing Systems*. 554–600.

[39] José A. Sáez, Julián Luengo, Jerzy Stefanowski, and Francisco Herrera. 2014. Managing borderline and noisy examples in imbalanced classification by combining SMOTE with ensemble filtering. In *Proceedings of the Intelligent Data Engineering and Automated Learning – IDEAL 2014*. Emilio Corchado, José A. Lozano, Héctor Quintián, and Hujun Yin (Eds.), Springer International Publishing, Cham, 61–68. DOI: https://doi.org/10.1007/978-3-319-10840-7_8

[40] Saravanan Jaichandaran. 2020. Standard Classification Library Banana Data Set. (2020). Retrieved 2020-6-25 from https://www.kaggle.com/saranchandar/standard-classification-with-banana-dataset.

[41] Chris Seiffert, Taghi Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. 2010. RUSBoost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 40, 1 (2010), 185–197. DOI: https://doi.org/10.1109/TSMCA.2009.2029559

[42] H. M. Srivastava and Junesang Choi. 2012. *Zeta and q-Zeta Functions and Associated Series and Integrals*. Elsevier. DOI: https://doi.org/10.1016/C2010-0-67023-4

[43] Victoria Stodden. 2020. The data science life cycle: A disciplined approach to advancing data science as a science. *Communications of the ACM* 63, 7 (2020), 58–66. DOI: https://doi.org/10.1145/3360646

[44] Boyu Wang and Joelie Pineau. 2016. Online bagging and boosting for imbalanced data streams. *IEEE Transactions on Knowledge and Data Engineering* 28, 12 (2016), 3353–3366. DOI: https://doi.org/10.1109/TKDE.2016.2609424

[45] Jixin Wang, Zhenyu Wang, ChaoYang, Naixiang Wang, and XiangjunYu. 2011. Optimization of the number of components in the mixed-model using multi-criteria decision making. In *Proceedings of the Applied Mathmatical Modeling*. Elsevier. DOI: https://doi.org/10.1016/j.apm.2011.11.053

[46] Qing Wang, Sanjeev R. Kulkarni, and Sergio Verdu. 2009. Divergence estimation for multidimensional densities via k-Nearest-Neighbor distances. *IEEE Transactions on Information Theory* 55, 5 (2009), 2392–2405. DOI: https://doi.org/10.1109/TIT.2009.2016060

[47] Shuo Wang and Xin Yao. 2013. Relationships between diversity of classification ensembles and single-class performance measures. *IEEE Transactions on Knowledge and Data Engineering* 25, 1 (2013), 206–219. DOI: https://doi.org/10.1109/TKDE.2011.207

[48] Huaxiang Zhang and Mingfang Li. 2014. RWO-Sampling: A random walk over-sampling approach to imbalanced data classification. *Information Fusion* 20 (2014), 99–116. DOI: https://doi.org/10.1016/j.inffus.2013.12.003

[49] Nai-Nan Zhang, Shao-Zhen Ye, and Ting-Ying Chien. 2018. Imbalanced data classification based on hybrid methods. In *Proceedings of the 2nd International Conference on Big Data Research*. 16–20. DOI: https://doi.org/10.1145/3291801.3291812

[50] Tingting Zhou, Wei Liu, Congyu Zhou, and Leiting Chen. 2018. GAN-based semi-supervised for imbalanced data classification. In *Proceedings of the 2018 4th International Conference on Information Management*. 17–21. DOI: https://doi.org/10.1109/INFOMAN.2018.8392662

Received January 2021; revised June 2021; accepted January 2022