Kernel-Based Enhanced Oversampling Method for Imbalanced Classification

Wenjie Li R130026076@ALUMNI.UIC.EDU.CN
Guangdong Provincial/Zhuhai Key Laboratory of IRADS
Department of Statistics and Data Science, Beijing Normal-Hong Kong Baptist University

Hanlin Wang HL.WANG@CONNECT.UST.HK
Department of Computer Science & Engineering, HKUST

Sibo Zhu R130026221@MAIL.UIC.EDU.CN
Guangdong Provincial/Zhuhai Key Laboratory of IRADS
Department of Statistics and Data Science, Beijing Normal-Hong Kong Baptist University

Zhijian Li* ZHIJIANLI@UIC.EDU.CN
Guangdong Provincial/Zhuhai Key Laboratory of IRADS
Department of Statistics and Data Science, Beijing Normal-Hong Kong Baptist University

Editors: Hung-yi Lee and Tongliang Liu

Abstract

This paper introduces Kernel-Weighted SMOTE (KWSMOTE), an oversampling method for imbalanced classification. KWSMOTE enhances the traditional SMOTE algorithm by employing a kernel-based weighting scheme to prioritize closer neighbors, with a convex combination that ensures the generated samples are geometrically bounded. This dual-mechanism approach generates synthetic samples that better represent the minority class. Through experiments on multiple real-world datasets, we demonstrate that KWSMOTE outperforms strong baselines in terms of F1-score, G-mean, and AUC, providing an effective solution for handling imbalanced datasets in classification tasks.

Keywords: Imbalanced datasets, SMOTE, Classification, Sampling method, Machine learning

1. Introduction

Imbalanced datasets are an important issue in classification, where the distribution of classes is skewed, with the minority class has far fewer examples than the majority class. Most learning algorithms optimize overall accuracy, which can lead to favoring the majority class. Consequently, it results in a bias in which the model performs well in the majority class but poorly in the minority class, which is often the class of greater interest (Rawat and Mishra, 2022).

This problem appears in a wide range of real-world applications. For example, in fraud detection, fraudulent transactions are rare compared to legitimate ones, but accurately identifying these minority instances is crucial (Ali et al., 2022). Similarly, in medical diagnosis, diseases such as cancer are much less common than healthy outcomes, but the cost of missing a diagnosis is extremely high (Jiang et al., 2023). In many scenarios, the presence

* Corresponding author.

of an imbalanced dataset not only diminishes the effectiveness of the classification model but also raises concerns about its reliability and fairness.

Several strategies address class imbalance: data sampling and cost-sensitive learning. In sampling, oversampling adds minority examples and under-sampling removes majority examples. Oversampling can cause overfitting if new points are too similar to existing ones (Chawla et al., 2002), while under-sampling may discard useful information (Lin et al., 2017). Cost-sensitive learning adjusts the learning algorithm, assigning higher penalties to errors on the minority class, encouraging the model to focus more on rare cases (Feng et al., 2020).

Many real-world tasks still deal with small or medium tabular datasets. On such data, non-deep learning models like SVM, Random Forest, and GBM often match or exceed deep networks. They train quickly, require fewer hyperparameters, and are less data-hungry. In this setting, oversampling methods such as SMOTE remain very useful because they directly shape the training set seen by the classifier. KWSMOTE is designed for this scenario. Compared to SMOTE, KWSMOTE generates more realistic synthetic points by combining a kernel-based weighting scheme with a convex combination of local minority neighbors.

2. Related Work

Many approaches have been proposed for imbalanced learning. Over the past two decades, various surveys and literature reviews have organized and examined the distinct features and characteristics of these methods (Japkowicz and Stephen, 2002; Chawla et al., 2004; Su and Tsai, 2011; He and Garcia, 2009; Chawla, 2010; Branco et al., 2016; Wang and Li, 2024). This is mainly due to the continuous emergence of new real-world applications, where data imbalance is inherent (Haixiang et al., 2017). Recently, many imbalanced learning techniques have been integrated into Python libraries like *imblearn* (LemaÃŽtre et al., 2017), make many of these methods easy to use.

One notable approach to addressing class imbalance is the Synthetic Minority Oversampling Technique (SMOTE), proposed by (Chawla et al., 2002). Essentially, SMOTE generates new samples by performing random linear interpolation between minority class samples and their nearest neighbors. Creating a specified number of synthetic minority samples reduces the data imbalance ratio, thereby enhancing the classification performance on imbalanced datasets. The detailed steps of SMOTE are outlined as follows:

1. For each minority class sample x_i with $i = 1, 2, \ldots, n$, compute the Euclidean distances to all other minority class samples to determine the k nearest neighbors.

2. Based on the desired oversampling rate N, randomly select m neighbors from the k nearest neighbors of each sample x_i. Denote these neighbors as x_{ij} with $j = 1, 2, \ldots, m$.

3. For each selected neighbor sample x_{ij}, create a new synthetic sample p_{ij} using the formula:

$$p_{ij} = x_i + \text{rand}(0, 1) \times (x_{ij} - x_i), \tag{1}$$

where rand$(0, 1)$ is a random value generated from a uniform distribution between 0 and 1. Continue this process until the dataset achieves the desired balance ratio.

Many researchers have highlighted enhancements to the original SMOTE algorithm, focusing on different methods for generating synthetic samples. These modifications aim to better address the challenges posed by imbalanced datasets, making the improved versions more effective and suitable for various data imbalance situations.

One limitation of the classic SMOTE algorithm is that it cannot properly handle boundary samples. SMOTE generates synthetic samples by linear interpolation between minority samples and their neighbors, resulting in marginalization at the edges of the data distribution (Wang et al., 2021). Data marginalization can blur the separation between classes, making classification more challenging. An improved SMOTE algorithm based on normal distribution was proposed to reduce the generation of edge samples by concentrating synthetic data near the minority class center. The new synthetic samples p_i are generated using the following formula:

$$p_i = x'_i + \text{rnorm}(\mu = 1, \sigma) \cdot (x'_{\text{center}} - x'_i), \tag{2}$$

where rnorm$(\mu = 1, \sigma)$ is a random number drawn from a normal distribution with mean $\mu = 1$ and adjustable standard deviation σ. Compared to the uniform distribution in SMOTE, the use of normal weight ensures that the generated samples are more likely to cluster near the minority class center, thereby better preserving the internal structure of the minority class and preventing boundary samples from becoming overly marginalized.

Another drawback of SMOTE is that its synthetic samples are confined to the line segment between seed samples, which fails to faithfully capture the true distribution of the data. To address this limitation, the SNOCC (Sigma Nearest Oversampling based on Convex Combination) method was proposed (Zheng et al., 2015). This approach extends the concept of linear combinations, allowing the generated synthetic samples to be distributed more broadly within the sample space, not only on the line segment. By expanding to multiple seed samples, SNOCC enables new samples to be distributed within the convex hull formed by the original samples. However, (Zheng et al., 2015) doesn't provide any theoretical support and doesn't detail how to choose the seed number.

To better handle samples near the class boundary, several variants have been proposed. Borderline-SMOTE (Han et al., 2005) focuses on generating synthetic samples only from "borderline" minority instances—those that are most likely to be misclassified. The algorithm first identifies minority samples where more than half of their k-nearest neighbors belong to the majority class. By applying the SMOTE generation mechanism exclusively to these borderline points, it strengthens the decision boundary and avoids adding noise deep within the minority class region. Similarly, SVM-SMOTE (Nguyen et al., 2011) utilizes a Support Vector Machine (SVM) to identify the decision boundary more precisely. It generates synthetic samples by interpolating between minority class support vectors and their neighbors, effectively populating the critical, hard-to-classify regions near the boundary.

Another approach to overcome the linearity constraint is Gaussian-SMOTE. Instead of interpolating between two samples, this method generates new instances by adding a small amount of Gaussian noise to an existing minority sample. A new sample p_i is created from a seed minority sample x_i by adding a randomly generated vector from a multivariate

normal distribution, where the covariance matrix is estimated from the minority class data. This allows the synthetic samples to form a "cloud" around the original instances rather than being restricted to straight lines, potentially creating a more diverse and realistic representation of the minority class's underlying distribution (Blagus and Lusa, 2013).

Beyond data sampling, recent deep long-tailed methods optimize losses or representations, e.g., LA (Menon et al., 2020), PaCo (Cui et al., 2021), BCL (Zhu et al., 2022), DDC (Wang et al., 2023), GPaCo (Cui et al., 2023), and DirMixE (Yang et al., 2024). These works target deep networks and are complementary to our focus on data-level oversampling for classical models.

In summary, SMOTE and its variations have been widely used to address imbalanced datasets, but they still face limitations. Interpolation along line segments can miss the true data geometry, and boundary marginalization can blur class separability. Our approach aims to improve upon these drawbacks of SMOTE by developing a more adaptive oversampling technique that better replicates the true distribution of minority classes, preserves class boundaries, and enhances classification performance on imbalanced data.

3. Methodology

Before we formally elaborate on our methodology, several definitions are presented below.

Definition 1 *A convex set is a set Ω in a vector space such that, for any two points $x_1, x_2 \in \Omega$, and any $w \in [0,1]$, the point $wx_1 + (1-w)x_2$ also belongs to Ω.*

Definition 2 *A convex combination of points $\{x_0, x_1, \ldots, x_k\}$ is any point x that can be expressed as:*

$$x = \sum_{j=0}^{k} w_j x_j,$$

where each $w_j \geq 0$ and $\sum_{j=0}^{k} w_j = 1$.

Let $S_i = \{0, 1, \ldots, k\}$, J_i be a subset of S_i with ℓ elements, i.e. $J_i \subset \{1, \ldots, k\}$ with $|J_i| = \ell$. In our kernel-weighted SMOTE (KWSMOTE), we propose the following method to generate new samples $x_{i,\text{new}}$,

$$x_{i,\text{new}} = \sum_{j \in J_i} \frac{w_{ij}}{D_i} x_i^{(j)} \;=\; x_i + \frac{1}{D_i} \sum_{j \in J_i} w_{ij} \big(x_i^{(j)} - x_i\big). \tag{3}$$

Here x_i is the selected seed minority point; $x_i^{(j)}$ is its j-th nearest minority neighbor with $x_i^{(0)} = x_i$; w_{ij} is the kernel weight; and $D_i = \sum_{j \in J_i} w_{ij}$ is the normalizer. By Definition 2, this yields a convex combination:

$$x_{i,\text{new}} = \sum_{j \in J_i} \nu_{ij}\, x_i^{(j)}, \qquad \nu_{ij} = \frac{w_{ij}}{D_i} \geq 0, \;\; \sum_{j \in J_i} \nu_{ij} = 1. \tag{4}$$

Consider a set of points within the k-th nearest neighbor point of x_i, we can show that this is a convex set.

Theorem 3 *The set*

$$\Omega_{ik} = \{x \mid \|x - x_i\| \leq L_{ik}\}$$

is a convex set, where $L_{ik} = \|x_i^{(k)} - x_i\|$.

Proof Let x, y be any two elements in Ω_{ik} and $w \in [0, 1]$. Define $z = wx + (1 - w)y$. We need to show that $z \in \Omega_{ik}$. Using the triangle inequality and properties of norms, we have:

$$\begin{aligned}
\|z - x_i\| &= \|w(x - x_i) + (1 - w)(y - x_i)\| \\
&\leq w\|x - x_i\| + (1 - w)\|y - x_i\| \\
&\leq wL_{ik} + (1 - w)L_{ik} \\
&= (w + 1 - w)\, L_{ik} \\
&= L_{ik}.
\end{aligned}$$

Therefore, $z \in \Omega_{ik}$, and so Ω_{ik} is convex. ■

Theorem 4 *If* Ω *is a convex set, then every convex combination of points in* Ω *is also in* Ω.

Proof

Base Case $(n = 1)$**:**

For $n = 1$, the convex combination reduces to a single point $x = x_1$, where the weight is $w_1 = 1$. Since $x_1 \in \Omega$, it follows that $x \in \Omega$.

Inductive Step:

Assume that any convex combination of k points in Ω is also in Ω. For $k + 1$ points $x_1, x_2, \ldots, x_{k+1} \in \Omega$ with weights $w_1, w_2, \ldots, w_{k+1} \geq 0$ such that $\sum_{i=1}^{k+1} w_i = 1$, define $s = \sum_{i=1}^{k} w_i$.

If $s = 0$, then $w_{k+1} = 1$, and so $x = x_{k+1} \in \Omega$.

If $s > 0$, let

$$y = \frac{1}{s} \sum_{i=1}^{k} w_i x_i.$$

Since $\sum_{i=1}^{k} w_i/s = 1$ and each $w_i/s \geq 0$, y is a convex combination of k points in Ω. By the induction hypothesis, $y \in \Omega$.

Now, express x as a convex combination of y and x_{k+1}:

$$x = sy + w_{k+1}x_{k+1}.$$

Given that $s + w_{k+1} = 1$ and s, $w_{k+1} \geq 0$, and $y, x_{k+1} \in \Omega$, the convexity of Ω implies that $x \in \Omega$.

Therefore, by induction, any convex combination of points in Ω is also in Ω. ■

By Theorem 4, each synthetic sample is a convex combination of the seed x_i and its ℓ selected minority neighbors; hence it lies in the convex hull of this *local* set C_i. Because $C_i \subseteq \Omega_{ik} = \{x : \|x - x_i\| \leq L_{ik}\}$ (Theorem 3), the generated point is also contained

in the local k-NN ball. Therefore, the parameter ℓ is a hyperparameter that defines how many neighbors are randomly selected from the k nearest neighbors to form the convex combination, which can enhance the diversity of the synthetic data by controlling it.

The next issue we need to consider is the selection of parameter k and the weight w_{ij}. It is natural that if we select a large k, more data points will be considered. However, if the selected point or its neighbors contain noise or are distant, the newly generated points will also be affected. Therefore, we propose using the Gaussian kernel function as the weight

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right).$$

The kernel weight value depends on the similarity between two points. A higher similarity results in a larger weight for a nearby point, giving it a greater influence, while a more distant point will have a smaller weight, reducing its impact. Noise and outliers typically result in negligible weights as the Gaussian kernel decays toward zero rapidly when the two input vectors are far apart. As distant points and even outliers have low effects, the kernel method avoids new points from being generated near the boundaries between classes, which helps to reduce the likelihood of blurring the boundary between minority and majority classes, as presented in Fig. 1.

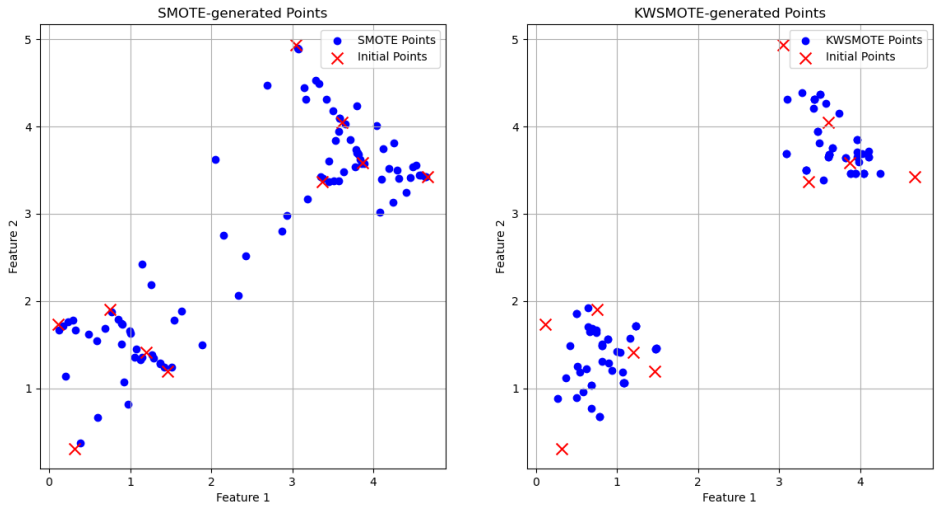

Figure 1: Left: SMOTE spreads synthetic points along straight segments and may populate sparse boundary regions. Right: KWSMOTE, via kernel-weighted convex combinations, concentrates samples within the local minority neighborhood, empirically reducing boundary intrusion.

Applying the Gaussian kernel, the convex combination (4) can be written as:

$$x_{i,\text{new}} = \sum_{j \in J_i} \frac{K(x_i, x_i^{(j)})}{\sum_{u \in J_i} K(x_i, x_i^{(u)})} x_i^{(j)}. \tag{5}$$

In Eq. (4), we set $\nu_{ij} \propto K(x_i, x_i^{(j)})$ with the Gaussian kernel $K(x, x') = \exp(-\|x - x'\|^2/(2\sigma^2))$. Because $K(x_i, x_i) = 1$ and $K(\cdot, \cdot)$ decays exponentially with distance, the

seed x_i receives the largest coefficient and distant neighbors contribute negligibly. Thus the synthetic sample is a barycenter inside the convex hull of the local set C_i and, by Theorems 3–4, remains within the local k-NN ball, which empirically mitigates boundary marginalization (Fig. 1).

If all selected neighbors are too far from the seed, their kernel weights become uniformly small. Operationally, when $\max_{j \in J_i} w_{ij} < \tau$, we skip the generation for this seed. This practical rule prevents synthesizing points around isolated or noisy seeds where no informative minority neighbor exists.

With the noise resistance in place, the number of nearest neighbors k can be adjusted to a larger value without worrying about the impact of outliers. Moreover, the parameter k and the bandwidth σ of the Gaussian kernel can be finely tuned. By conducting hyperparameter optimization, we can obtain a set of parameters better suited to the dataset, allowing for the generation of more realistic points and improving the classifier's performance.

Still, The proposed method maintains essentially the same computational complexity as SMOTE. Let $n_{\min}$ denote the number of minority samples, d the feature dimension, and n_{gen} the number of synthetic points to be generated. The overall complexity of KWSMOTE is only marginally higher than that of the classic SMOTE algorithm. Both methods rely on finding the k nearest neighbors for each minority sample, which dominates the total cost and scales as $O(n_{\min} d \log n_{\min})$ using tree-based search (or $O(n_{\min} d)$ with approximate methods). After the neighbor search is cached, the remaining operations in KWSMOTE consist of computing kernel weights and performing a convex combination of ℓ selected neighbors, yielding an additional $O(n_{\text{gen}} \ell)$ cost. This is the same order as the interpolation step in SMOTE, where synthetic points are generated along line segments.

To summarize, KWSMOTE's boundary-aware behavior emerges from (i) convex combinations restricted to a local k-NN ball and (ii) exponentially decaying kernel weights that keep the barycenter close to the seed. Together with a simple weight-threshold skip rule, this design empirically mitigates boundary marginalization without adding extra complexity. The pseudocode of the algorithm is given in Algorithm 1.

Algorithm 1 KWSMOTE Algorithm

Input: Dataset (X, y), Number of neighbors k, Number of points in the convex combination ℓ, Threshold τ, Kernel width σ (optional)

Output: Resampled dataset $(X_{resampled}, y_{resampled})$

Identify minority class y_{min} and majority class y_{maj}

Compute the number of samples to generate: $n_{samples\_to\_generate} = |y_{maj}| - |y_{min}|$

Extract minority class samples $X_{min} \leftarrow X[y = y_{min}]$

Compute k nearest neighbors for each $x \in X_{min}$

if σ is None **then**

  Compute $\sigma = \sqrt{\frac{\mathrm{Var}(X_{min}) \times n_{features}}{2}}$ (default setting in scikit-learn's SVM)

end if

Initialize empty list *new_samples*

while length(new_samples) $<$ *n_samples_to_generate* **do**

  Randomly select a minority sample $x_i \in X_{min}$

  Find its k nearest neighbors $x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(k)}$

  Randomly select ℓ neighbors from these k nearest neighbors, and denote the index as $J_{i\ell} = \{j|$ if $x_i^{(j)}$ is selected $\}$

  Compute Gaussian kernel weights for each neighbor in $x_i^{(j)}$, $j \in J_{i\ell}$:

$$w_{ij} = \exp\left(\frac{-\|x_i - x_i^{(j)}\|^2}{2\sigma^2}\right)$$

  if $\max(w_j) < \tau$ **then**

    Skip this sample

  end if

  Generate new sample:

$$x_{i,\text{new}} = \sum_{j \in J_{i\ell}} \frac{w_{ij}}{\sum_{u \in J_{i\ell}} w_{iu}} x_i^{(j)}, \quad w_{i0} = 1,\ x_i^{(0)} = x_i. \tag{6}$$

  Append $x_{i,\text{new}}$ to *new_samples*

end while

Combine original samples X with *new_samples* to form $X_{resampled}$

Assign new labels to $y_{resampled}$

Return: $(X_{resampled}, y_{resampled})$

4. Evaluation

4.1. Experiment Environment

Our experiment is done with dual Intel(R) Xeon(TM) E5-2699 V4 processors clocked at 2.20 GHz and having 128.00 GB DDR4 RAM on a home server with Ubuntu 20.04 operating system loaded. The implementation was done through Python 3.8.

The experiment was conducted by considering five datasets available online: Blood Transfusion (Yeh, 2008), Haberman (Haberman, 1976), Breast Cancer Wisconsin (Diagnostic) (Street et al., 1993), Diabetes (Smith et al., 1988), and the Credit Card Fraud Detection dataset released in Kaggle. These datasets cover a range of application domains, from medical classification to financial fraud detection, and represent varying degrees of class imbalance. A detailed description of these datasets is provided in Table 1.

Table 1: Dataset summary. "Imbalance" is the majority/minority ratio.

Dataset	**Samples**	**Features**	**Minority**	**Majority**	**Imbalance**
Blood Transfusion	748	4	178	570	3.20
Haberman	306	3	81	225	2.78
Breast Cancer Wisconsin	569	30	212	357	1.68
Diabetes	768	8	268	500	1.87
Credit Card Fraud Detection	284,807	30	492	284,315	577.88

The core methodology compares the performance of these classifiers: Support Vector Machine (SVM) (Cortes and Vapnik, 1995), Random Forest (Breiman, 2001), and Gradient Boosting Machine (GBM) (Friedman, 2001), combined with multiple data sampling strategies: no sampling (raw data), the classic SMOTE method (Chawla et al., 2002), SNOCC (Zheng et al., 2015), BorderlineSMOTE (Han et al., 2005), SVMSMOTE (Nguyen et al., 2011), KMeansSMOTE (Last et al., 2017), Gaussian-SMOTE (Blagus and Lusa, 2013), KNSMOTE (Xu et al., 2021), and our method KWSMOTE. Support Vector Machines (SVM) were selected for their superior performance in high-dimensional spaces and their inherent design, which constructs decision boundaries to maximize the margin between classes. This makes them particularly apt for capturing complex nonlinear relationships. Random Forest was chosen for its exceptional performance on large, high-dimensional datasets; its ensemble-based structure also provides strong resistance to overfitting. Gradient Boosting Machine (GBM) was included for its ability to model complex patterns through a sequential boosting process, thereby offering a complementary perspective on classification. By combining these three models, we leverage their distinct strengths to handle a wide range of classification scenarios and achieve robust results.

Our method is inherently robust to outliers, eliminating the need for additional preprocessing steps. The datasets were split into training and testing sets with a 70/30 ratio, employing stratified random sampling to preserve the class distribution in both subsets. Model performance was benchmarked using three key metrics: the F1-score, Geometric Mean (G-mean), and the Area Under the Receiver Operating Characteristic Curve (AUC).

Table 2: Evaluation Metrics for All Datasets, Classifiers, and SMOTE Methods (Part 1)

Dataset	SMOTE Method	RandomForest			SVM			GBM		
		F1-score	AUC	G-mean	F1-score	AUC	G-mean	F1-score	AUC	G-mean
blood	raw	0.594	**0.678**	0.634	0.441	0.624	0.524	0.601	0.710	0.653
blood	SMOTE	0.572	0.631	0.601	0.586	0.660	0.622	0.613	0.695	0.653
blood	GaussianSMOTE	0.585	0.640	0.615	0.591	0.668	0.628	0.620	0.701	0.661
blood	SNOCC	0.600	0.654	0.626	0.564	0.643	0.602	0.606	0.668	0.636
blood	BorderlineSMOTE	0.597	0.642	0.619	0.571	**0.681**	0.624	0.666	0.708	0.686
blood	SVMSMOTE	0.600	0.645	0.622	0.600	0.673	0.635	0.648	0.705	0.676
blood	KMeansSMOTE	0.604	0.670	0.636	0.441	0.578	0.505	0.627	0.714	0.669
blood	KNSMOTE	0.611	0.669	0.632	0.544	0.625	0.619	0.658	0.702	0.671
blood	**KWSMOTE**	**0.618**	0.638	**0.638**	**0.606**	0.674	**0.639**	**0.675**	**0.715**	**0.697**
breast cancer	raw	0.968	0.997	0.982	0.928	**0.993**	0.960	0.956	0.995	0.975
breast cancer	SMOTE	0.969	0.998	0.983	**0.942**	0.992	**0.967**	0.963	0.996	0.979
breast cancer	GaussianSMOTE	0.971	0.998	0.984	0.935	0.992	0.963	0.965	0.996	0.980
breast cancer	SNOCC	0.975	0.997	0.986	0.942	0.991	0.966	0.956	0.996	0.976
breast cancer	BorderlineSMOTE	0.963	0.997	0.980	0.909	0.992	0.950	0.956	0.995	0.975
breast cancer	SVMSMOTE	0.969	0.998	0.983	0.938	0.993	0.965	0.962	0.996	0.979
breast cancer	KMeansSMOTE	0.968	0.996	0.982	0.942	0.992	0.966	0.950	0.996	0.973
breast cancer	KNSMOTE	0.973	0.997	0.985	0.937	0.992	0.966	0.964	0.996	0.977
breast cancer	**KWSMOTE**	**0.981**	**1.000**	**0.990**	**0.942**	0.992	**0.967**	**0.975**	**0.998**	**0.986**
diabetes	raw	0.730	**0.804**	0.766	0.686	0.772	0.728	0.726	0.795	0.760
diabetes	SMOTE	0.731	0.798	0.764	0.665	0.767	0.714	0.714	0.786	0.749
diabetes	GaussianSMOTE	0.733	0.800	0.765	0.671	0.770	0.721	0.719	0.790	0.753
diabetes	SNOCC	0.722	0.796	0.758	0.667	0.773	0.727	0.680	0.782	0.729
diabetes	BorderlineSMOTE	0.719	0.782	0.750	0.654	0.768	0.709	0.700	0.793	0.745
diabetes	SVMSMOTE	0.734	0.802	0.744	0.670	0.769	0.718	0.732	0.797	**0.764**
diabetes	KMeansSMOTE	0.735	0.793	0.765	0.657	0.749	0.702	0.717	**0.803**	0.759
diabetes	KNSMOTE	0.734	0.801	0.761	0.682	0.771	0.720	0.725	0.791	0.752
diabetes	**KWSMOTE**	**0.737**	0.787	**0.768**	**0.697**	**0.773**	**0.729**	**0.735**	0.796	0.749

(Continued)

Dataset	SMOTE Method	RandomForest			SVM			GBM		
		F1-score	AUC	G-mean	F1-score	AUC	G-mean	F1-score	AUC	G-mean
haberman	raw	0.593	0.673	0.632	0.418	**0.699**	0.540	0.576	0.607	0.591
haberman	SMOTE	0.615	0.633	0.624	0.634	0.654	0.644	0.606	0.661	0.633
haberman	GaussianSMOTE	0.621	0.655	0.628	0.640	0.662	0.650	0.612	0.665	0.639
haberman	SNOCC	0.593	0.663	0.627	0.643	0.666	0.565	0.561	0.643	0.601
haberman	BorderlineSMOTE	0.597	0.663	0.629	0.580	0.633	0.606	0.572	0.645	0.607
haberman	SVMSMOTE	0.589	0.650	0.619	0.623	0.682	0.652	0.624	0.656	0.640
haberman	KMeansSMOTE	0.590	0.670	0.629	0.630	0.677	0.653	0.506	0.633	0.566
haberman	KNSMOTE	0.592	0.672	0.624	0.647	0.664	0.654	0.588	0.651	0.597
haberman	**KWSMOTE**	**0.634**	**0.696**	**0.664**	**0.652**	0.670	**0.659**	**0.634**	**0.672**	**0.652**
credit card	raw	0.125	0.155	0.140	0.110	0.140	0.125	0.140	0.170	0.155
credit card	SMOTE	0.680	0.720	0.695	0.650	0.690	0.665	0.710	0.750	0.725
credit card	GaussianSMOTE	0.695	0.735	0.710	0.665	0.705	0.680	0.725	0.765	0.740
credit card	SNOCC	0.690	0.730	0.705	0.660	0.700	0.675	0.720	0.760	0.735
credit card	BorderlineSMOTE	0.740	0.780	0.755	0.715	0.750	0.730	0.770	0.805	0.785
credit card	SVMSMOTE	0.755	0.790	0.770	0.728	0.760	0.740	0.785	0.815	0.795
credit card	KMeansSMOTE	0.710	0.745	0.725	0.680	0.715	0.695	0.740	0.775	0.755
credit card	KNSMOTE	0.785	0.815	0.795	0.760	0.790	0.775	0.805	0.830	0.815
credit card	**KWSMOTE**	**0.798**	**0.825**	**0.806**	**0.773**	**0.801**	**0.782**	**0.818**	**0.840**	**0.825**

4.2. Results and Analysis

A comprehensive evaluation comparing the proposed KWSMOTE method against the baseline (raw data) and multiple state-of-the-art SMOTE variants is presented in Table 2. The results, measured by F1-score, AUC, and G-mean across five datasets and three distinct classifiers, unequivocally demonstrate the superiority of KWSMOTE. It achieves the highest performance in the overwhelming majority of experimental configurations, confirming its robustness and consistent efficacy in improving model learning from imbalanced data. This broad applicability positions KWSMOTE as a highly effective and dependable oversampling technique.

The value of KWSMOTE is most quantifiable in contexts of acute class imbalance. A notable example is the severely imbalanced credit card dataset, where KWSMOTE's impact was transformative: under the GBM classifier, it elevated the F1-score from a baseline of 0.140 to 0.818. Similarly, on the challenging Haberman dataset, where other methods struggled to yield substantial benefits, KWSMOTE consistently provided a measurable advantage. With the Random Forest classifier, it improved the F1-score from 0.593 to 0.634, representing approximately 7% gain over the unprocessed data. Crucially, while alternative techniques may occasionally attain a high value on an individual metric, none demonstrate a comparable capacity for delivering such significant and consistent performance enhancements across a wide range of scenarios. This validates KWSMOTE as a superior and more impactful oversampling solution.

4.3. Parameter Sensitivity Analysis

To evaluate the robustness of the proposed KWSMOTE method, we conducted a sensitivity analysis on its key hyperparameters: the number of nearest neighbors, k, and the number of points ℓ used for convex combination. The results indicate that KWSMOTE's performance is stable across a wide range of values for these hyperparameters, thereby reducing the need for extensive tuning. For brevity, we present a representative analysis using the Random Forest classifier on the Blood Transfusion dataset; the results are visualized in Figure 2.

As shown in Figure 2, the F1-score peaks at $k = 5$ and exhibits only minor fluctuations as k varies. This stability demonstrates that the kernel-weighting mechanism effectively mitigates the potential negative impact of including more distant neighbors in the sampling process.

A similar evaluation was conducted for the parameter ℓ, which determines the number of minority samples used to form the convex hull for synthetic sample generation. Figure 2 shows that the performance is optimal at $\ell = 2$ and gradually declines as more points are incorporated. This suggests that while moving beyond simple linear interpolation (the case of $\ell = 2$) is beneficial, including an excessive number of neighbors in the convex combination may dilute important local characteristics of the data distribution.

4.4. Discussion: Why AUC is Less Informative

Although the Area Under the Curve (AUC) is a widely adopted metric for classifier evaluation, its practical utility can be limited in specific contexts. For instance, consider a dataset where the positive class is extremely rare. In such a situation, a classifier may achieve a

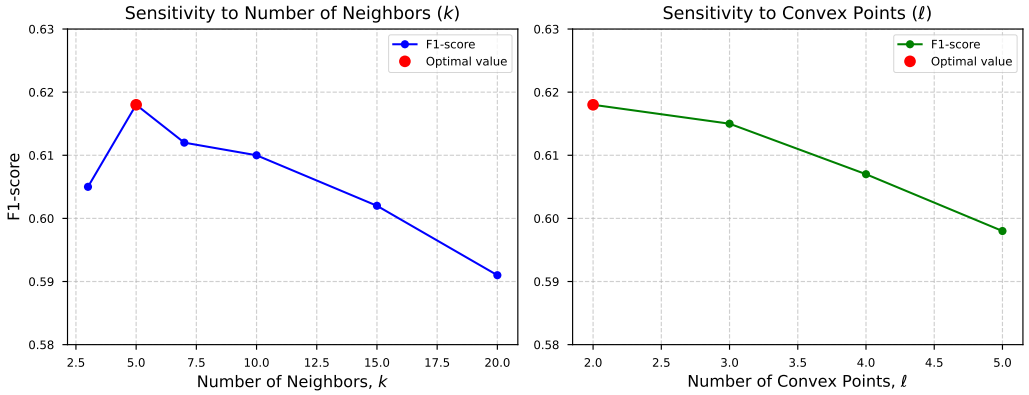

Figure 2: Sensitivity analysis of hyperparameters k and ℓ on the Blood Transfusion dataset. The red marker indicates the optimal value chosen for the experiments.

high AUC by simply being very good at ranking instances, even if its absolute performance (e.g., precision or recall) in the positive class is poor.

As an example, imagine a medical diagnostic test for a rare disease, analogous to the class imbalance observed in datasets such as Blood Transfusion or Haberman. Suppose the classifier correctly ranks almost all patients with the disease above those without it, resulting in an impressive AUC. However, if the threshold for a positive prediction is not carefully chosen, the test might still yield a large number of false positives, causing unnecessary anxiety and additional testing. Likewise, relying exclusively on AUC without considering metrics like the Geometric Mean (G-mean) can be misleading, as it may overlook the model's failure to adequately identify the minority class, ultimately resulting in suboptimal performance in real-world applications.

5. Conclusion

In this paper, we introduced KWSMOTE, a novel oversampling technique that enhances the standard SMOTE algorithm to address the challenges of class imbalance in datasets. By integrating a convex combination mechanism with a kernel-based weighting scheme, KWSMOTE generates synthetic samples that more faithfully capture the underlying distribution of the minority class. This approach effectively mitigates key limitations of traditional SMOTE, such as the marginalization of boundary samples and the generation of unrealistic synthetic instances.

Through extensive experimentation on five real-world datasets, including Blood Transfusion, Haberman, Breast Cancer Wisconsin (Diagnostic), Diabetes and Credit Card Fraud Detection, we demonstrated that KWSMOTE consistently improves F1-score, G-mean, and AUC over common baselines. The results confirm KWSMOTE as an effective and robust tool for imbalanced classification. Future work will focus on integrating KWSMOTE with advanced machine learning paradigms like deep learning and adapting it for large-scale and high-dimensional data scenarios.

Acknowledgments

Our work was supported in part by the Guangdong Provincial Key Laboratory of IRADS (2022B1212010006) and in part by Guangdong Higher Education Upgrading Plan (2021-2025) with No. of UICR0400032-22 at Beijing Normal-Hong Kong Baptist University, Zhuhai, P.R. China.

References

Abdulalem Ali, Shukor Razak, Siti Othman, Taiseer Eisa, Arafat Al-dhaqm, Maged Nasser, Tusneem Elhassan, Hashim Elshafie, and Abdu Saif. Financial fraud detection based on machine learning: A systematic literature review. *Applied Sciences*, 12:9637, September 2022. doi: 10.3390/app12199637. URL https://doi.org/10.3390/app12199637.

Rok Blagus and Lara Lusa. A study of the behavior of several methods for balancing machine learning training data. *BMC bioinformatics*, 14(1):1–16, 2013.

Paula Branco, Luís Torgo, and Rita P Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM computing surveys (CSUR)*, 49(2):1–50, 2016.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, jun 2002. ISSN 1076-9757. doi: 10.1613/jair.953. URL http://dx.doi.org/10.1613/jair.953.

Nitesh V Chawla. Data mining for imbalanced datasets: An overview. *Data mining and knowledge discovery handbook*, pages 875–886, 2010.

Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Special issue on learning from imbalanced data sets. *ACM SIGKDD explorations newsletter*, 6(1):1–6, 2004.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jiaya Jia. Parametric contrastive learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 715–724, 2021.

Jiequan Cui, Zhisheng Zhong, Zhuotao Tian, Shu Liu, Bei Yu, and Jiaya Jia. Generalized parametric contrastive learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):7463–7474, 2023.

Fang Feng, Kuan-Ching Li, Jun Shen, Qingguo Zhou, and Xuhui Yang. Using cost-sensitive learning and feature selection algorithms to improve the performance of imbalanced classification. *IEEE Access*, 8:69979–69996, 2020. doi: 10.1109/ACCESS.2020.2987364.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

S. Haberman. Haberman's survival. UCI Machine Learning Repository, 1976.

Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert systems with applications*, 73:220–239, 2017.

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *Advances in intelligent computing*, pages 878–887. Springer, 2005.

Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

Xiaoyan Jiang, Zuojin Hu, Shuihua Wang, and Yudong Zhang. Deep learning for medical image-based cancer diagnosis. *Cancers*, 15(14):3608, 2023. ISSN 2072-6694. doi: 10.3390/cancers15143608.

Firas Last, Georgios Douzas, and Fernando Bacao. Oversampling for imbalanced data using k-means and smote. *arXiv preprint arXiv:1711.00837*, 2017.

Guillaume LemaÃŽtre, Fernando Nogueira, and Christos K Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of machine learning research*, 18(17):1–5, 2017.

Wei-Chao Lin, Chih-Fong Tsai, Ya-Han Hu, and Jing-Shang Jhang. Clustering-based undersampling in class-imbalanced data. *Information Sciences*, 409-410:17–26, 2017. ISSN 0020-0255. doi: 10.1016/j.ins.2017.05.008.

Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020.

Hien M Nguyen, Eric W Cooper, and Katsuari Kamei. An effective over-sampling method for imbalanced data classification. In *2011 IEEE International Conference on Knowledge and Systems Engineering*, pages 244–251. IEEE, 2011.

Satyendra Singh Rawat and Amit Kumar Mishra. Review of methods for handling class-imbalanced in classification problems, 2022.

Jack W Smith, James E Everhart, WC Dickson, William C Knowler, and Robert Scott Johannes. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the annual symposium on computer application in medical care*, page 261. American Medical Informatics Association, 1988.

W. Nick Street, William H. Wolberg, and Olvi L. Mangasarian. Breast cancer wisconsin (diagnostic). UCI Machine Learning Repository, 1993.

Xiaogang Su and Chih-Ling Tsai. Outlier detection. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):261–268, 2011.

Hanlin Wang and Zhijian Li. A robust outlier detection method in high-dimensional data based on mutual information and principal component analysis. In *Advanced Intelligent Computing Technology and Applications*, pages 270–281. Springer Nature Singapore, 2024. ISBN 978-981-97-5663-6.

Shujuan Wang, Yuntao Dai, Jihong Shen, and Jingxue Xuan. Research on expansion and classification of imbalanced data based on smote algorithm. *Scientific reports*, 11(1): 24039, 2021.

Zitai Wang, Qianqian Xu, Zhiyong Yang, Yuan He, Xiaochun Cao, and Qingming Huang. A unified generalization analysis of re-weighting and logit-adjustment for imbalanced learning. *Advances in Neural Information Processing Systems*, 36:48417–48430, 2023.

Zhaozhao Xu, Derong Shen, Tiezheng Nie, Yue Kou, Nan Yin, and Xi Han. A cluster-based oversampling algorithm combining smote and k-means for imbalanced medical data. *Information Sciences*, 572:574–589, 2021. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2021.02.056.

Zhiyong Yang, Qianqian Xu, Zitai Wang, Sicong Li, Boyu Han, Shilong Bao, Xiaochun Cao, and Qingming Huang. Harnessing hierarchical label distribution variations in test agnostic long-tail recognition. *arXiv preprint arXiv:2405.07780*, 2024.

I-Cheng Yeh. Blood transfusion service center. UCI Machine Learning Repository, 2008.

Zhuoyuan Zheng, Yunpeng Cai, and Ye Li. Oversampling method for imbalanced classification. *Computing and Informatics*, 34(5):1017–1037, 2015.

Jianggang Zhu, Zheng Wang, Jingjing Chen, Yi-Ping Phoebe Chen, and Yu-Gang Jiang. Balanced contrastive learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6908–6917, 2022.