Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

What actually matters in synthetic oversampling? A factorial decomposition of SMOTE-family methods and an evidence-based configuration

Mehmet Ali Cengiz

mamcengiz@imamu.edu.sa

Imam Mohammad Ibn Saud Islamic University (IMSIU)

Emre Dünder

Ondokuz Mayıs University

Research Article

Keywords: Class imbalance, Oversampling, SMOTE, Experimental methodology, Bayesian hierarchical model, Data complexity

Posted Date: November 2nd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11405914/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

What actually matters in synthetic oversampling? A factorial decomposition of SMOTE-family methods and an evidence-based configuration

Mehmet Ali Cengiz[1*] · Emre Dünder[2]

[1] Department of Mathematics and Statistics, College of Science, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia

[2] Department of Statistics, Faculty of Science, Ondokuz Mayıs University, Samsun, Turkey

[*] Corresponding author: [mamcengiz@imamu.edu.sa]

Abstract

Around a hundred variants of the synthetic minority oversampling technique (SMOTE) have been proposed, each changing some part of the original algorithm, yet it is rarely known which part is responsible for a reported gain. We decompose SMOTE-family oversamplers into five interchangeable components (seed allocation, partner selection, generator, filter and budget) and study them in a full factorial design complemented by level sweeps, reference strategies and eleven published methods. Fifty-four imbalanced data sets, six learners, repeated 5×2 cross-validation and an inner-validated decision threshold are combined with Bayesian hierarchical models that separate component effects from data-set and learner heterogeneity and assess practical equivalence with a region of practical equivalence (ROPE). Partner selection was practically inert in both learner groups and for every neighbourhood size. Allocation, filtering and budget mainly shifted the operating point between recall and precision, a shift that a tuned threshold removed. Edited-nearest-neighbour cleaning was consistently harmful, and difficulty-based (ADASYN-type) allocation slightly so. The largest gains came from volume-filling generators combined with tree ensembles, about two points of PR-AUC and tuned MCC, larger on average in data sets whose minority examples overlap the majority class. Borderline-SMOTE, ADASYN, SMOTE-ENN and G-SMOTE behaved like single components. A configuration derived from these results, safety-allocated local Gaussian oversampling, retained a smaller advantage over SMOTE on 57 hold-out data sets, credible in PR-AUC for both learner groups. Component terms explained less than one percent of the variance of PR-AUC and tuned MCC, far less than the data set.

Keywords Class imbalance · Oversampling · SMOTE · Experimental methodology · Bayesian hierarchical model · Data complexity

1 Introduction

Learning from imbalanced data remains one of the most persistent practical problems in supervised classification, because standard learners optimise criteria in which the rare class carries little weight and because the rare class is usually the one of interest (He and Garcia 2009; Sun et al. 2009; Branco et al. 2016; Krawczyk 2016). Among the many remedies, data-level oversampling is the most widely used, and the synthetic minority oversampling technique (SMOTE) of Chawla et al. (2002) has become its reference point. Rather than duplicating minority examples, SMOTE draws a minority seed, chooses one of its minority nearest neighbours and places a synthetic point uniformly on the segment between the two. Its

simplicity and generality explain why it is still the default baseline two decades later (Fernández et al. 2018).

The success of SMOTE has generated a remarkably large family of variants. Fernández et al. (2018) counted close to one hundred extensions at the method's fifteenth anniversary, and the smote-variants library implements 85 of them (Kovács 2019b). Most variants keep the overall structure of SMOTE and modify one or two of its steps: which minority examples receive synthetic neighbours (Han et al. 2005; He et al. 2008; Barua et al. 2014), with which neighbour a seed is paired (Bunkhumpornpat et al. 2009; Zhang and Li 2014), how the new point is placed (Douzas and Bacao 2019; Bej et al. 2021; Kachan et al. 2025), whether candidate points are cleaned afterwards (Batista et al. 2004; Sáez et al. 2015), or how many points are created (Douzas et al. 2018). Each new method is typically presented with a benchmark in which it outperforms a selection of earlier ones, often with an ablation of its own parts. What such studies cannot show is which of the many modifications proposed across the literature actually carry the improvements, because every published method changes a different combination of steps and is evaluated under a different protocol.

Large empirical comparisons have sharpened this question without answering it. Kovács (2019a) compared 85 oversamplers on more than one hundred data sets and found that a few relatively old methods ranked best, with modest differences overall. Elor and Averbuch-Elor (2022) showed that, with strong learners such as gradient-boosted trees and proper evaluation, balancing often brings no improvement at all, and Sakho et al. (2026) reached a similar conclusion through a theoretical analysis of the SMOTE generator. Recent benchmarks such as CLIMB (Liu et al. 2025) and critical studies (Hassanat et al. 2022; Andersen et al. 2026) point in the same direction. All of these studies compare whole methods. When two methods differ, it remains unknown whether the difference comes from allocation, partner selection, the generator or a filter, and when a method shows no gain, it is unknown whether all of its parts are inert or whether helpful and harmful parts cancel out.

In this paper we take the opposite route and treat an oversampler as a combination of interchangeable parts. We identify five components that together describe most SMOTE-family methods: seed allocation (A), partner selection (B), the generator (C), a post-generation filter (D) and the oversampling budget (E). We implement each component at several levels in a single modular oversampler that reproduces SMOTE, Borderline-SMOTE, ADASYN, SMOTE-ENN and related methods as special cases, and we study the components in a full 2^5 factorial design, a one-factor level sweep and a direct comparison with eleven published implementations. The experiments use 54 imbalanced binary data sets, six learners from linear models to gradient boosting, repeated 5×2 cross-validation and a decision threshold tuned by inner cross-validation, so that a change in the learnt ranking can be separated from a mere shift of the operating point (Zou et al. 2016; Elor and Averbuch-Elor 2022). The results are analysed with a Bayesian hierarchical model that accounts for the dependence between data sets derived from the same feature matrix and that

allows statements of practical equivalence through a region of practical equivalence (ROPE) rather than only rejections of a null hypothesis (Benavoli et al. 2017; Kruschke 2018).

The study makes five contributions. First, it provides, to our knowledge, the first factorial component-level decomposition of SMOTE-family oversampling across method families, together with an open modular implementation. Second, it quantifies which components matter: partner selection is practically inert, allocation, filtering and budget mainly act as implicit threshold moving, edited-nearest-neighbour cleaning is harmful, and the largest gains come from volume-filling generators combined with tree ensembles. Third, it shows that the published implementations of Borderline-SMOTE, ADASYN, SMOTE-ENN and G-SMOTE behave, data set by data set, like the corresponding single component, so that their remaining design details add nothing measurable, whereas methods that combine several modifications are not reducible to one component but do not exceed the best one either. Fourth, it relates the generator effect to data difficulty, measured by the k-disagreeing-neighbour score of minority examples (Smith et al. 2014; Napierała and Stefanowski 2016) and the imbalance ratio. Fifth, it turns these findings into a simple evidence-based configuration, which combines a ROSE-type local Gaussian generator (Menardi and Torelli 2014) with safety-based allocation, and a meta-feature-based selector, both evaluated on 57 data sets from feature-matrix families that do not occur in the benchmark.

The remainder of the paper is organised as follows. Section 2 reviews related work. Section 3 formalises the component view and maps published methods onto it. Section 4 describes the experimental design and the Bayesian analysis. Section 5 presents the results, Section 6 discusses their implications and Section 7 their limitations, and Section 8 concludes.

2 Related work

2.1 SMOTE and its building blocks

The original SMOTE algorithm generates, for every minority example, synthetic points on segments to randomly chosen members of its k minority nearest neighbours (Chawla et al. 2002). Its descendants can be organised by the step they modify. Allocation methods decide which seeds receive synthetic offspring. Borderline-SMOTE oversamples only minority examples whose neighbourhood is dominated, but not exhausted, by the majority class (Han et al. 2005), ADASYN allocates more points to minority examples with more majority neighbours (He et al. 2008), and MWMOTE weights seeds by their proximity to the majority class (Barua et al. 2014). Partner-selection methods change the neighbour with which a seed is paired. Safe-Level-SMOTE weights the interpolation by the number of minority neighbours of seed and partner (Bunkhumpornpat et al. 2009), LN-SMOTE extends this idea to local neighbourhoods (Maciejewski and Stefanowski 2011), and random-walk and Markov-chain approaches replace the nearest-neighbour graph by a transition structure (Zhang and Li 2014; Sağlam and Cengiz 2025). Generator methods alter the geometry of the synthetic point. ROSE draws synthetic examples from a smoothed bootstrap, that is, from a kernel density estimate centred at randomly chosen examples (Menardi and Torelli 2014), and radial-

based oversampling places points according to potential functions of both classes (Koziarski et al. 2019). G-SMOTE samples inside a truncated and deformed hypersphere around the seed (Douzas and Bacao 2019), LoRAS draws convex combinations of locally perturbed points (Bej et al. 2021), Simplicial SMOTE samples from simplices of the neighbourhood complex (Kachan et al. 2025), manifold-based approaches generate along an estimated data manifold (Bellinger et al. 2016; Bellinger et al. 2018), and probabilistic or generative models replace interpolation altogether (Das et al. 2015; Douzas and Bacao 2018). Filtering methods remove synthetic or original points after generation, most prominently edited nearest neighbours (Wilson 1972) and Tomek links (Tomek 1976) in SMOTE-ENN and SMOTE-Tomek (Batista et al. 2004), or iterative partitioning filters in SMOTE-IPF (Sáez et al. 2015). Finally, budget and hybrid approaches combine oversampling with undersampling or ensembles (Chawla et al. 2003; Liu et al. 2009; Seiffert et al. 2010; Ramentol et al. 2012; Douzas et al. 2018). Many recent methods combine several of these modifications, for instance clustering-based allocation with interpolation (Ma and Fan 2017; Douzas et al. 2018), or safety-aware allocation with gravity-weighted partners (Dünder et al. 2027), which makes their individual contributions impossible to read off a whole-method benchmark.

2.2 Empirical comparisons and critiques

Early comparative studies established that the benefit of balancing depends strongly on the learner and on the evaluation criterion (Japkowicz and Stephen 2002; Weiss and Provost 2003; Batista et al. 2004; Van Hulse et al. 2007). Large-scale evaluations followed. López et al. (2013) related the performance of preprocessing methods to data-intrinsic characteristics, Luengo et al. (2011) to data complexity measures, and Prati et al. (2015) proposed an experimental setup that controls the degree of imbalance. Kovács (2019a) remains the most extensive comparison of oversamplers to date. More recent work has questioned the usefulness of oversampling itself. Blagus and Lusa (2013) showed that SMOTE can be ineffective in high dimensions, van den Goorbergh et al. (2022) that imbalance corrections damage calibration of clinical risk models without improving discrimination, and Elor and Averbuch-Elor (2022) that strong learners with tuned thresholds rarely benefit from balancing. Similar conclusions appear in recent reviews and benchmarks (Hassanat et al. 2022; Liu et al. 2025; Andersen et al. 2026; Musaev 2026; Sakho et al. 2026). Closer to our perspective, Carvalho et al. (2025) reviewed resampling approaches from a data perspective, and the smote-variants library annotates each method with descriptive attributes (Kovács 2019b). These taxonomies describe methods by their parts, but the parts are never varied experimentally. The only experimental studies of individual parts that we are aware of concern allocation alone: Skryjomski and Krawczyk (2017) restricted SMOTE seeds to particular types of minority examples, and Sáez et al. (2016) examined which classes and example types to oversample in multi-class problems. Ablation studies in method papers, finally, vary only the parts of the method being proposed. A factorial decomposition across method families, as carried out here, is therefore still missing.

2.3 Data difficulty factors

The difficulty of an imbalanced problem depends less on the imbalance ratio itself than on class overlap, small disjuncts, rare examples and noise (Jo and Japkowicz 2004; Prati et al. 2004; García et al. 2008; López et al. 2013). Napierała and Stefanowski (2016) proposed classifying minority examples as safe, borderline, rare or outliers according to the number of majority examples among their k = 5 nearest neighbours, a quantity closely related to the k-disagreeing-neighbours (kDN) measure of instance hardness (Smith et al. 2014). Complexity measures such as the N3 error of the one-nearest-neighbour classifier summarise the same information at the data-set level (Ho and Basu 2002; Lorena et al. 2019). Class overlap in particular has been identified as the dominant source of difficulty in imbalanced classification (Vuttipittayamongkol et al. 2021). Several studies relate preprocessing performance to these factors (Napierała et al. 2010; Wojciechowski and Wilk 2017), and we use them to explain where the generator matters.

2.4 Evaluation methodology

The choice of metric strongly affects conclusions in imbalanced learning. Precision–recall based measures are more informative than ROC analysis under strong imbalance (Davis and Goadrich 2006; Saito and Rehmsmeier 2015), and the Matthews correlation coefficient (MCC) uses all four cells of the confusion matrix and remains informative when the classes are unbalanced (Boughorbel et al. 2017; Chicco and Jurman 2020). Because oversampling shifts a classifier's scores, part of its apparent effect at the default threshold of 0.5 can be reproduced by moving the threshold (Provost 2000; Elkan 2001; Zou et al. 2016; Esposito et al. 2021), and since the choice of threshold changes threshold-dependent metrics substantially (Koyejo et al. 2014; Lipton et al. 2014; Collell et al. 2018), such metrics are informative only when the threshold is reported and, ideally, tuned. Probability calibration is affected as well (Zadrozny and Elkan 2002; Niculescu-Mizil and Caruana 2005; Dal Pozzolo et al. 2015; van den Goorbergh et al. 2022). For comparisons across many data sets, the rank-based tests of Demšar (2006) and García and Herrera (2008) have been complemented by Bayesian analyses that estimate effect sizes directly and can establish practical equivalence through a ROPE (Benavoli et al. 2014; Corani and Benavoli 2015; Benavoli et al. 2017; Kruschke 2018; Kruschke and Liddell 2018). We use the repeated two-fold cross-validation design of Dietterich (1998) and hierarchical modelling of repeated measurements (Gelman 2006; Gelman et al. 2013). Instead of correcting the variance of fold-level differences (Nadeau and Bengio 2003), we average scores over folds and model the dependence between learners and between data sets that share a feature matrix.

3 A component view of synthetic oversampling

3.1 Notation

Let $D = \{(x_i, y_i)\}$, $i = 1, \ldots, n$, be a training sample with $x_i \in R^p$ and $y_i \in \{0, 1\}$, where the minority class is coded as 1. Let $P = \{x_i : y_i = 1\}$ and $N = \{x_i : y_i = 0\}$ denote the minority and majority examples, with $n_+ = |P| < n_- = |N|$ and imbalance ratio $IR = n_-/n_+$. An oversampler adds a set S of synthetic minority points and

returns $D \cup \{(s, 1) : s \in S\}$. For $x \in P$ we write $NN_k^P(x)$ for its k nearest minority neighbours and $NN_m(x)$ for its m nearest neighbours in the whole training sample, and $r_m(x) = |NN_m(x) \cap N|/m$ for the share of majority examples among the latter. The quantity $r_5(x)$ is the kDN score of x, and $1 - r_m(x)$ is its local safety.

3.2 Five components

Every oversampler considered in this paper can be written as the composition of five components. The budget (E) fixes the number of synthetic points $|S| = B$. The allocation (A) is a probability vector w over P from which B seeds are drawn. For each seed, the partner rule (B) chooses a second minority example, and the generator (C) produces a synthetic point from the seed and its partner. The filter (D) finally accepts, rejects or removes points. Formally,

$$S = F_D(\ \{\ G_C(x_s, x_t) : x_s \sim w_A,\ \ x_t \sim \pi_B(\cdot \mid x_s)\ \}\ ),\ \text{with } |S| = B_E \text{ before filtering} \qquad (1)$$

where w_A is the allocation distribution, $\pi_B(\cdot \mid x_s)$ the partner distribution, G_C the generator and F_D the filter. We implemented the following levels.

A. Seed allocation. A0 is uniform, $w(x) = 1/n_+$, as in SMOTE. A1 restricts seeds to the danger set of Borderline-SMOTE, $\{x : 1/2 \le r_m(x) < 1\}$ with m = 10 (Han et al. 2005). A2 and A3 make w proportional to local safety $1 - r_m(x)$, either after min–max scaling (A2), as in safety-guided methods (Dünder et al. 2027), or on the raw probability scale (A3). A4 makes w proportional to $r_5(x)$, the difficulty-based allocation of ADASYN (He et al. 2008). In all cases the B seeds are drawn from w with replacement, whereas ADASYN rounds the expected number of points per seed.

B. Partner selection. B0 draws the partner uniformly from $NN_k^P(x_s)$ with k = 5. B1 weights these neighbours by a Gaussian kernel of their distance, $\exp(-d^2/h)$, with h the median squared distance of the seed's k neighbours. B2 weights them by their own safety, in the spirit of Safe-Level-SMOTE (Bunkhumpornpat et al. 2009). B3 draws the partner from the two-step transition matrix of a random walk on the kernel-weighted minority kNN graph, so that partners are neighbours of neighbours and the seed itself is excluded, a simplified version of Markov-chain partner selection (Zhang and Li 2014; Sağlam and Cengiz 2025).

C. Generator. C0 is SMOTE interpolation, $s = x_s + u(x_t - x_s)$ with $u \sim U(0, 1)$. C1 draws a point uniformly from the half-ball centred at x_s with radius $\|x_t - x_s\|$ that faces the partner, which is G-SMOTE with truncation factor 1 and deformation factor 0 (Douzas and Bacao 2019). C2 draws $s \sim N(x_s, \mathrm{diag}(\sigma^2))$, where σ_j^2 is half the variance of feature j among x_s and its k minority neighbours. This local Gaussian generator is close to the smoothed bootstrap of ROSE (Menardi and Torelli 2014), but with a local, data-adaptive diagonal bandwidth instead of a global one, and it does not use the partner. The factor one half was fixed before the experiments and not tuned. C3 draws Dirichlet(1, 1, 1) weights over the seed and two partners and returns their convex combination, a two-dimensional simplex generator (Kachan et al. 2025). In addition, a duplication level (ROS) returns the seed itself.

D. Filter. D0 applies no filter. D1 generates three candidates per required point and keeps the B candidates with the lowest majority share r_5 among their nearest training neighbours, so that the budget is preserved. D2 applies edited nearest neighbours with three neighbours to both classes of the resampled sample (Wilson 1972), as in SMOTE-ENN (Batista et al. 2004). When the weights of an allocation rule sum to zero, for instance when the danger set of A1 is empty, uniform allocation is used instead.

E. Budget. E0 creates $B = n_- - n_+$ points (full balance), E1 creates half of that number.

SMOTE corresponds to the configuration A0 B0 C0 D0 E0. Algorithm 1 gives the modular oversampler in pseudocode.

Algorithm 1 Modular SMOTE-family oversampler

Input: training sample $D = P \cup N$; component levels (a, b, c, d, e); neighbourhood sizes $k = 5$, $m = 10$; $\varepsilon = 10^{-3}$
Output: resampled training sample D'

$B \leftarrow n_- - n_+$ if e = E0, else $\text{round}((n_- - n_+)/2)$ ▷ *budget*
for each $x \in P$ compute $NN_k^P(x)$, $NN_m(x)$ and $r_m(x)$, $r_5(x)$
$w \leftarrow$ allocation weights of level a: ▷ *allocation*
    A0: $w(x) \propto 1$; A1: $w(x) \propto 1[1/2 \le r_m(x) < 1]$; A2: $w(x) \propto \text{minmax}(1 - r_m(x)) + \varepsilon$;
    A3: $w(x) \propto 1 - r_m(x) + \varepsilon$; A4: $w(x) \propto r_5(x)$; if $\Sigma w = 0$ then $w \propto 1$
$B^* \leftarrow 3B$ if d = D1, else B ▷ *candidate pool for the safety filter*
$S \leftarrow \emptyset$
repeat B^* times
    draw seed $x_s \sim w$
draw partner $x_t \sim \pi_b(\cdot \mid x_s)$: ▷ *partner (not used by C2 and Cdup)*
    B0 uniform on $NN_k^P(x_s)$; B1 $\propto \exp(-\|x_t - x_s\|^2/h_s)$ on $NN_k^P(x_s)$; B2 $\propto 1 - r_m(x_t) + \varepsilon$ on $NN_k^P(x_s)$; B3 two-step walk T^2 on the B1-weighted kNN graph, diagonal removed
$s \leftarrow G_c(x_s, x_t)$: ▷ *generator*
    C0 $x_s + u(x_t - x_s)$, $u \sim U(0, 1)$; C1 uniform point in the half-ball $B(x_s, \|x_t - x_s\|)$ facing x_t;
    C2 $N(x_s, \text{diag}(\sigma^2))$ with $\sigma_j^2 = \tfrac{1}{2}\,\text{Var}_j(x_s \cup NN_k^P(x_s))$; C3 Dirichlet(1,1,1) mixture of x_s, x_t, x_t'; Cdup x_s
$S \leftarrow S \cup \{s\}$
if d = D1 then keep the B points of S with the lowest majority share among their 5 nearest training neighbours ▷ *filter*
$D' \leftarrow D \cup \{(s, 1) : s \in S\}$
if d = D2 then $D' \leftarrow \text{ENN}(D')$ with 3 neighbours, applied to both classes
return D'

3.3 Published methods in component space

Table 1 maps the published methods evaluated in this study onto the component space. Borderline-SMOTE (variant 1) differs from SMOTE only in its allocation and is represented by A1, ADASYN by the difficulty allocation A4, SMOTE-ENN by the ENN filter D2 and G-SMOTE with its default settings by the half-ball generator C1, although its default selection strategy also draws some partners from the majority class. Safe-Level-SMOTE is closest to safety-weighted partner selection B2, although its rule also changes the position of the point on the segment, and MCSMOTE to Markov partner selection B3, although its anchor is the nearest neighbour of a uniformly drawn seed (Sağlam and Cengiz 2025). SMOTE-Tomek removes Tomek links after interpolation (Tomek 1976; Batista et al. 2004), KMeans-SMOTE first clusters the data and allocates points to sparse minority clusters (Douzas et al. 2018), SVM-SMOTE generates around support

vectors (Nguyen et al. 2011), and the gravity-based G-SMOTE-Safe combines min–max safety allocation, gravitationally weighted partners and interpolation with u ~ U(0.1, 0.9) (Dünder et al. 2027). The last four have no single-component counterpart. Their comparison with single components therefore tests whether combining several modifications produces more than the sum of its parts.

Table 1 Published methods evaluated in this study and their position in the component space (A allocation, B partner, C generator, D filter, E budget).

Method	Reference	A	B	C	D	E	Implementation	Counterpart
SMOTE	Chawla et al. (2002)	A0	B0	C0	D0	E0	imbalanced-learn	itself
Borderline-SMOTE 1	Han et al. (2005)	A1	B0	C0	D0	E0	imbalanced-learn	A1
ADASYN	He et al. (2008)	A4	B0	C0	D0	E0	imbalanced-learn	A4
SMOTE-ENN	Batista et al. (2004)	A0	B0	C0	D2	E0	imbalanced-learn	D2
SMOTE-Tomek	Batista et al. (2004)	A0	B0	C0	Tomek	E0	imbalanced-learn	–
KMeans-SMOTE	Douzas et al. (2018)	cluster	B0	C0	D0	E0	imbalanced-learn	–
SVM-SMOTE	Nguyen et al. (2011)	SV	B0	C0*	D0	E0	imbalanced-learn	–
Safe-Level-SMOTE	Bunkhumpornpat et al. (2009)	A0	B2*	C0*	D0	E0	own	B2 (approx.)
G-SMOTE	Douzas and Bacao (2019)	A0	B0	C1	D0	E0	own	C1
G-SMOTE-Safe	Dünder et al. (2027)	A2	gravity	C0*	D0	E0	re-implemented	–
MCSMOTE	Sağlam and Cengiz (2025)	in-degree	B3*	C0	D0	E0	reference port	B3 (approx.)

* modified version of the level: SVM-SMOTE also extrapolates away from support vectors, Safe-Level-SMOTE changes the interpolation range according to the safe levels, G-SMOTE-Safe interpolates with u ~ U(0.1, 0.9), and MCSMOTE draws the partner from a transition matrix anchored at the nearest neighbour. cluster: allocation to sparse minority clusters; SV: allocation to minority support vectors; in-degree: anchor selection by 1-NN in-degree; gravity: gravity-weighted partners.

4 Experimental design

4.1 Data

We assembled a pool of 122 binary imbalanced data sets from the KEEL repository (Alcalá-Fdez et al. 2009; Alcalá-Fdez et al. 2011), the UCI repository (Kelly et al. n.d.) and data sets distributed with R packages. Several data sets share the same feature matrix and differ only in the definition of the positive class, for example the one-versus-rest problems derived from the same glass, ecoli, yeast or vowel samples. We call such groups feature-matrix families. The pool contains 97 families. Data sets with a minority class of fewer than ten examples or an imbalance ratio below 1.5 were not included, and exact duplicates were removed. Data sets with more than 6000 observations were reduced to 6000 by stratified subsampling.

The main benchmark consists of 54 data sets (53 families) and was composed of two groups. The first group contains all 34 data sets whose minority examples have a mean kDN score of at least 0.5, that is, whose minority examples are on average not safe in the sense of Napierała and Stefanowski (2016), and whose imbalance ratio is at least nine, the threshold used for highly imbalanced problems in the KEEL literature (Alcalá-Fdez et al. 2011; López et al. 2013). The second group contains 20 further data sets. From each remaining family, the alphabetically first data set was taken, these candidates were sorted by imbalance ratio, and 20 of them were chosen at equally spaced positions of that order. Both selection rules use only data descriptors that are computed before any classifier is fitted, and no data set was selected or excluded on the basis of classification results. The composition of the benchmark was nevertheless fixed after an initial run of all configurations on the whole pool, so the results below refer to a benchmark that deliberately emphasises difficult problems; the same analysis on all 122 data sets is reported in Supplementary Table

S8 and leads to the same conclusions (Section 7). A further 57 data sets from 44 families that share no feature matrix with the benchmark serve as an external hold-out, and eleven data sets that share a family with the benchmark were discarded. The configuration and the selector of Section 5.7 were constructed from the benchmark results alone and then evaluated on the hold-out. Because the initial pool run had included the hold-out data sets, the hold-out is not untouched in the strictest sense. Table 2 describes the benchmark, and Fig. 1 summarises the design.

Table 2 Benchmark data sets: number of examples n, number of features p, number of minority examples n+, imbalance ratio IR, mean kDN score of minority examples (share of majority examples among the five nearest neighbours), share of safe minority examples (Napierała and Stefanowski 2016), the N3 complexity measure, and group (H: hard, kDN ≥ 0.5 and IR ≥ 9; O: other).

Data set	Source	n	p	n+	IR	kDN	Safe	N3	Group
abalone19	KEEL	4174	9	32	129.4	0.97	0.00	0.015	H
abalone2	KEEL	4177	7	136	29.7	0.85	0.00	0.053	H
abalone9-18	KEEL	731	9	42	16.4	0.83	0.05	0.067	H
abalone-17-vs-7-8-9-10	KEEL	2338	9	58	39.3	0.83	0.02	0.030	H
abalone-19-vs-10-11-12-13	KEEL	1622	9	32	49.7	0.96	0.00	0.038	H
abalone-20-vs-8-9-10	KEEL	1916	9	26	72.7	0.89	0.00	0.020	H
abalone-21-vs-8	KEEL	581	9	14	40.5	0.63	0.14	0.010	H
car-good	KEEL	1728	15	69	24.0	0.80	0.00	0.052	H
car-vgood	KEEL	1728	15	65	25.6	0.85	0.00	0.069	H
cleveland-0-vs-4	KEEL	173	13	13	12.3	0.55	0.00	0.058	H
flare-F	KEEL	1066	17	43	23.8	0.74	0.00	0.054	H
glass2	KEEL	214	9	17	11.6	0.81	0.00	0.112	H
glass4	KEEL	214	9	13	15.5	0.51	0.31	0.033	H
glass-0-1-4-6-vs-2	KEEL	205	9	17	11.1	0.85	0.00	0.112	H
glass-0-1-5-vs-2	KEEL	172	9	17	9.1	0.82	0.00	0.122	H
glass-0-1-6-vs-2	KEEL	192	9	17	10.3	0.82	0.00	0.120	H
led7digit-0-2-4-5-6-7-8-9-vs-1	KEEL	443	7	37	11.0	0.82	0.11	0.065	H
poker-8-9-vs-5	KEEL	2075	10	25	82.0	0.97	0.00	0.018	H
poker-8-9-vs-6	KEEL	1485	10	25	58.4	0.63	0.04	0.003	H
poker-8-vs-6	KEEL	1477	10	17	85.9	0.71	0.06	0.003	H
thyroid-disease	UCI (1986)	3771	25	231	15.3	0.79	0.00	0.078	H
winequality-red-3-9-vs-5	KEEL	1482	11	25	58.3	0.94	0.00	0.019	H
winequality-red-3-vs-5	KEEL	691	11	10	68.1	0.92	0.00	0.022	H
winequality-red-3-vs-7	KEEL	900	11	20	44.0	0.92	0.00	0.022	H
winequality-red-4	KEEL	1599	11	53	29.2	0.92	0.00	0.053	H
winequality-red-8-vs-6	KEEL	656	11	18	35.4	0.91	0.00	0.035	H
winequality-red-8-vs-6-7	KEEL	855	11	18	46.5	0.94	0.00	0.032	H
yeast4	KEEL	1484	8	51	28.1	0.75	0.06	0.040	H
yeast6	KEEL	1484	8	35	41.4	0.54	0.37	0.030	H
yeast-0-2-5-6-vs-3-7-8-9	KEEL	1004	8	99	9.1	0.51	0.35	0.080	H
yeast-0-3-5-9-vs-7-8	KEEL	506	8	50	9.1	0.68	0.16	0.115	H
yeast-1-2-8-9-vs-7	KEEL	947	8	30	30.6	0.83	0.00	0.043	H
yeast-1-4-5-8-vs-7	KEEL	693	8	30	22.1	0.89	0.00	0.062	H
yeast-1-vs-7	KEEL	459	7	30	14.3	0.73	0.07	0.078	H
Fishing1	Herriges and Kling (1999)	1182	5	134	7.8	0.65	0.01	0.139	O
Ionosphere	UCI (1989b, 1989a)	351	33	126	1.8	0.42	0.44	0.134	O
ecoli-0-1-4-7-vs-5-6	KEEL	332	6	25	12.3	0.26	0.72	0.036	O
ecoli-0-1-vs-2-3-5	KEEL	244	7	24	9.2	0.28	0.62	0.045	O
ecoli-0-3-4-7-vs-5-6	KEEL	257	7	25	9.3	0.28	0.68	0.043	O
ecoli-0-3-4-vs-5	KEEL	200	7	20	9.0	0.20	0.75	0.030	O
ecoli-0-6-7-vs-3-5	KEEL	222	7	22	9.1	0.44	0.41	0.050	O
ecoli-0-6-7-vs-5	KEEL	220	6	20	10.0	0.40	0.45	0.050	O
kddcup-buffer-overflow-vs-back	KEEL	2233	32	30	73.4	0.08	0.87	0.000	O
kddcup-rootkit-imap-vs-back	KEEL	2225	40	22	100.1	0.30	0.50	0.001	O
khan1	Khan et al. (2001)	89	306	29	2.1	0.05	0.90	0.022	O
kr-vs-k-one-vs-fifteen	KEEL	2244	20	78	27.8	0.06	0.94	0.004	O

Data set	Source	n	p	n+	IR	kDN	Safe	N3	Group
kr-vs-k-three-vs-eleven	KEEL	2935	20	81	35.2	0.44	0.47	0.008	O
kyphosis	R rpart (1992)	81	3	17	3.8	0.56	0.12	0.210	O
long-stayers	KEEL	699	9	241	1.9	0.62	0.17	0.316	O
newthyroid2	KEEL	215	5	35	5.1	0.17	0.69	0.014	O
page-blocks-1-3-vs-4	KEEL	472	10	28	15.9	0.18	0.79	0.002	O
spam	UCI (1999)	4601	57	1813	1.5	0.12	0.85	0.065	O
vehicle1	KEEL	846	18	217	2.9	0.47	0.24	0.255	O
wisconsin	KEEL	683	9	239	1.9	0.07	0.91	0.047	O

KEEL: KEEL imbalanced data-set repository (Alcalá-Fdez et al. 2011). Data sets with more than 6000 examples were subsampled to 6000 in a stratified way. The 57 hold-out data sets are listed in the supplementary material.

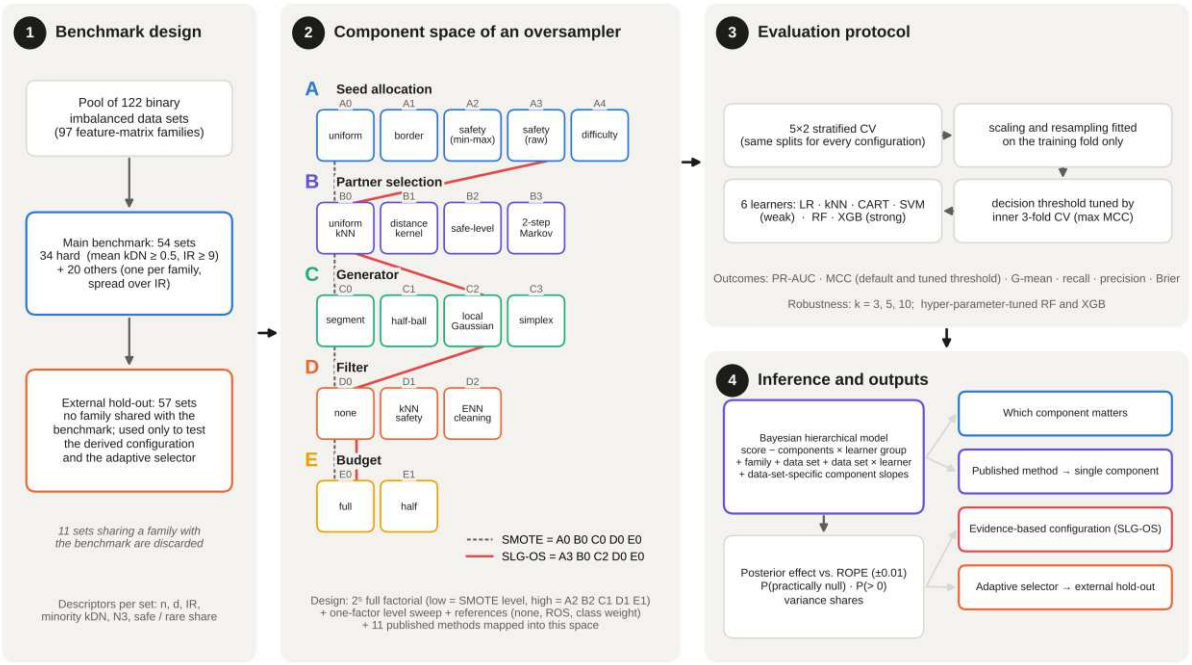


Fig. 1 Overview of the study. (1) Construction of the benchmark and of the external hold-out from a pool of 122 data sets. (2) The five-component space of a SMOTE-family oversampler with the implemented levels; the dashed grey path marks SMOTE and the red path the derived configuration SLG-OS. (3) Evaluation protocol. (4) Bayesian analysis and the four outputs of the study.

4.2 Configurations

The design has three parts. The first is a full 2^5 factorial in which each component is set either to its SMOTE level (A0, B0, C0, D0, E0) or to a plausible alternative (A2, B2, C1, D1, E1). It yields 32 configurations in which all main effects and interactions are estimable. The second is a one-factor level sweep that changes a single component to each of its remaining levels (A1, A3, A4, B1, B3, C2, C3 and D2) while the other components stay at the SMOTE level. The third adds three reference strategies, namely no balancing, random oversampling (ROS) and class weighting. Together these give 43 configurations. In addition, eleven published implementations were run under the same protocol: SMOTE, Borderline-SMOTE, ADASYN, SMOTE-ENN, SMOTE-Tomek, KMeans-SMOTE and SVM-SMOTE from imbalanced-learn (Lemaître et al. 2017), our implementations of Safe-Level-SMOTE (Bunkhumpornpat et al. 2009) and G-SMOTE (Douzas and Bacao 2019), a Python re-implementation of G-SMOTE-Safe from its published equations, since no Python implementation was available (Dünder et al. 2027), and a Python port of the

reference implementation of MCSMOTE (Sağlam and Cengiz 2025). All methods were run with their published default parameters, which were fixed before the experiments.

The modular implementation was validated against imbalanced-learn. Synthetic points of the SMOTE and Borderline-SMOTE levels lie on seed–neighbour segments, Borderline seeds belong to the same danger set, ADASYN allocation weights coincide exactly, and class counts after resampling agree. On six data sets, the difference in PR-AUC and MCC between our SMOTE, Borderline-SMOTE, ADASYN and SMOTE-ENN and the imbalanced-learn versions was between −0.004 and 0.000 on average, with a spread equal to that of two runs of our own implementation with different seeds. On the 54 benchmark data sets, the posterior difference between modular and imbalanced-learn SMOTE was below 0.002 in every metric and learner group.

4.3 Learners and protocol

Six learners were used: logistic regression, k-nearest neighbours (k = 5), CART, a support vector machine with radial basis kernel, a random forest with 100 trees and XGBoost with 200 trees of depth four (Cox 1958; Cover and Hart 1967; Breiman et al. 1984; Cortes and Vapnik 1995; Breiman 2001; Pedregosa et al. 2011; Chen and Guestrin 2016). Following Elor and Averbuch-Elor (2022), we distinguish weak learners (logistic regression, kNN, CART, SVM) from strong learners (random forest, XGBoost), since earlier studies found that the benefit of balancing depends mainly on this distinction. Each data set was split by five repetitions of stratified two-fold cross-validation (Dietterich 1998), with identical splits for all configurations and learners. Within each training fold, features were standardised and the oversampler was applied. The decision threshold was chosen by a separate three-fold cross-validation inside the training fold, in which each inner training part was resampled again, and the threshold maximising the MCC of the out-of-fold scores was retained. Test folds were never resampled. Algorithm 2 summarises the protocol.

Algorithm 2 Evaluation protocol for one data set

Input: data set D; configurations $\mathcal{C}$; learners $\mathscr{L}$
Output: fold-level scores for every configuration and learner

generate 5 × 2 stratified folds with a fixed seed; use the same folds for all c ∈ $\mathcal{C}$ and l ∈ $\mathscr{L}$
for each fold (D_{tr}, D_{te}) do
  fit a standard scaler on D_{tr} and apply it to D_{tr} and D_{te}
  for each configuration c ∈ $\mathcal{C}$ and learner l ∈ $\mathscr{L}$ do
    split D_{tr} into three stratified inner folds
    for each inner fold: resample the inner training part with c, fit l, score the inner validation part
    t* ← threshold maximising the MCC of the pooled out-of-fold scores
    D_{tr}' ← resample D_{tr} with c; fit l on D_{tr}' ▷ *test fold is never resampled*
    score D_{te}; record PR-AUC, Brier, and MCC, G-mean, recall, precision at 0.5 and at t*
end for
end for
return the 10 × |$\mathcal{C}$| × |$\mathscr{L}$| score records

4.4 Metrics

The primary metrics are the area under the precision–recall curve (PR-AUC), computed as average precision, which does not depend on a threshold (Davis and Goadrich 2006; Saito and Rehmsmeier 2015), and the MCC at the tuned threshold (Chicco and Jurman 2020). The MCC at the default threshold, the G-mean, recall, precision and the Brier score were recorded in addition, so that changes in the ranking of examples can be separated from changes in the operating point.

4.5 Bayesian hierarchical analysis

Let y_{dlc} denote a score averaged over the ten folds for data set d, learner l and configuration c. For the factorial part we fitted

$$y_{dlc} = \alpha_l + u_{f(d)} + u_{dl} + \Sigma_j \, x_{cj} \, (\beta_{lj} + b_{d,g(l),j}) + \Sigma_{j<h} \, x_{cj} \, x_{ch} \, \delta_{jh} + \varepsilon_{dlc} \qquad (2)$$

where $x_{cj} \in \{-1/2, +1/2\}$ codes the level of component j in configuration c, g(l) is the learner group, β_{lj} is the effect of component j for learner l (the expected high-minus-low difference), $b_{d,g,j} \sim N(0, \tau_{g,j}^2)$ is the data-set-specific deviation from it, δ_{jh} are two-way interactions, $u_{f(d)}$ and u_{dl} are random intercepts for the feature-matrix family and the data-set–learner combination, and $\varepsilon_{dlc} \sim N(0, \sigma_l^2)$. The effect of a component in a learner group is the unweighted mean of β_{lj} over the learners of the group, so that every learner counts equally. A separate data-set intercept was not included, because in the benchmark 54 data sets come from 53 families and the two variance components are not separately identifiable. For single comparisons, such as a level of the sweep or a published method against SMOTE, we modelled the fold-averaged difference Δ_{dl} as

$$\Delta_{dl} = \theta_l + v_{f(d),g(l)} + e_{dl}, \quad e_{dl} \sim N(0, s_l^2) \qquad (3)$$

with learner-specific mean differences θ_l, family effects $v_{f,g} \sim N(0, \omega_g^2)$ specific to the learner group, and learner-specific residual standard deviations. The group effect is again the unweighted mean of θ_l over the learners of the group. Effects were given $N(0, 0.05^2)$ priors, standard deviations of random effects half-normal priors with scale 0.05 to 0.2, and $\log s_l$ a $N(-3.5, 1)$ prior, all weakly informative for scores bounded in [0, 1] (Gelman 2006). A Student-t likelihood was also tried, but it shrank the large per-data-set gains of the generators towards zero and its estimates departed from the raw means, so the Gaussian likelihood was retained. Models were fitted with the No-U-Turn sampler (Hoffman and Gelman 2014) in NumPyro (Bingham et al. 2019; Phan et al. 2019) with four chains (1500 warm-up and 1500 retained draws per chain for model (3), 3000 and 3000 for the five comparisons whose $\hat{R}$ first exceeded 1.01, and 600 and 600 for model (2)), and convergence was checked with the rank-normalised split-$\hat{R}$ and the bulk effective sample size (Vehtari et al. 2021).

We report posterior means, 95% credible intervals and the posterior probability that an effect lies inside the region of practical equivalence (ROPE) of ±0.01, which we regard as the smallest difference of practical interest for both metrics (Kruschke 2018; Kruschke and Liddell 2018). Following Kruschke (2018), an effect is called practically equivalent to zero when its 95% interval lies inside the ROPE, practically relevant

when the interval lies entirely outside it, and undecided otherwise. In addition, we call an effect credible when its interval excludes zero, which indicates its direction but not its size. As a sensitivity analysis, all effects were also estimated by a non-parametric bootstrap over feature-matrix families.

4.6 Robustness checks and external validation

Two robustness runs repeated the analysis of the most important configurations on the benchmark. The first used neighbourhood sizes $k = 3$ and $k = 10$ instead of $k = 5$ for all components. The second replaced the default random forest and XGBoost by versions whose hyper-parameters were selected in every training fold from a six-point grid by inner cross-validation on PR-AUC (random forest: maximum number of features and minimum leaf size, 300 trees; XGBoost: maximum depth and learning rate, 300 trees), before the threshold was tuned (Kong et al. 2020). The configuration and the selector derived in Section 5.7 were finally evaluated on the 57 hold-out data sets with the same protocol. All code was written in Python with scikit-learn, imbalanced-learn and XGBoost (Pedregosa et al. 2011; Chen and Guestrin 2016; Lemaître et al. 2017).

5 Results

5.1 Where the variance lies

Before looking at individual components it is useful to ask how much of the variation in performance they can explain at all. Fig. 2 decomposes the variance of the fold-averaged scores in the factorial part of the design into sequential sums of squares. For the threshold-free PR-AUC and the tuned MCC, the data set alone accounts for 89% and 93% of the total variance, the data set–learner combination for a further 5%, and the learner for 5% (PR-AUC) and 1% (tuned MCC). The five component main effects together explain about 0.01% of the variance of either metric, and all two-way component interactions together less than 0.02%. The data-set-specific component effects, that is, the extent to which a component helps on some data sets and hurts on others, account for 0.3% (PR-AUC) and 0.5% (tuned MCC), roughly thirty times more than the average effects. Taken together, all terms that involve a component account for 0.3% of the variance of PR-AUC, 0.5% of the tuned MCC and 0.7% of the MCC at the default threshold. Only for the G-mean at the default threshold, which reacts strongly to the balance between recall and specificity, do they reach 4.2%, a first indication that much of what components do is to move the operating point.

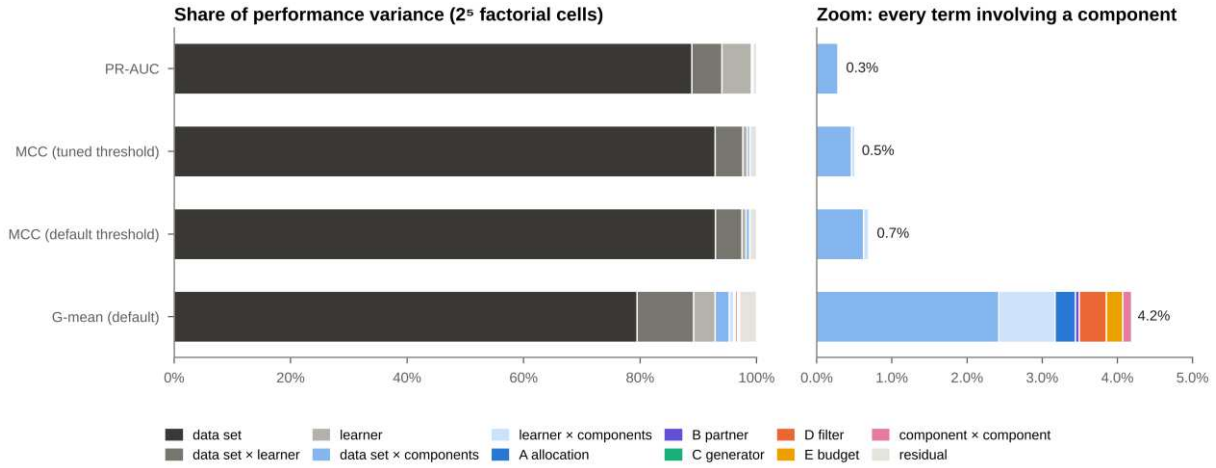


Fig. 2 Share of the variance of fold-averaged scores in the 2^5 factorial explained by data set, learner, components and their interactions (left), and the shares of all terms involving a component (right, enlarged scale). Shares are sequential (type I) sums of squares of a linear model with the data set × learner interaction, all two-way component interactions, and learner × component and data set × component terms; the residual contains the higher-order interactions.

Comparing component effects with the variance between data sets is not the whole story, since a practitioner cannot choose the data set. Within data sets, the component terms explain 2.8% of the variance of PR-AUC and 7.3% of the variance of the tuned MCC, and once the learner is fixed as well, they explain about a third of the remaining variation in both metrics. Almost all of this share, however, comes from the data-set-specific effects rather than from the average effects. Choosing an oversampler therefore matters for a given data set and learner, but which choice is best varies from data set to data set, and the average effects reported below must be read against this heterogeneity, which the hierarchical models represent explicitly.

5.2 Factorial main effects

Table 3 reports the posterior main effects of the five components in the 2^5 factorial, where each effect is the expected difference between the high and the low level averaged over the levels of all other components. For partner selection (safe-level versus uniform), the posterior mean was 0.000 [−0.001, 0.001] for weak and 0.000 [−0.003, 0.004] for strong learners in PR-AUC, with posterior probabilities of practical equivalence of 1.00 and 1.00. Safety allocation (A2), the kNN safety filter (D1) and the half budget (E1) had PR-AUC effects of 0.000 [−0.002, 0.003], +0.003 [−0.001, 0.006] and +0.002 [0.001, 0.003] over all six learners. The generator was the only component with an appreciable effect. Replacing segment interpolation by the half-ball generator changed PR-AUC by +0.010 [−0.004, 0.024] and the tuned MCC by +0.012 [−0.002, 0.025] for strong learners, with posterior probabilities of exceeding the ROPE of 0.54 and 0.62, whereas for weak learners the effects were +0.001 [−0.005, 0.007] and −0.002 [−0.010, 0.007].

Table 3 Posterior main effects (high minus low level) of the five components in the 2^5 factorial, from model (2): posterior mean, 95% credible interval and, in parentheses, the posterior probability that the effect lies inside the ROPE of ±0.01.

Component change	PR-AUC weak	PR-AUC strong	MCC (tuned) weak	MCC (tuned) strong
A0 → A2 safety allocation (min–max)	0.000 [−0.003, 0.003] (1.00)	+0.001 [−0.004, 0.005] (1.00)	−0.002 [−0.006, 0.001] (1.00)	0.000 [−0.005, 0.005] (1.00)

Component change	PR-AUC weak	PR-AUC strong	MCC (tuned) weak	MCC (tuned) strong
B0 → B2 safe-level partner	0.000 [−0.001, 0.001] (1.00)	0.000 [−0.003, 0.004] (1.00)	−0.001 [−0.003, 0.000] (1.00)	−0.001 [−0.005, 0.003] (1.00)
C0 → C1 half-ball generator	+0.001 [−0.005, 0.007] (1.00)	+0.010 [−0.004, 0.024] (0.46)	−0.002 [−0.010, 0.007] (0.97)	+0.012 [−0.002, 0.025] (0.38)
D0 → D1 kNN safety filter	+0.004 [0.001, 0.007] (1.00)	+0.001 [−0.008, 0.009] (0.98)	+0.001 [−0.003, 0.005] (1.00)	+0.001 [−0.005, 0.008] (1.00)
E0 → E1 half budget	+0.003 [0.002, 0.004] (1.00)	+0.001 [−0.000, 0.003] (1.00)	+0.001 [−0.000, 0.003] (1.00)	0.000 [−0.002, 0.002] (1.00)

Weak: LR, kNN, CART, SVM. Strong: RF, XGB. Maximum rank-normalised $\hat{R}$ of the reported effects: 1.028.

None of the ten two-way interactions was practically relevant. The largest in PR-AUC was the interaction between allocation and the filter (−0.006 [−0.007, −0.004]). Within the range of the design, the effect of a component therefore hardly depends on the levels of the others, so that components can be studied, and combined, one at a time. The between-data-set standard deviations of the component effects add an important qualification to the means. For the generator, τ was 0.047 for strong learners, larger than its mean effect, so that the half-ball generator improves PR-AUC substantially on some data sets and reduces it on others. For partner selection τ was 0.013, smaller but not negligible relative to the ROPE, so that even an inert average hides small gains and losses on individual data sets.

5.3 Single components

The level sweep extends the factorial to all implemented levels. Fig. 3 shows the posterior difference from SMOTE of each level, estimated with model (3) separately for weak and strong learners.

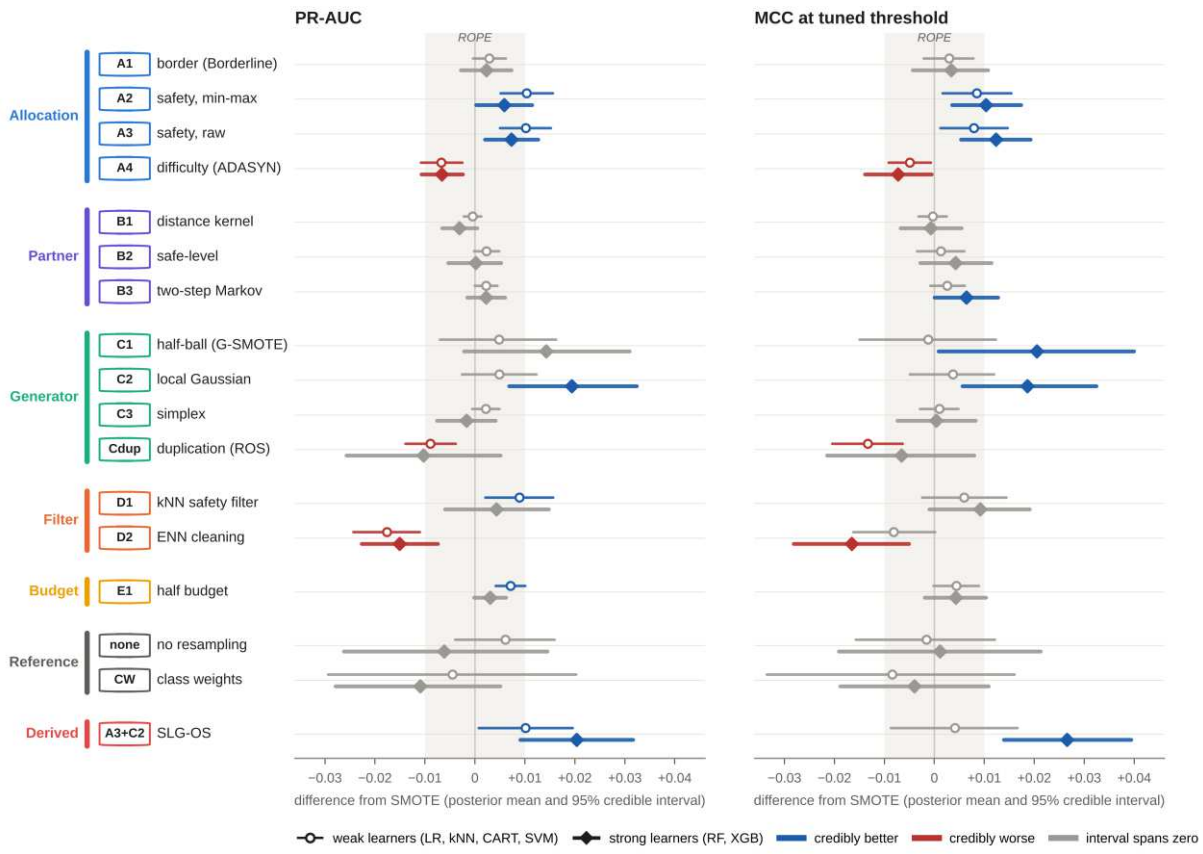


Fig. 3 Posterior difference from SMOTE (A0 B0 C0 D0 E0) when a single component is changed to the indicated level, from model (3). Points are posterior means and lines 95% credible intervals, for weak (open circles) and strong

learners (filled diamonds), each an unweighted mean over the learners of the group. Blue and red mark intervals that exclude zero. The shaded band is the ROPE of ±0.01: an interval inside it indicates practical equivalence, an interval entirely outside it a practically relevant effect.

Partner selection was inert at every level. The distance kernel (B1), the safe-level weights (B2) and the two-step Markov walk (B3) changed PR-AUC by at most 0.003 on average. For ten of the twelve combinations of level, learner group and metric, the 95% interval lay inside the ROPE, and the remaining two, the tuned MCC of strong learners under B2 and B3 (+0.006 [0.000, 0.013] for B3), were undecided with posterior probabilities of practical equivalence of 0.94 and 0.87. Given that partner rules are among the most frequently modified parts of SMOTE (Bunkhumpornpat et al. 2009; Maciejewski and Stefanowski 2011; Zhang and Li 2014; Sağlam and Cengiz 2025), this is perhaps the most consequential negative finding of the study.

Allocation had small effects whose sign depended on the rule. Safety-based allocation was credibly but modestly beneficial. Raw safety (A3) changed PR-AUC by +0.010 [0.005, 0.015] for weak and +0.007 [0.002, 0.013] for strong learners and the tuned MCC of strong learners by +0.012 [0.005, 0.019], and min–max scaled safety (A2) behaved almost identically. All of these effects were undecided with respect to the ROPE. Borderline allocation (A1) was practically equivalent to uniform allocation except for the tuned MCC of strong learners, where its interval barely touched the ROPE boundary. Difficulty-based allocation in the manner of ADASYN (A4) went in the opposite direction, with effects of −0.007 [−0.011, −0.003] (weak) and −0.007 [−0.011, −0.002] (strong) in PR-AUC. These effects are small, and most of their posterior mass lies inside the ROPE, but their direction is credible: concentrating synthetic points on the hardest minority examples, which are often noise or outliers (Napierała and Stefanowski 2016), makes classifiers slightly worse, whereas concentrating them on safe examples makes them slightly better.

The largest positive effects among single components came from volume-filling generators combined with strong learners. The local Gaussian generator (C2) improved PR-AUC by +0.019 [0.007, 0.032] and the tuned MCC by +0.019 [0.006, 0.032] for random forests and XGBoost, with posterior probabilities of exceeding the ROPE of 0.94 and 0.90. The half-ball generator (C1) had a similar mean effect on the tuned MCC (+0.020 [0.001, 0.040]) but a smaller and less certain effect on PR-AUC (+0.014 [−0.002, 0.031]). For weak learners, neither volume-filling generator differed credibly from segment interpolation (+0.005 [−0.003, 0.012] for C2 in PR-AUC). The simplex generator (C3), which still places points inside the convex hull of a few minority examples, was practically equivalent to SMOTE, and duplication (ROS) was credibly worse than interpolation for weak learners (−0.013 [−0.020, −0.006] in tuned MCC). The ordering of generators follows the volume they cover rather than their sophistication: the more the synthetic points spread beyond the segments and simplices spanned by existing minority examples, the more tree ensembles benefit.

Filtering produced the clearest harmful effect of the study. Edited-nearest-neighbour cleaning of the resampled sample (D2) reduced PR-AUC by −0.018 [−0.024, −0.011] for weak learners, an interval entirely

outside the ROPE, and by −0.015 [−0.023, −0.007] for strong learners, and it reduced the tuned MCC of strong learners by −0.017 [−0.028, −0.005]. Applied after SMOTE to the complete benchmark data sets, ENN removed on average 13% of the majority examples, 9% of the original minority examples and 6% of the synthetic points, that is, genuine examples of both classes in exactly the overlapping regions where the ranking of examples is decided. The kNN safety filter (D1), which discards only synthetic candidates and preserves the budget, was mildly positive instead (+0.009 [0.002, 0.016] in PR-AUC for weak learners). Halving the budget (E1) was practically equivalent, or nearly so, to full balancing.

Among the reference strategies, training on the original data without any resampling did not differ credibly from SMOTE once the threshold was tuned. The differences were +0.006 [−0.004, 0.016] (weak) and −0.006 [−0.026, 0.015] (strong) in PR-AUC and −0.002 [−0.016, 0.012] and +0.001 [−0.019, 0.021] in tuned MCC, with posterior means close to zero but intervals too wide to establish practical equivalence. Class weighting was similarly undecided (−0.011 [−0.028, 0.005] in PR-AUC for strong learners). These results agree with the conclusion of Elor and Averbuch-Elor (2022) that, with a tuned threshold, balancing by SMOTE offers no reliable advantage over no balancing at all.

The group averages conceal systematic differences between the learners, which Fig. 4 shows for every level of the sweep. The gains of the volume-filling generators were concentrated in the three tree-based learners: the local Gaussian generator improved the tuned MCC of CART, random forests and XGBoost by 0.017 to 0.026, left the SVM practically unchanged and lowered the tuned MCC of logistic regression and kNN slightly. The harm of ENN cleaning was largest for kNN (−0.040 in PR-AUC), whose decision rule depends directly on the examples that ENN removes. The reference strategies split the strong learners: without resampling, the tuned MCC of random forests was 0.023 higher than with SMOTE, whereas that of XGBoost was 0.021 lower, so that the near-zero group average of no resampling hides two opposite learner-specific effects. Safety allocation, by contrast, was positive or neutral for every learner.

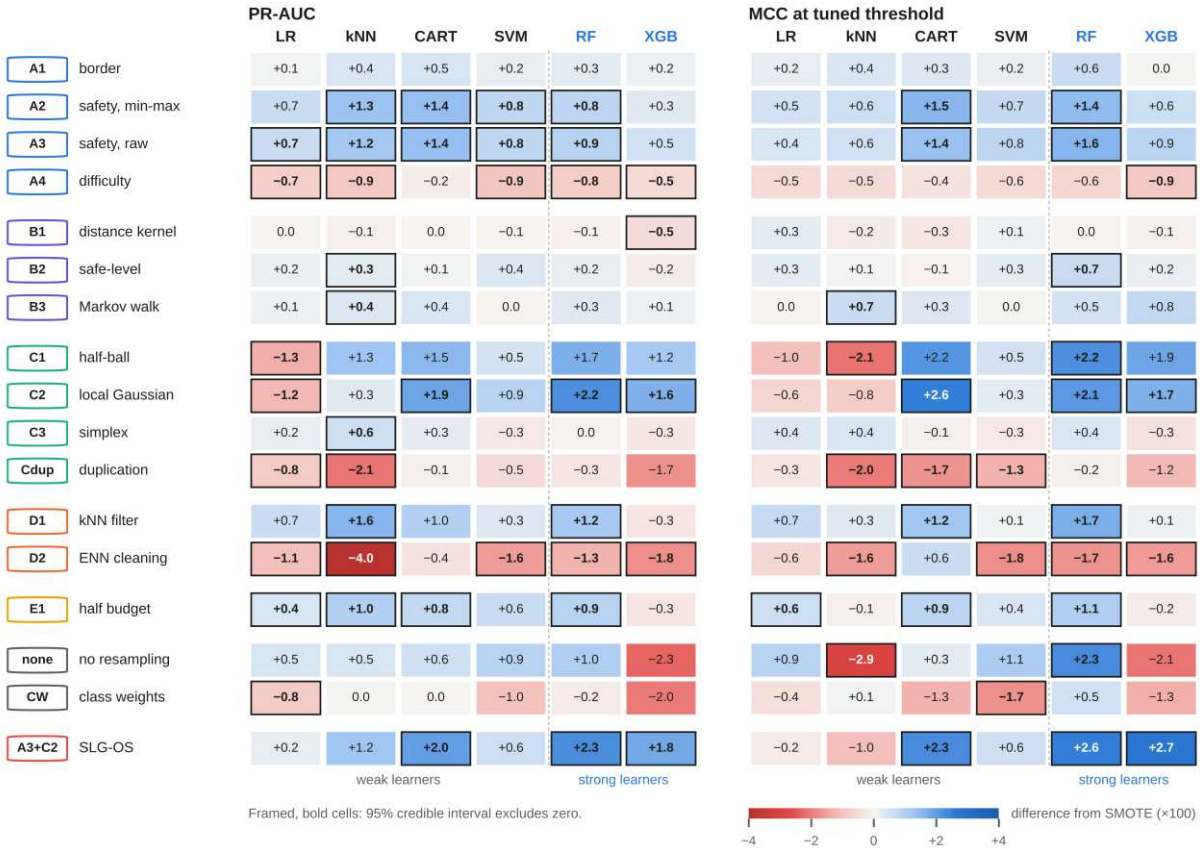


Fig. 4 Posterior mean difference from SMOTE (×100) of each level of the sweep for each of the six learners, from model (3). Columns left of the dashed line are the weak learners, those right of it the strong learners. Framed cells with bold numbers have a 95% credible interval that excludes zero.

5.4 Allocation and filtering move the operating point

The small effects of allocation, filtering and budget at the tuned threshold contrast with clear effects at the default threshold. Fig. 5 shows, for every level of the sweep, the change in recall, precision and MCC at the default threshold of 0.5 next to the change in MCC at the tuned threshold. The kNN safety filter reduced recall by 0.061 (weak) and 0.046 (strong) and raised precision by 0.050 and 0.043. Safety allocation (A2) and the half budget showed the same pattern at a smaller scale, with recall losses of 0.022 to 0.042 and precision gains of 0.015 to 0.034. At the tuned threshold, the same three components changed the MCC by at most 0.011 on average. Training without resampling is the extreme case of this mechanism. At the default threshold, it reduced recall by 0.287 for weak and 0.148 for strong learners and the MCC by 0.070 and 0.059, but at the tuned threshold its MCC differed from that of SMOTE by less than 0.003 on average.

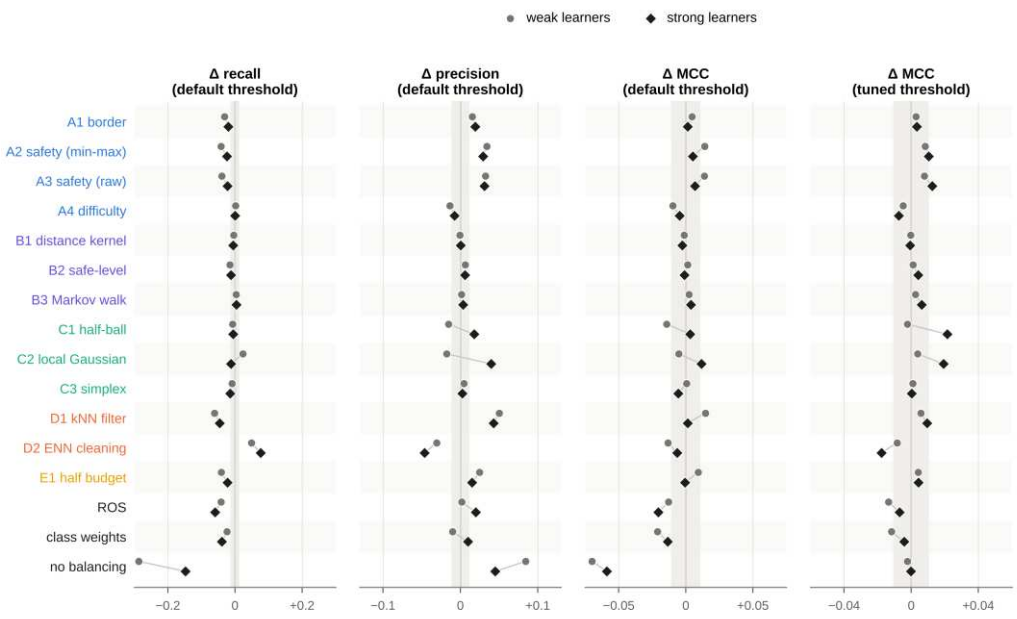


Fig. 5 Effect profile of each level of the sweep relative to SMOTE: change in recall, precision and MCC at the default threshold of 0.5 and in MCC at the tuned threshold, averaged over the benchmark for weak (grey circles) and strong learners (black diamonds). Large changes in recall and precision that disappear at the tuned threshold indicate a shift of the operating point.

The factorial confirms the same picture across all configurations. The main effects of safety allocation, the kNN filter and the half budget on recall at the default threshold were between −0.044 and −0.032 and their effects on precision between +0.019 and +0.039, all with intervals far from zero, while their effects on PR-AUC and tuned MCC were close to zero (Section 5.2). Components that make oversampling more conservative therefore mainly produce classifiers whose scores are shifted towards the majority class, and a tuned threshold absorbs most of the shift. The shift is not the only effect, since safety allocation and the kNN filter also had small credible effects on the threshold-free PR-AUC of weak learners (Section 5.3), but these were an order of magnitude smaller than the changes in recall and precision. Volume-filling generators behaved differently. They changed recall and precision at the default threshold only modestly, yet they changed PR-AUC most, so their effect lies in the ranking itself.

5.5 Published methods in component space

Fig. 6 compares, data set by data set, the gain of each published method over imbalanced-learn SMOTE with the gain of its single-component counterpart (Table 1) over the modular SMOTE. The modular and the imbalanced-learn SMOTE were practically equivalent in both metrics and learner groups. For Borderline-SMOTE (A1), ADASYN (A4) and SMOTE-ENN (D2), the agreement is partly a validation of the modular implementation, since these components re-implement the corresponding rules, although the published implementations differ in details such as the rounding of the ADASYN allocation and the streams of random numbers. G-SMOTE is more informative, because its default implementation also selects partners from the majority class and uses truncation and deformation parameters. For all four methods, the

posterior mean difference between method and component was below 0.006 in both metrics and learner groups, and the correlation between their per-data-set gains was 0.80 to 0.95 for all combinations except the tuned MCC of ADASYN for strong learners (0.67). The remaining design choices of these algorithms thus add nothing measurable to the effect of the single component. Safe-Level-SMOTE was close to, but not fully captured by, safety-weighted partner selection (correlation 0.61 to 0.87), consistent with its additional change of the interpolation range.

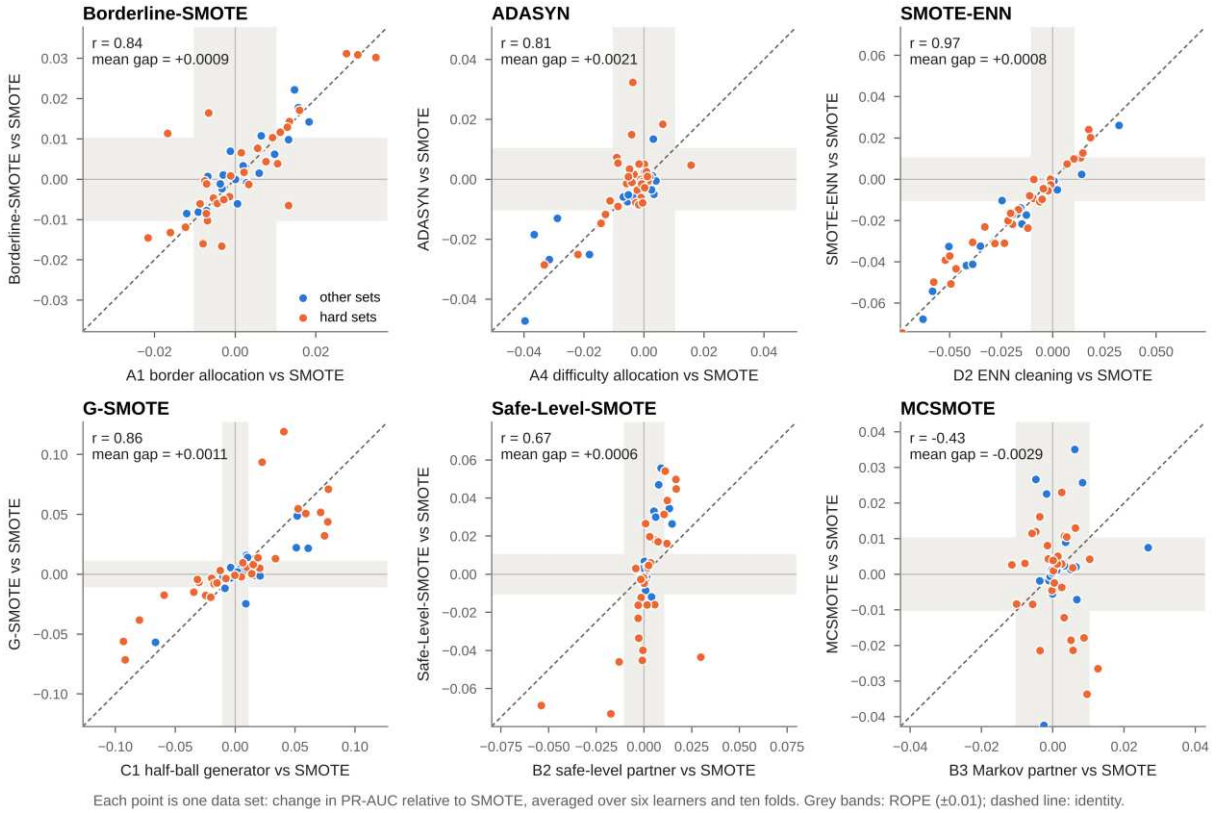


Fig. 6 Published methods against their single-component counterparts. Each point is a benchmark data set (orange: hard group). The horizontal axis shows the PR-AUC gain of the component level over the modular SMOTE and the vertical axis the gain of the published implementation over imbalanced-learn SMOTE, both averaged over all six learners and ten folds. The dashed line marks equal gains and the grey bands the ROPE; r is the Pearson correlation of the learner-averaged gains and the mean gap the average vertical distance from the diagonal.

Measured against SMOTE, the published methods reproduced the component results. G-SMOTE improved PR-AUC by +0.018 [0.002, 0.034] and the tuned MCC by +0.023 [0.005, 0.041] for strong learners, SVM-SMOTE had small credible gains (+0.009 [0.002, 0.016] in PR-AUC for strong learners), Borderline-SMOTE, SMOTE-Tomek and KMeans-SMOTE were practically equivalent to SMOTE, ADASYN was practically equivalent to SMOTE but credibly slightly worse in the PR-AUC of weak learners (−0.005 [−0.009, −0.001]), and SMOTE-ENN was clearly worse (−0.017 [−0.024, −0.010] and −0.015 [−0.022, −0.008] in PR-AUC). Among the methods that combine several modifications, the gravity-based G-SMOTE-Safe (Dünder et al. 2027) had small positive effects whose intervals included zero, for instance +0.005 [−0.001, 0.011] in the tuned MCC of weak learners, and MCSMOTE (Sağlam and Cengiz 2025) was practically equivalent to SMOTE in the PR-AUC of weak learners and undecided elsewhere,

with posterior means between −0.005 [−0.012, 0.003] (tuned MCC, weak) and −0.004 [−0.013, 0.004] (PR-AUC, strong). The gains of neither method correlated strongly with those of its closest single component, which suggests that their combined modifications do something different from any one component, but not something larger. Fig. 7 ranks all 17 strategies of the component sweep and the published methods, with family-bootstrap intervals for the mean ranks. For strong learners, the local Gaussian generator, raw safety allocation and SVM-SMOTE were among the first four in both metrics and G-SMOTE was third in tuned MCC, while SMOTE was in the lower third and ADASYN and SMOTE-ENN were last. The intervals of the leading strategies overlap widely, with mean ranks between about 7 and 8 and interval widths of about two ranks, so the figure supports a division into a leading group, a middle group around SMOTE and two clearly inferior methods rather than a strict ordering. For weak learners, raw safety allocation ranked first in both metrics and the spread of ranks was smaller, consistent with the smaller effects in this group.

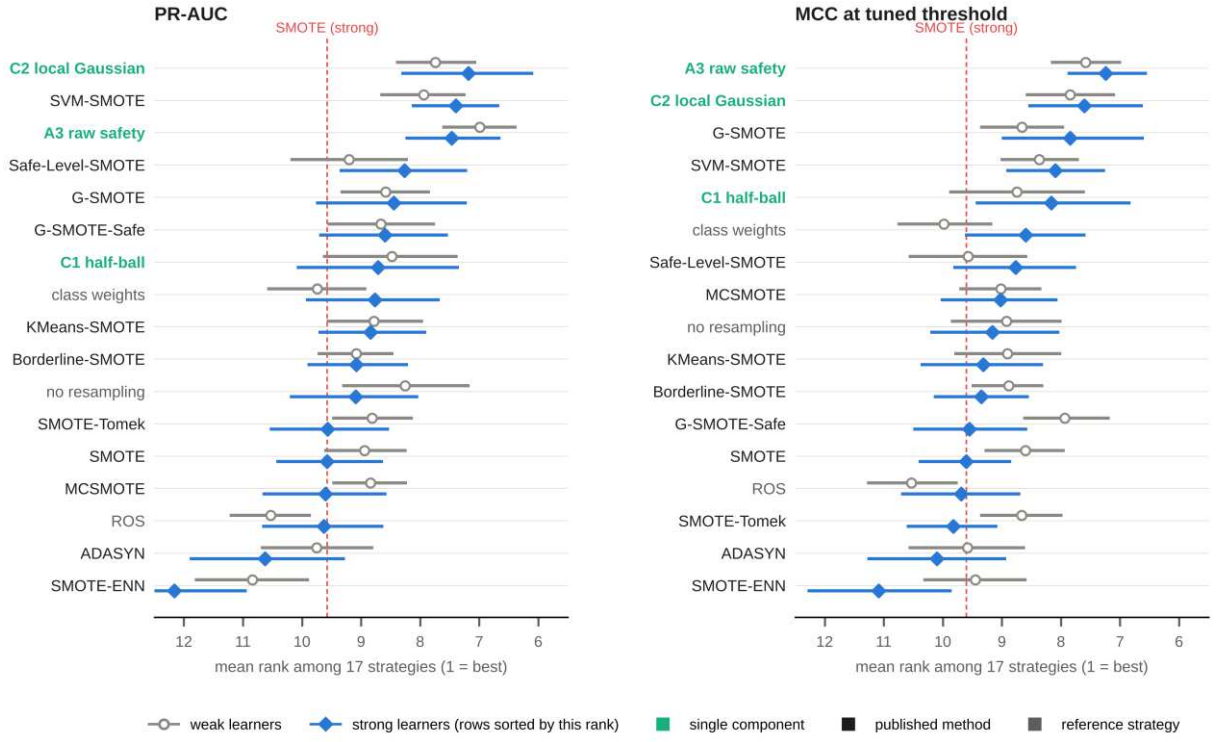


Fig. 7 Mean rank of the 17 strategies (1 = best) for strong learners (diamonds, rows sorted by this rank) and weak learners (circles), with 95% family-bootstrap intervals. Ranks were computed within each data set and learner and averaged over the learners of the group and over the 54 benchmark data sets. Colours of the labels distinguish single components, published methods and reference strategies; the dashed line marks the rank of SMOTE for strong learners.

5.6 Where the generator matters, and robustness

The generator effect was heterogeneous across data sets (Section 5.2). Fig. 8 places the benchmark and hold-out data sets in the plane spanned by the mean minority kDN score and the imbalance ratio and colours them by the PR-AUC gain of the local Gaussian generator for strong learners. In the hard region (kDN ≥ 0.5, IR ≥ 9) the mean gain was 0.027, compared with 0.008 among the remaining benchmark data sets. The share of data sets with a positive gain was, however, lower in the hard region (65%) than outside it (80%),

so the hard region contains both the largest gains and a larger share of losses. A regression of the per-data-set gain on kDN, log IR, log n and log p with family-clustered standard errors explained only 5% of its variance, and no coefficient was significant at the 5% level. The data-difficulty descriptors used here thus locate the generator effect only coarsely, a point we return to with the adaptive selector.

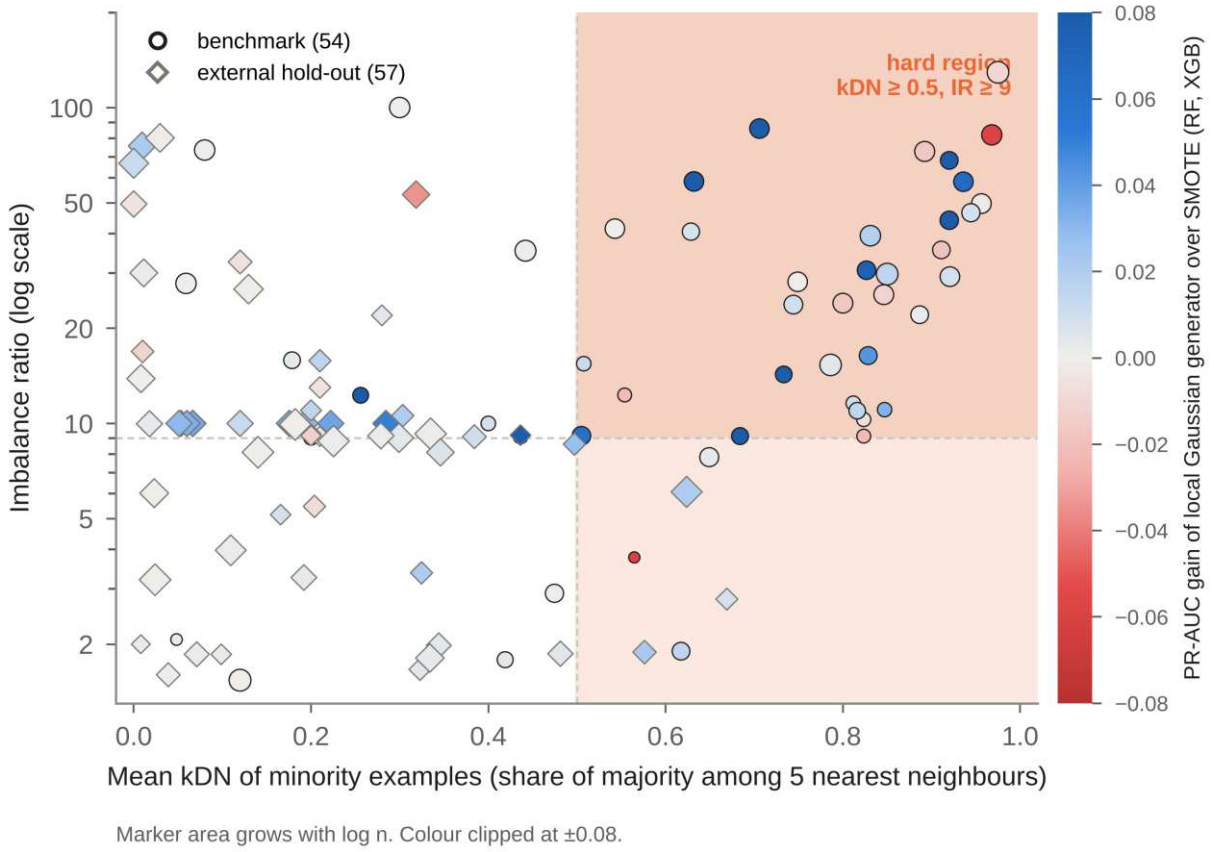


Fig. 8 PR-AUC gain of the local Gaussian generator over SMOTE for strong learners in the plane of minority kDN and imbalance ratio. Circles: benchmark; diamonds: external hold-out; marker area increases with log n. The shaded rectangle is the hard region (kDN ≥ 0.5, IR ≥ 9).

The main results did not depend on the neighbourhood size or on the hyper-parameters of the strong learners. These robustness runs were analysed with the family bootstrap. With k = 3, 5 and 10, the local Gaussian generator improved the PR-AUC of strong learners by 0.022, 0.020 and 0.019 and their tuned MCC by 0.021, 0.019 and 0.018, with intervals excluding zero in all cases. The half-ball generator weakened as k grew (PR-AUC gain 0.023, 0.015, 0.010), because its radius is set by an increasingly distant neighbour. Partner selection stayed within ±0.006 for every k. When random forests and XGBoost were tuned in every training fold, the PR-AUC gain of the local Gaussian generator was 0.021 [0.010, 0.033] and its tuned MCC gain 0.025 [0.011, 0.041], as large as with default hyper-parameters, and the harm of ENN cleaning persisted (−0.013 in PR-AUC, −0.020 in tuned MCC), whereas the small negative effect of difficulty-based allocation (−0.005) remained inside the ROPE. The objection that oversampling only compensates for poorly configured learners (Elor and Averbuch-Elor 2022) therefore does not apply to the generator effect.

5.7 An evidence-based configuration and an adaptive selector

The preceding results suggest a simple recipe. Partner selection can be left at its SMOTE default, ENN cleaning should be avoided, allocation should, if anything, favour safe minority examples, and the generator should fill volume. We therefore evaluated a configuration that combines the generator with the largest effect, the local Gaussian (C2), with the allocation rule with the most consistent small effect, raw safety (A3), together with uniform partners, no filter and full balancing. Safety allocation was preferred to the kNN safety filter, whose effects were similar, because it does not require generating three times as many candidates. We call the result safety-allocated local Gaussian oversampling (SLG-OS); Algorithm 3 gives its pseudocode. Its generator is a ROSE-type smoothed bootstrap (Menardi and Torelli 2014) with a local diagonal bandwidth, so SLG-OS is best understood as a principled combination of existing ideas rather than a new algorithm. Apart from the neighbourhood sizes k and m that SMOTE and Borderline-SMOTE also require, its only constant is the variance factor of one half, which was fixed in advance. Two variants, with min–max instead of raw safety and with half the budget, were statistically indistinguishable from SLG-OS, and the simplest variant was retained.

Algorithm 3 Safety-allocated local Gaussian oversampling (SLG-OS)

Input: training sample $D = P \cup N$; neighbourhood sizes $k = 5$, $m = 10$
Output: resampled training sample D'

$B \leftarrow n_- - n_+$
for each $x \in P$ do
  $r_m(x) \leftarrow |NN_m(x) \cap N| / m$ ▷ *local majority share*
  $w(x) \leftarrow 1 - r_m(x) + \varepsilon$ ▷ *raw safety, $\varepsilon = 10^{-3}$*
normalise w to sum to one
$S \leftarrow \emptyset$
repeat B times
  draw seed $x_s \sim w$
  $Q \leftarrow \{x_s\} \cup NN_k^P(x_s)$
$\sigma_j^2 \leftarrow \frac{1}{2} \cdot$ variance of feature j over Q, for $j = 1, \ldots, p$
draw $s \sim N(x_s, \mathrm{diag}(\sigma_1^2, \ldots, \sigma_p^2))$; $S \leftarrow S \cup \{s\}$
return $D' = D \cup \{(s, 1) : s \in S\}$

On the benchmark, SLG-OS improved on imbalanced-learn SMOTE by +0.021 [0.010, 0.033] in PR-AUC and +0.025 [0.012, 0.039] in tuned MCC for strong learners (Table 4), and the latter interval lay entirely outside the ROPE. For weak learners its PR-AUC gain was +0.011 [0.002, 0.021] and its tuned MCC was undecided (+0.004 [−0.009, 0.017]). Relative to its own components, SLG-OS was credibly better than raw safety allocation alone for strong learners (+0.014 [0.003, 0.025] in tuned MCC), but it did not differ credibly from the local Gaussian generator alone (+0.008 [−0.001, 0.016] in tuned MCC, 0.000 [−0.008, 0.008] in PR-AUC). It did not differ credibly from G-SMOTE (+0.002 [−0.009, 0.013]). Since the configuration was chosen after inspecting the benchmark results, the benchmark comparison is optimistic by construction, and selecting the best two of sixteen levels is subject to a winner's curse. The hold-out is the more relevant test.

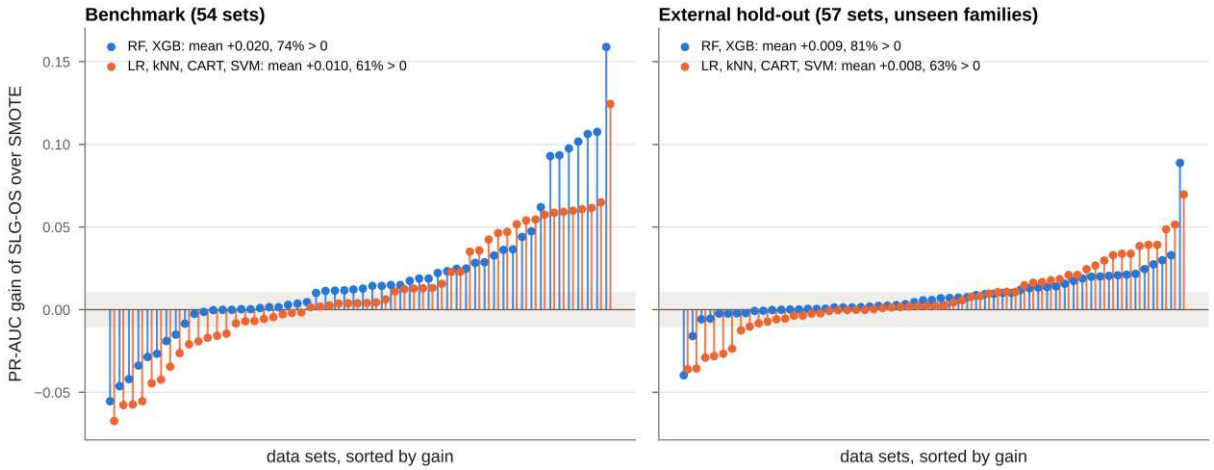


Fig. 9 Per-data-set PR-AUC gain of SLG-OS over SMOTE on the benchmark (left) and on the external hold-out (right), sorted by gain, for strong (blue) and weak learners (orange). The shaded band is the ROPE.

On the 57 hold-out data sets, SLG-OS improved on SMOTE for strong learners by +0.008 [0.002, 0.013] in PR-AUC and +0.007 [−0.000, 0.015] in tuned MCC (Fig. 9, Table 4), and its PR-AUC was higher than that of SMOTE on 81% of the hold-out data sets. The gains for strong learners were about a third of those on the benchmark, as expected from the winner's curse. The PR-AUC gain remained credible, the interval of the tuned MCC gain just reached zero, and both were mostly inside the ROPE, with posterior probabilities of practical equivalence of 0.80 and 0.74. For weak learners, the hold-out gains were +0.010 [0.006, 0.015] (PR-AUC) and +0.007 [0.002, 0.013] (tuned MCC), both credible and somewhat larger than on the benchmark. The two components contributed differently. For strong learners, SLG-OS was practically equivalent to the local Gaussian generator alone (+0.001 [−0.001, 0.003] in PR-AUC), so its advantage over SMOTE came from the generator, whereas for weak learners adding safety allocation to the generator produced a credible gain (+0.012 [0.008, 0.017] in PR-AUC and +0.011 [0.005, 0.016] in tuned MCC against C2). At the level of individual learners, the largest gains were obtained with XGBoost, random forests and CART (0.023 to 0.028 in tuned MCC on the benchmark) and the gains for logistic regression and the SVM were close to zero. For kNN, SLG-OS improved PR-AUC but lowered the tuned MCC relative to SMOTE (−0.011 on the benchmark, −0.008 on the hold-out). On average SLG-OS was not credibly worse than SMOTE in either learner group or set of data sets, but it lost more than 0.01 against SMOTE on 4% to 19% of the individual hold-out data sets, depending on metric and learner group.

Table 4 SLG-OS and the adaptive selector on the benchmark and on the external hold-out: posterior mean difference, 95% credible interval and posterior probability of practical equivalence (in parentheses), from model (3).

Comparison	PR-AUC weak	PR-AUC strong	MCC (tuned) weak	MCC (tuned) strong
Benchmark (54 data sets)				
SLG-OS vs SMOTE	+0.011 [0.002, 0.021] (0.41)	+0.021 [0.010, 0.033] (0.03)	+0.004 [−0.009, 0.017] (0.79)	+0.025 [0.012, 0.039] (0.01)
SLG-OS vs C2 local Gaussian	+0.005 [0.000, 0.010] (0.96)	0.000 [−0.008, 0.008] (0.99)	0.000 [−0.007, 0.008] (0.99)	+0.008 [−0.001, 0.016] (0.71)
SLG-OS vs A3 raw safety	0.000 [−0.006, 0.007] (1.00)	+0.013 [0.004, 0.022] (0.24)	−0.004 [−0.013, 0.005] (0.91)	+0.014 [0.003, 0.025] (0.22)
SLG-OS vs G-SMOTE	+0.007 [−0.000, 0.014] (0.84)	+0.003 [−0.008, 0.015] (0.86)	+0.002 [−0.006, 0.010] (0.97)	+0.002 [−0.009, 0.013] (0.92)
External hold-out (57 data sets)				

Comparison	PR-AUC weak	PR-AUC strong	MCC (tuned) weak	MCC (tuned) strong
SLG-OS vs SMOTE	+0.010 [0.006, 0.015] (0.43)	+0.008 [0.002, 0.013] (0.80)	+0.007 [0.002, 0.013] (0.82)	+0.007 [−0.000, 0.015] (0.74)
SLG-OS vs C2 local Gaussian	+0.012 [0.008, 0.017] (0.16)	+0.001 [−0.001, 0.003] (1.00)	+0.011 [0.005, 0.016] (0.41)	+0.001 [−0.004, 0.005] (1.00)
SLG-OS vs A3 raw safety	+0.003 [0.000, 0.006] (1.00)	+0.005 [0.001, 0.009] (0.99)	+0.001 [−0.002, 0.005] (1.00)	+0.007 [0.000, 0.014] (0.81)
SLG-OS vs no resampling	+0.006 [0.002, 0.010] (0.97)	+0.006 [0.001, 0.011] (0.94)	+0.012 [0.007, 0.017] (0.23)	+0.010 [0.001, 0.018] (0.53)
Adaptive selector vs SMOTE	+0.011 [0.005, 0.016] (0.41)	+0.008 [0.002, 0.014] (0.74)	+0.009 [0.003, 0.015] (0.62)	+0.008 [0.000, 0.016] (0.67)

On the benchmark, SMOTE is the imbalanced-learn implementation; on the hold-out, it is the modular A0 B0 C0 D0 E0. The configuration was chosen on the benchmark, so only the hold-out comparisons are free of selection effects.

Because the best component differed between data sets, we finally asked whether data descriptors can predict it. For each learner, a random forest regressor was trained to predict the gain over SMOTE of each candidate strategy from eight meta-features (minority kDN, the shares of safe and of rare or outlying minority examples (Napierała and Stefanowski 2016), N3 (Ho and Basu 2002), and the logarithms of IR, n, p and n/p), and the strategy with the largest predicted gain was selected (Rice 1976; Moniz and Cerqueira 2021; Brazdil et al. 2022). Ridge regression and a depth-two classification tree were tried as alternative meta-learners under the same validation. The tree was worse throughout, and ridge regression was similar for strong but clearly worse for weak learners, so only the random forest selector was carried forward to the hold-out. Under leave-one-family-out validation on the benchmark, with seven candidates (SMOTE, no resampling, A3, C1, C2, A2 combined with C1, and G-SMOTE), the selector improved on SMOTE by 0.023 (PR-AUC) and 0.024 (tuned MCC) for strong and 0.022 and 0.021 for weak learners, compared with 0.021 and 0.018 for the fixed local Gaussian generator for strong learners. On the hold-out, where G-SMOTE had not been run and the candidate set therefore contained six strategies, the selector trained on all 54 benchmark data sets improved on SMOTE by +0.008 [0.002, 0.014] (PR-AUC, strong) and +0.011 [0.005, 0.016] (PR-AUC, weak), gains that are practically identical to those of the fixed configuration SLG-OS in both learner groups (Table 4). An oracle that picks the best strategy for every hold-out data set would have gained 0.017 to 0.027, an upward-biased bound, since it maximises over noisy test-set estimates, but one that indicates room for better selectors. The meta-features used here capture only a small part of it.

6 Discussion

The central result of this study is that most of the design space of SMOTE-family oversampling is flat. Of the five components that we varied, partner selection was practically inert in both learner groups, in both metrics and for every neighbourhood size, allocation, filtering and budget mainly changed the operating point of the classifier rather than the ranking it learnt, and the largest changes in the ranking came from volume-filling generators, and only for tree ensembles. In a literature in which a large share of new methods modify which minority examples are oversampled and with which neighbour they are paired (Han et al. 2005; He et al. 2008; Bunkhumpornpat et al. 2009; Maciejewski and Stefanowski 2011; Barua et al. 2014; Zhang and Li 2014; Sağlam and Cengiz 2025; Dünder et al. 2027), these findings shift the burden of proof. A new allocation or partner rule should be expected to improve on SMOTE at a tuned threshold, not only

at the default threshold, before it can be regarded as an improvement, and its evaluation should include the corresponding single-component baseline, so that the contribution of the new rule can be separated from that of components it shares with other methods.

The mechanism behind the operating-point effects is straightforward. Every component that makes oversampling more conservative, by placing synthetic points in safe regions, discarding unsafe candidates or creating fewer points, reduces the shift of the decision function towards the minority class that full SMOTE induces. At the default threshold this appears as a gain in precision and a loss in recall, and since recall and precision enter the MCC and the G-mean differently, the same component can look beneficial under one metric and harmful under another. Once the threshold is chosen on held-out data, most of the shift is undone. This reading agrees with theoretical work showing that rebalancing mainly corrects the prior implicit in a classifier's scores (Provost 2000; Elkan 2001; Sakho et al. 2026) and with empirical work showing that threshold moving recovers much of the effect of resampling (Zou et al. 2016; Esposito et al. 2021; Elor and Averbuch-Elor 2022). It also helps to explain why comparisons of oversamplers at a fixed threshold, which are still common, can produce contradictory rankings across studies (Kovács 2019a; Musaev 2026): they partly measure where a method places the operating point.

The generator result requires a different explanation. Segment interpolation and simplex sampling place synthetic points inside the convex hull of a few minority examples, and the distribution of SMOTE points is concentrated around the existing ones (Elreedy and Atiya 2019; Elreedy et al. 2024). Volume-filling generators, such as the half-ball of G-SMOTE and the local Gaussian used here, also place points in directions orthogonal to the segments and therefore change the shape of the minority region seen by the learner, not only its density. The local Gaussian generator is, in effect, a local version of the smoothed bootstrap that Menardi and Torelli (2014) proposed in ROSE, and our results suggest that this old idea deserves more attention than the many interpolation variants that followed it. For axis-aligned tree ensembles, whose splits are placed where the class proportions change, a smoother and wider minority region plausibly makes minority areas easier to isolate by splits, which would explain why the gain is concentrated in random forests, XGBoost and, to a lesser extent, CART, and why it is largest on average when minority examples are embedded in the majority class. For kNN, by contrast, synthetic points that leave the minority region become neighbours of majority test examples, and for linear models and kernel SVMs, which are already smooth in the input space, there is little to gain. That tree ensembles remain the strongest learners on tabular data (Grinsztajn et al. 2022) makes this a practically relevant combination, even though the effect is small.

The harmful components deserve separate comment, because both are widely used. SMOTE-ENN is one of the most frequently cited hybrid methods (Batista et al. 2004), yet in our experiments it was among the worst strategies in both learner groups and both metrics, and the harm persisted when strong learners were tuned. Its cleaning step removed genuine examples of both classes from overlapping regions, and in data sets in which most minority examples are borderline or rare, as in the hard region of our benchmark,

this removes much of the information that is needed to rank examples correctly. The difficulty-based allocation of ADASYN follows the opposite logic and concentrates synthetic points on exactly those examples, which, if they are noise or outliers (Napierała et al. 2010; Napierała and Stefanowski 2016), amplifies them, although its harm was small enough to remain inside the ROPE. That allocating to safe examples helps slightly while allocating to difficult examples hurts slightly is consistent with earlier observations on the influence of minority example types on SMOTE (Skryjomski and Krawczyk 2017) and with the general finding that noisy and borderline examples are a main obstacle in imbalanced learning (Napierała et al. 2010; Vuttipittayamongkol et al. 2021).

The correspondence between published methods and single components has implications beyond these particular methods. If Borderline-SMOTE, ADASYN, SMOTE-ENN and G-SMOTE behave, data set by data set, like SMOTE with one component replaced, then benchmarks that include several of them partly count the same effect more than once, and ranking-based comparisons over many methods can be dominated by the number of variants that share a component. A component-level baseline would make such comparisons more informative. Methods that combine several modifications, such as G-SMOTE-Safe and MCSMOTE, were not reducible to one component, but they did not exceed the best single component either. Combinations are not helpful in themselves; they help when they combine helpful components, which in our study meant a volume-filling generator and, for weak learners, safety allocation, the combination that SLG-OS formalises.

For practitioners, the results suggest a short sequence of decisions. The decision threshold should be tuned on held-out data in any case, since this alone removes most of the differences between oversamplers and between oversampling and no oversampling. With a tuned threshold and a weak learner, the choice of oversampler matters less, and no resampling is a defensible default. With a tree ensemble, a volume-filling generator is worth trying, and SLG-OS or the local Gaussian generator alone are simple choices that retained a small advantage over SMOTE on unseen data sets. For weak learners, SLG-OS gave small but credible gains on the hold-out, which came from combining safety allocation with the generator. ENN cleaning should be avoided, partner rules need not be considered at all, and for kNN the gain of volume-filling generators in PR-AUC does not carry over to the tuned MCC.

For researchers, the study suggests that new oversamplers should be compared at both a default and a tuned threshold with threshold-free metrics as the primary criterion, that ablation studies should include single-component baselines from the existing design space and not only the parts of the new method, and that comparisons should account for the dependence between data sets derived from the same feature matrix and should report practical equivalence rather than only significance. With a ROPE, absence of evidence and evidence of absence can be distinguished, and our conclusions about partner selection are of the second kind, whereas several of the reference strategies remained undecided despite 54 data sets.

7 Limitations

Several limitations qualify these conclusions. The component space is not exhaustive. We implemented five components with two to five levels each and did not cover clustering-based allocation, iterative filters such as SMOTE-IPF, or generators based on learnt densities or deep generative models (Das et al. 2015; Sáez et al. 2015; Douzas and Bacao 2018). The finding that partner selection is inert, for example, refers to the four rules studied here, although these include the safe-level and Markov-chain ideas that motivate many recent methods. The factorial part of the design compares only two levels per component, so that interactions between other levels, for instance between ENN cleaning and a volume-filling generator, are not estimated, although the absence of interactions in the factorial makes large interactions elsewhere unlikely.

The benchmark contains 54 data sets, most of them from the KEEL repository, with a deliberate emphasis on hard problems with overlapping classes and high imbalance. This emphasis increases the power to detect component effects where they are expected to be largest, but it also means that the average effects describe a population of rather difficult problems. The composition of the benchmark, including the thresholds of the hard group, was fixed after an initial run of all configurations on the full pool of 122 data sets, and although the thresholds follow established definitions (Alcalá-Fdez et al. 2011; López et al. 2013; Napierała and Stefanowski 2016), the benchmark results should not be read as estimates for imbalanced data in general. On the full pool of 122 data sets (Supplementary Table S8), every main finding had the same direction, and the generator effect for strong learners was smaller, about 0.014 in both PR-AUC and tuned MCC instead of about 0.020, because the pool contains many easy problems on which oversampling matters little. The external hold-out of 57 data sets from disjoint families mitigates the selection concern for SLG-OS and the selector, but it is not untouched in the strictest sense, since its data sets were part of that initial run, and the published methods were not run on it, so the equivalence of SLG-OS and G-SMOTE has not been tested out of sample. All data sets are of small to moderate size (at most 6000 observations after subsampling) and moderate dimension, and high-dimensional problems, for which SMOTE is known to behave differently (Blagus and Lusa 2013), are underrepresented.

The learners were limited to six standard algorithms with mostly default hyper-parameters. The robustness run with tuned random forests and XGBoost confirmed the main results for strong learners, but we did not tune the weak learners, did not include neural networks or other gradient-boosting libraries, and did not combine oversampling with cost-sensitive learning or ensembles designed for imbalance (Chawla et al. 2003; Seiffert et al. 2010; Galar et al. 2012). The decision threshold was tuned for the MCC only. Other criteria lead to other thresholds, although the conclusion that components shift the operating point does not depend on the criterion. Calibration was recorded through the Brier score but not analysed in detail, and since oversampling is known to distort probability estimates (Dal Pozzolo et al. 2015; van den Goorbergh et al. 2022), the configurations discussed here are not intended for applications that require

calibrated risks without recalibration. The variance factor of the local Gaussian generator was fixed at one half and not varied, so the sensitivity of the generator effect to this constant is unknown.

On the statistical side, the models treat fold-averaged scores as Gaussian with learner-specific residual variances, which is an approximation for bounded scores, and they ignore the variation between folds within a data set. The Student-t alternative shrank large per-data-set gains excessively, and the Gaussian estimates agreed closely with the model-free family bootstrap. Because 54 data sets come from 53 families, family and data-set effects could not be separated in the benchmark. The rank-normalised $\hat{R}$ was at most 1.006 for the effects of all single comparisons and 1.028 for the effects in the larger factorial model, and some variance parameters mixed more slowly ($\hat{R}$ up to 1.095); the factorial estimates nevertheless agreed with the family bootstrap to within 0.001. Each comparison was fitted as a separate model, so the estimates are not pooled across the levels of the sweep, and the many comparisons reported here increase the chance that some credible effects are false positives, which is one more reason to give the hold-out results more weight than the benchmark results for the derived configuration. Finally, cross-platform runs showed that 85% of the fold-level results were bit-identical and that the remaining differences, mostly from multi-threaded XGBoost, averaged 0.0001. Exact reproduction requires the library versions listed in the supplementary material.

8 Conclusion

We have decomposed SMOTE-family oversampling into five interchangeable components and estimated their effects in a factorial design on 54 imbalanced data sets with six learners, a tuned decision threshold and Bayesian hierarchical models. Partner selection, a component often modified in the literature, was practically inert. Allocation, filtering and budget mostly moved the operating point, an effect that a tuned threshold largely removes, ENN cleaning was consistently harmful and difficulty-based allocation slightly so. Volume-filling generators combined with tree ensembles produced the largest gains, about two points of PR-AUC and tuned MCC, which were robust to the neighbourhood size and to hyper-parameter tuning, larger on average in data sets with strong class overlap, and smaller on a broader pool of data sets. Four widely used methods behaved like single components, and component terms explained less than one percent of the variance of PR-AUC and tuned MCC, although within a given data set the choice of oversampler can matter. The configuration derived from these results, safety-allocated local Gaussian oversampling, retained a smaller advantage over SMOTE on 57 hold-out data sets, credible in PR-AUC for both learner groups, and a meta-feature-based selector added nothing beyond it. Future work should extend the component space to clustering-based allocation and learnt generators, relate the generator effect to better descriptors of local class geometry, and develop selectors that close the gap to the oracle. The modular implementation and all results are openly available to support such work.

Declarations

Funding. This work was supported and funded by the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University (IMSIU) (grant number IMSIU-DDRSP2702).

Competing interests. The authors declare no competing interests.

Data availability. All data sets, the modular oversampler, the experiment scripts, the raw fold-level results and the analysis code are available on Zenodo at https://doi.org/10.5281/zenodo.23396738.

Code availability. The Python implementation of the modular oversampler, SLG-OS and the adaptive selector is included in the Zenodo archive, together with the exact library versions and random seeds used.

Author contributions. M.A. Cengiz: conceptualisation, methodology, supervision, writing (review and editing). E. Dünder: data curation, software, formal analysis, investigation, visualisation, writing (original draft).

Use of generative AI. During the preparation of this work the authors used Claude (Anthropic) to assist with code development and language editing. The authors reviewed and edited all content and take full responsibility for the content of the publication.

Ethics approval. Not applicable.

References

Alcalá-Fdez J, Sánchez L, García S, del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F (2009) KEEL: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput 13(3):307–318. https://doi.org/10.1007/s00500-008-0323-y

Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Mult-Valued Log Soft Comput 17(2–3):255–287

Andersen AK, Mehdizavareh H, Khan A, Becher T, Britsch S, Britsch M, Bøttcher M, Winther S, Rohde PD, Jensen MH, Cichosz SL (2026) Tipping the balance: impact of class imbalance correction on the performance of clinical risk prediction models. arXiv:2603.00208

Barua S, Islam MM, Yao X, Murase K (2014) MWMOTE—majority weighted minority oversampling technique for imbalanced data set learning. IEEE Trans Knowl Data Eng 26(2):405–425. https://doi.org/10.1109/TKDE.2012.232

Batista GEAPA, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 6(1):20–29. https://doi.org/10.1145/1007730.1007735

Bej S, Davtyan N, Wolfien M, Nassar M, Wolkenhauer O (2021) LoRAS: an oversampling approach for imbalanced datasets. Mach Learn 110(2):279–301. https://doi.org/10.1007/s10994-020-05913-4

Bellinger C, Drummond C, Japkowicz N (2016) Beyond the boundaries of SMOTE: a framework for manifold-based synthetically oversampling. In: Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2016). Springer, Cham, pp 248–263. https://doi.org/10.1007/978-3-319-46128-1_16

Bellinger C, Drummond C, Japkowicz N (2018) Manifold-based synthetic oversampling with manifold conformance estimation. Mach Learn 107(3):605–637. https://doi.org/10.1007/s10994-017-5670-4

Benavoli A, Corani G, Mangili F, Zaffalon M, Ruggeri F (2014) A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. In: Proceedings of the 31st International Conference on Machine Learning (ICML 2014), PMLR 32:1026–1034

Benavoli A, Corani G, Demšar J, Zaffalon M (2017) Time for a change: a tutorial for comparing multiple classifiers through Bayesian analysis. J Mach Learn Res 18(77):1–36

Bingham E, Chen JP, Jankowiak M, Obermeyer F, Pradhan N, Karaletsos T, Singh R, Szerlip P, Horsfall P, Goodman ND (2019) Pyro: deep universal probabilistic programming. J Mach Learn Res 20(28):1–6

Blagus R, Lusa L (2013) SMOTE for high-dimensional class-imbalanced data. BMC Bioinformatics 14:106. https://doi.org/10.1186/1471-2105-14-106

Boughorbel S, Jarray F, El-Anbari M (2017) Optimal classifier for imbalanced data using Matthews correlation coefficient metric. PLoS ONE 12(6):e0177678. https://doi.org/10.1371/journal.pone.0177678

Branco P, Torgo L, Ribeiro RP (2016) A survey of predictive modeling on imbalanced domains. ACM Comput Surv 49(2):1–50. https://doi.org/10.1145/2907070

Brazdil P, van Rijn JN, Soares C, Vanschoren J (2022) Metalearning: applications to automated machine learning and data mining, 2nd edn. Springer, Cham (Cognitive Technologies). https://doi.org/10.1007/978-3-030-67024-5

Breiman L, Friedman JH, Olshen RA, Stone CJ (1984) Classification and regression trees. Routledge, New York. https://doi.org/10.1201/9781315139470

Breiman L (2001) Random forests. Mach Learn 45(1):5–32. https://doi.org/10.1023/A:1010933404324

Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C (2009) Safe-Level-SMOTE: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In: Advances in Knowledge Discovery and Data Mining (PAKDD 2009), Lecture Notes in Computer Science, vol 5476. Springer, Berlin, pp 475–482. https://doi.org/10.1007/978-3-642-01307-2_43

Carvalho M, Pinho AJ, Brás S (2025) Resampling approaches to handle class imbalance: a review from a data perspective. J Big Data 12:71. https://doi.org/10.1186/s40537-025-01119-4

Chambers JM, Hastie TJ (eds) (1992) Statistical models in S. Wadsworth & Brooks/Cole, Pacific Grove, CA

Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357. https://doi.org/10.1613/jair.953

Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) SMOTEBoost: improving prediction of the minority class in boosting. In: Knowledge Discovery in Databases: PKDD 2003, Lecture Notes in Computer Science. Springer, Berlin, pp 107–119. https://doi.org/10.1007/978-3-540-39804-2_12

Chen T, Guestrin C (2016) XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp 785–794. https://doi.org/10.1145/2939672.2939785

Chicco D, Jurman G (2020) The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics 21:6. https://doi.org/10.1186/s12864-019-6413-7

Collell G, Prelec D, Patil KR (2018) A simple plug-in bagging ensemble based on threshold-moving for classifying binary and multiclass imbalanced data. Neurocomputing 275:330–340. https://doi.org/10.1016/j.neucom.2017.08.035

Corani G, Benavoli A (2015) A Bayesian approach for comparing cross-validated algorithms on multiple data sets. Mach Learn 100(2–3):285–304. https://doi.org/10.1007/s10994-015-5486-z

Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297. https://doi.org/10.1007/BF00994018

Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27. https://doi.org/10.1109/TIT.1967.1053964

Cox DR (1958) The regression analysis of binary sequences. J R Stat Soc Ser B 20(2):215–232. https://doi.org/10.1111/j.2517-6161.1958.tb00292.x

Dal Pozzolo A, Caelen O, Johnson RA, Bontempi G (2015) Calibrating probability with undersampling for unbalanced classification. In: 2015 IEEE Symposium Series on Computational Intelligence. IEEE, pp 159–166. https://doi.org/10.1109/SSCI.2015.33

Das B, Krishnan NC, Cook DJ (2015) RACOG and wRACOG: two probabilistic oversampling techniques. IEEE Trans Knowl Data Eng 27(1):222–234. https://doi.org/10.1109/TKDE.2014.2324567

Davis J, Goadrich M (2006) The relationship between precision-recall and ROC curves. In: Proceedings of the 23rd International Conference on Machine Learning (ICML 2006). ACM, pp 233–240. https://doi.org/10.1145/1143844.1143874

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Dietterich TG (1998) Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput 10(7):1895–1923. https://doi.org/10.1162/089976698300017197

Douzas G, Bacao F (2018) Effective data generation for imbalanced learning using conditional generative adversarial networks. Expert Syst Appl 91:464–471. https://doi.org/10.1016/j.eswa.2017.09.030

Douzas G, Bacao F, Last F (2018) Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. Inf Sci 465:1–20. https://doi.org/10.1016/j.ins.2018.06.056

Douzas G, Bacao F (2019) Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE. Inf Sci 501:118–135. https://doi.org/10.1016/j.ins.2019.06.007

Dünder E, Cengiz MA, Hawrami ZSM, Alharthi A (2027) Gravity-based synthetic oversampling: directing minority samples toward safe regions for imbalanced data. Appl Soft Comput 204:116594. https://doi.org/10.1016/j.asoc.2026.116594

Elkan C (2001) The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), pp 973–978

Elor Y, Averbuch-Elor H (2022) To SMOTE, or not to SMOTE? arXiv:2201.08528

Elreedy D, Atiya AF (2019) A comprehensive analysis of synthetic minority oversampling technique (SMOTE) for handling class imbalance. Inf Sci 505:32–64. https://doi.org/10.1016/j.ins.2019.07.070

Elreedy D, Atiya AF, Kamalov F (2024) A theoretical distribution analysis of synthetic minority oversampling technique (SMOTE) for imbalanced learning. Mach Learn 113(7):4903–4923. https://doi.org/10.1007/s10994-022-06296-4

Esposito C, Landrum GA, Schneider N, Stiefl N, Riniker S (2021) GHOST: adjusting the decision threshold to handle imbalanced data in machine learning. J Chem Inf Model 61(6):2623–2640. https://doi.org/10.1021/acs.jcim.1c00160

Fernández A, García S, Herrera F, Chawla NV (2018) SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. J Artif Intell Res 61:863–905. https://doi.org/10.1613/jair.1.11192

Galar M, Fernández A, Barrenechea E, Bustince H, Herrera F (2012) A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Trans Syst Man Cybern C 42(4):463–484. https://doi.org/10.1109/TSMCC.2011.2161285

García S, Herrera F (2008) An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J Mach Learn Res 9:2677–2694

García V, Mollineda RA, Sánchez JS (2008) On the k-NN performance in a challenging scenario of imbalance and overlapping. Pattern Anal Appl 11(3–4):269–280. https://doi.org/10.1007/s10044-007-0087-5

Gelman A (2006) Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). Bayesian Anal 1(3):515–534. https://doi.org/10.1214/06-BA117A

Gelman A, Carlin JB, Stern HS, Dunson DB, Vehtari A, Rubin DB (2013) Bayesian data analysis, 3rd edn. Chapman and Hall/CRC, Boca Raton. https://doi.org/10.1201/b16018

Grinsztajn L, Oyallon E, Varoquaux G (2022) Why do tree-based models still outperform deep learning on typical tabular data? In: Advances in Neural Information Processing Systems 35 (NeurIPS 2022), Datasets and Benchmarks Track. arXiv:2207.08815

Han H, Wang WY, Mao BH (2005) Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Advances in Intelligent Computing (ICIC 2005), Lecture Notes in Computer Science, vol 3644. Springer, Berlin, pp 878–887. https://doi.org/10.1007/11538059_91

Hassanat AB, Tarawneh AS, Altarawneh GA, Almuhaimeed A (2022) Stop oversampling for class imbalance learning: a critical review. arXiv:2202.03579

He H, Bai Y, Garcia EA, Li S (2008) ADASYN: adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). IEEE, pp 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969

He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284. https://doi.org/10.1109/TKDE.2008.239

Herriges JA, Kling CL (1999) Nonlinear income effects in random utility models. Rev Econ Stat 81(1):62–72. https://doi.org/10.1162/003465399767923827

Ho TK, Basu M (2002) Complexity measures of supervised classification problems. IEEE Trans Pattern Anal Mach Intell 24(3):289–300. https://doi.org/10.1109/34.990132

Hoffman MD, Gelman A (2014) The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. J Mach Learn Res 15(47):1593–1623

Hopkins M, Reeber E, Forman G, Suermondt J (1999) Spambase [dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C53G6X

Japkowicz N, Stephen S (2002) The class imbalance problem: a systematic study. Intell Data Anal 6(5):429–449. https://doi.org/10.3233/IDA-2002-6504

Jo T, Japkowicz N (2004) Class imbalances versus small disjuncts. ACM SIGKDD Explor Newsl 6(1):40–49. https://doi.org/10.1145/1007730.1007737

Kachan O, Savchenko A, Gusev G (2025) Simplicial SMOTE: oversampling solution to the imbalanced learning problem. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1. ACM, pp 625–635. https://doi.org/10.1145/3690624.3709268

Kelly M, Longjohn R, Nottingham K (n.d.) The UCI Machine Learning Repository. https://archive.ics.uci.edu

Khan J, Wei JS, Ringnér M, et al. (2001) Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nat Med 7(6):673–679. https://doi.org/10.1038/89044

Kong J, Rios T, Kowalczyk W, Menzel S, Bäck T (2020) On the performance of oversampling techniques for class imbalance problems. In: Advances in Knowledge Discovery and Data Mining (PAKDD 2020), Lecture Notes in Artificial Intelligence, vol 12085. Springer, Cham, pp 84–96. https://doi.org/10.1007/978-3-030-47436-2_7

Kovács G (2019a) An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. Appl Soft Comput 83:105662. https://doi.org/10.1016/j.asoc.2019.105662

Kovács G (2019b) Smote-variants: a python implementation of 85 minority oversampling techniques. Neurocomputing 366:352–354. https://doi.org/10.1016/j.neucom.2019.06.100

Koyejo O, Natarajan N, Ravikumar P, Dhillon IS (2014) Consistent binary classification with generalized performance metrics. In: Advances in Neural Information Processing Systems 27 (NIPS 2014)

Koziarski M, Krawczyk B, Woźniak M (2019) Radial-based oversampling for noisy imbalanced data classification. Neurocomputing 343:19–33. https://doi.org/10.1016/j.neucom.2018.04.089

Krawczyk B (2016) Learning from imbalanced data: open challenges and future directions. Prog Artif Intell 5(4):221–232. https://doi.org/10.1007/s13748-016-0094-0

Kruschke JK (2018) Rejecting or accepting parameter values in Bayesian estimation. Adv Methods Pract Psychol Sci 1(2):270–280. https://doi.org/10.1177/2515245918771304

Kruschke JK, Liddell TM (2018) The Bayesian new statistics: hypothesis testing, estimation, meta-analysis, and power analysis from a Bayesian perspective. Psychon Bull Rev 25(1):178–206. https://doi.org/10.3758/s13423-016-1221-4

Lemaître G, Nogueira F, Aridas CK (2017) Imbalanced-learn: a Python toolbox to tackle the curse of imbalanced datasets in machine learning. J Mach Learn Res 18(17):1–5

Lipton ZC, Elkan C, Narayanaswamy B (2014) Optimal thresholding of classifiers to maximize F1 measure. In: Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2014), Lecture Notes in Computer Science. Springer, Berlin, pp 225–239. https://doi.org/10.1007/978-3-662-44851-9_15

Liu XY, Wu J, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man Cybern B 39(2):539–550. https://doi.org/10.1109/TSMCB.2008.2007853

Liu Z, Li Z, Yang Z, Wei T, Kang J, Zhu Y, Hamann H, He J, Tong H (2025) CLIMB: class-imbalanced learning benchmark on tabular data. In: Advances in Neural Information Processing Systems 38 (NeurIPS 2025), Datasets and Benchmarks Track. arXiv:2505.17451

Lorena AC, Garcia LPF, Lehmann J, Souto MCP, Ho TK (2019) How complex is your classification problem? A survey on measuring classification complexity. ACM Comput Surv 52(5):1–34. https://doi.org/10.1145/3347711

López V, Fernández A, García S, Palade V, Herrera F (2013) An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics. Inf Sci 250:113–141. https://doi.org/10.1016/j.ins.2013.07.007

Luengo J, Fernández A, García S, Herrera F (2011) Addressing data complexity for imbalanced data sets: analysis of SMOTE-based oversampling and evolutionary undersampling. Soft Comput 15(10):1909–1936. https://doi.org/10.1007/s00500-010-0625-8

Ma L, Fan S (2017) CURE-SMOTE algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests. BMC Bioinformatics 18:169. https://doi.org/10.1186/s12859-017-1578-z

Maciejewski T, Stefanowski J (2011) Local neighbourhood extension of SMOTE for mining imbalanced data. In: 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM). IEEE, pp 104–111. https://doi.org/10.1109/CIDM.2011.5949434

Menardi G, Torelli N (2014) Training and assessing classification rules with imbalanced data. Data Min Knowl Disc 28:92–122. https://doi.org/10.1007/s10618-012-0295-5

Moniz N, Cerqueira V (2021) Automated imbalanced classification via meta-learning. Expert Syst Appl 178:115011. https://doi.org/10.1016/j.eswa.2021.115011

Musaev D (2026) When single-dataset conclusions fail: a 45-task study of threshold tuning and resampling for imbalanced classification. arXiv:2608.16147

Nadeau C, Bengio Y (2003) Inference for the generalization error. Mach Learn 52(3):239–281. https://doi.org/10.1023/A:1024068626366

Napierała K, Stefanowski J, Wilk S (2010) Learning from imbalanced data in presence of noisy and borderline examples. In: Rough Sets and Current Trends in Computing (RSCTC 2010), Lecture Notes in Artificial Intelligence, vol 6086. Springer, Berlin, pp 158–167. https://doi.org/10.1007/978-3-642-13529-3_18

Napierała K, Stefanowski J (2016) Types of minority class examples and their influence on learning classifiers from imbalanced data. J Intell Inf Syst 46(3):563–597. https://doi.org/10.1007/s10844-015-0368-1

Nguyen HM, Cooper EW, Kamei K (2011) Borderline over-sampling for imbalanced data classification. Int J Knowl Eng Soft Data Paradig 3(1):4–21. https://doi.org/10.1504/IJKESDP.2011.039875

Niculescu-Mizil A, Caruana R (2005) Predicting good probabilities with supervised learning. In: Proceedings of the 22nd International Conference on Machine Learning (ICML 2005). ACM, pp 625–632. https://doi.org/10.1145/1102351.1102430

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, et al. (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12(85):2825–2830

Phan D, Pradhan N, Jankowiak M (2019) Composable effects for flexible and accelerated probabilistic programming in NumPyro. arXiv:1912.11554

Prati RC, Batista GEAPA, Monard MC (2004) Class imbalances versus class overlapping: an analysis of a learning system behavior. In: MICAI 2004: Advances in Artificial Intelligence, Lecture Notes in Computer Science. Springer, Berlin, pp 312–321. https://doi.org/10.1007/978-3-540-24694-7_32

Prati RC, Batista GEAPA, Silva DF (2015) Class imbalance revisited: a new experimental setup to assess the performance of treatment methods. Knowl Inf Syst 45(1):247–270. https://doi.org/10.1007/s10115-014-0794-3

Provost F (2000) Machine learning from imbalanced data sets 101. In: Proceedings of the AAAI-2000 Workshop on Imbalanced Data Sets (invited paper). Technical report CeDER-PP-2000-02, Center for Digital Economy Research, NYU Stern School of Business

Quinlan R (1986) Thyroid disease [dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5D010

Ramentol E, Caballero Y, Bello R, Herrera F (2012) SMOTE-RSB*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory. Knowl Inf Syst 33(2):245–265. https://doi.org/10.1007/s10115-011-0465-6

Rice JR (1976) The algorithm selection problem. Adv Comput 15:65–118. https://doi.org/10.1016/S0065-2458(08)60520-3

Sağlam F, Cengiz MA (2025) MCSMOTE: a transition matrix-driven oversampling technique for class imbalance. Appl Soft Comput 185:113906. https://doi.org/10.1016/j.asoc.2025.113906

Saito T, Rehmsmeier M (2015) The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS ONE 10(3):e0118432. https://doi.org/10.1371/journal.pone.0118432

Sakho A, Malherbe E, Scornet E (2026) Do we need rebalancing strategies? A theoretical and empirical study around SMOTE and its variants. In: Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS 2026), PMLR 300. arXiv:2402.03819

Sáez JA, Luengo J, Stefanowski J, Herrera F (2015) SMOTE–IPF: addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. Inf Sci 291:184–203. https://doi.org/10.1016/j.ins.2014.08.051

Sáez JA, Krawczyk B, Woźniak M (2016) Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. Pattern Recognit 57:164–178. https://doi.org/10.1016/j.patcog.2016.03.012

Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2010) RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern A 40(1):185–197. https://doi.org/10.1109/TSMCA.2009.2029559

Sigillito V, Wing S, Hutton L, Baker K (1989a) Ionosphere [dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5W01B

Sigillito VG, Wing SP, Hutton LV, Baker KB (1989b) Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Tech Dig 10(3):262–266

Skryjomski P, Krawczyk B (2017) Influence of minority class instance types on SMOTE imbalanced data oversampling. In: Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications (LIDTA 2017), PMLR 74:7–21

Smith MR, Martinez T, Giraud-Carrier C (2014) An instance level analysis of data complexity. Mach Learn 95:225–256. https://doi.org/10.1007/s10994-013-5422-z

Sun Y, Wong AKC, Kamel MS (2009) Classification of imbalanced data: a review. Int J Pattern Recognit Artif Intell 23(4):687–719. https://doi.org/10.1142/S0218001409007326

Tomek I (1976) Two modifications of CNN. IEEE Trans Syst Man Cybern SMC-6(11):769–772. https://doi.org/10.1109/TSMC.1976.4309452

van den Goorbergh R, van Smeden M, Timmerman D, Van Calster B (2022) The harm of class imbalance corrections for risk prediction models: illustration and simulation using logistic regression. J Am Med Inform Assoc 29(9):1525–1534. https://doi.org/10.1093/jamia/ocac093

Van Hulse J, Khoshgoftaar TM, Napolitano A (2007) Experimental perspectives on learning from imbalanced data. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007). ACM, pp 935–942. https://doi.org/10.1145/1273496.1273614

Vehtari A, Gelman A, Simpson D, Carpenter B, Bürkner PC (2021) Rank-normalization, folding, and localization: an improved $\hat{R}$ for assessing convergence of MCMC (with discussion). Bayesian Anal 16(2):667–718. https://doi.org/10.1214/20-BA1221

Vuttipittayamongkol P, Elyan E, Petrovski A (2021) On the class overlap problem in imbalanced data classification. Knowl-Based Syst 212:106631. https://doi.org/10.1016/j.knosys.2020.106631

Weiss GM, Provost F (2003) Learning when training data are costly: the effect of class distribution on tree induction. J Artif Intell Res 19:315–354. https://doi.org/10.1613/jair.1199

Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern SMC-2(3):408–421. https://doi.org/10.1109/TSMC.1972.4309137

Wojciechowski S, Wilk S (2017) Difficulty factors and preprocessing in imbalanced data sets: an experimental study on artificial data. Found Comput Decis Sci 42(2):149–176. https://doi.org/10.1515/fcds-2017-0007

Zadrozny B, Elkan C (2002) Transforming classifier scores into accurate multiclass probability estimates. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp 694–699. https://doi.org/10.1145/775047.775151

Zhang H, Li M (2014) RWO-Sampling: a random walk over-sampling approach to imbalanced data classification. Inf Fusion 20:99–116. https://doi.org/10.1016/j.inffus.2013.12.003

Zou Q, Xie S, Lin Z, Wu M, Ju Y (2016) Finding the best classification threshold in imbalanced classification. Big Data Res 5:2–8. https://doi.org/10.1016/j.bdr.2015.12.001

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- OVSsupplementaryDMKD.docx