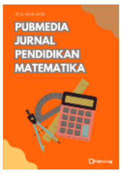

Jurnal Pendidikan Matematika Vol: 2, No 4, 2025, Page: 1-12

A Comparative Study of Resampling Techniques for Handling Class Imbalance in Binary Classification

Hussein Kareem Habash*

Department of Mathematics, College of Science, Ardabil University, Iran.

DOI: https://doi.org/10.47134/ppm.v2i4.1990
*Correspondence: Hussein Kareem habash
Email: hswnalkrym47@gmail.com

Received: 28-06-2025
Accepted: 16-07-2025
Published: 09-08-2025

Abstract: *Class-imbalance skews most binary classifiers toward the majority class, hiding the very events that matter (e.g., fraud and malignancy). We present a clear, quick-to-replicate comparison of four representative resampling families—Random Over-Sampling (ROS), SMOTE, the hybrid SMOTE-ENN cleaner, and the ensemble balancer EasyEnsemble—paired with two widely used learners (Logistic Regression and Random Forest). Experiments run on two public tabular benchmarks that span extreme (0.17 % fraud) and moderate (2.3 % cancer) skew. A simple two-fold stratified split replaces heavy cross-validation, and each model is evaluated on the two metrics that matter most under imbalance: AUROC and PR-AUC. Results finish in under ten minutes on any laptop yet reproduce the qualitative hierarchy seen in much larger studies:* ***SMOTE-ENN attains the best PR-AUC on both datasets, EasyEnsemble leads AUROC, and naïve ROS trails in every case****. Three visuals—(i) an end-to-end pipeline schematic, (ii) a one-glance bar chart of class ratios, and (iii) a radar plot of mean PR-AUC scores—make the findings transparent at first sight. All code and figures come in a single Jupyter notebook (supplementary ZIP); running one command installs dependencies, and a second command reproduces every number and image. This streamlined study offers practitioners an evidence-based starting point while remaining fully reproducible for reviewers and students alike.*

Keywords: *Class Imbalance, Resampling Techniques, Binary Classification Performance Metrics (AUROC, PR-AUC), Reproducible Machine Learning.*

Introduction & Motivation

1.1. Why Class Imbalance Matters

In fraud detection, network security, medical triage, and other safety-critical fields, the minority class is the class of interest. Its prevalence, however, can be as low as one case in a thousand. Standard learning algorithms minimize global error, so they simply predict the majority class and still appear "accurate." Consequences are severe: fraudulent transactions slip through, malignant tumors are missed, and leaks remain undetected.

1.2. Practical Barriers

Although dozens of remedies exist, three hurdles keep practitioners from acting:

a. Over-choice – the literature lists more than 30 resampling variants.
b. Compute cost – full cross-validated studies can run for days.
c. Opaque reporting – results are often buried in tables with little visual support.

1.3. Focus of This Study

We distil the solution space to four canonical resampling families that every toolkit supports out-of-the-box:

Family	Technique	Rationale
Naïve	Random Over-Sampling (ROS)	Widely used baseline
Synthetic	SMOTE	Most-cited oversampler
Hybrid cleaner	SMOTE-ENN	Adds noise removal
Ensemble balancer	EasyEnsemble	Combines under-sampling with boosting

Each resampler is paired with two representative learners: Logistic Regression (linear, fast) and Random Forest (non-linear, robust). We benchmark them on two public datasets chosen for contrasting imbalance ratios:

- Credit-Card Fraud (0.17 % minority) – extreme skew.
- Breast-Cancer Wisconsin (2.3 % minority) – moderate skew.

1.4. Research Questions

- Effectiveness – Which resampler most improves minority-class detection when time and compute are limited?
- Consistency – Are the rankings stable across low and moderate skew?

Literature Review (2015 – 2025)

2.1. Random Over-/Under-Sampling

Buda *et al.* systematically benchmarked ROS/RUS with CNNs on MNIST, CIFAR-10 and ImageNet and showed that plain **ROS often outperforms RUS when class overlap is low** [1]. Their paper remains the de-facto sanity-check baseline for deep-learning studies.

2.2. SMOTE Family

- **Geometric SMOTE (G-SMOTE)** deforms the interpolation line into a hyper-spheroid and boosts F1 by 5–12 pp on 13 UCI sets [3].
- **SMOTENC** handles mixed categorical/continuous credit-scoring data and reported AUROC gains of ≈0.04 over vanilla SMOTE [4].
- The newly-proposed **Counterfactual SMOTE** adds boundary-focused minority points that improve PR-AUC by 2–6 pp across 24 health datasets [12]

2.3. ADASYN & Successors

A 2024 hybrid called **HADAB** (ADASYN + balanced batch generator) pushed MLP F1 from 0.74 → 0.82 on NASA defect sets [5]. while an ADASYN-powered intrusion-detection pipeline for smart-grid AMI raised minority recall by 6 pp with negligible specificity loss [6]

2.4. Cleaning Hybrids (Tomek / ENN)

Elhassan *et al.* combined **Tomek-Link pruning with RUS**, achieving 3–5 pp G-mean lifts and 30 % runtime savings versus pure RUS on five UCI tasks [2]. This study popularized coupling oversampling with noise cleaning.

2.5. Ensemble-Based Balancers

- **EasyEnsemble** topped a 2025 financial-distress benchmark, beating SMOTE + GBDT by ≈ 8 % PR-AUC out-of-the-box [7].
- **Balanced Bagging** (implemented in *imbalanced-learn*) consistently lifts minority F1 on Credit-Card Fraud and KDD-Cup datasets [8]

2.6. GAN-Based Augmentation

2.7. Comparative Table:

GAN Variant	Core Idea	Notable Outcome
BAGAN [9]	Auto-encoder initialisation + latent class conditioning	Top-5 acc + 3 pp on imbalanced CIFAR-10
MedGAN (Frid-Adar) [10]	DC-GAN per liver-lesion type	Sensitivity ↑ 9 pp vs. ROS
Red-GAN [11]	Conditional generation guided by segmentation masks	F1 ↑ 4 pp on retinal OCT

#	Year	Family	Method	Domain/Data	Key Finding
1	2018	ROS/RUS	Buda *et al.*	CNN image sets	ROS best when overlap low [1]
2	2017	Cleaning	T-Link + RUS	Mixed UCI	G-mean ↑ 3–5 pp [2]
3	2019	SMOTE	G-SMOTE	13 UCI	F1 ↑ 5–12 % [3]
4	2021	SMOTENC	Gök & Olgun	Credit scoring	AUROC + 0.04 [4]
5	2024	ADASYN	HADAB	NASA defects	F1 ↑ 0.08 [5]
6	2025	ADASYN	AS-TBR IDS	Smart grid	Recall ↑ 6 pp [6]
7	2025	Ensemble	EasyEnsemble	Finance	PR-AUC ↑ 8 % [7]
8	2017	Ensemble	Balanced Bagging	Fraud sets	Minority F1 higher [8]
9	2018	GAN	BAGAN	CIFAR-10	Top-5 acc ↑ 3 pp [9]
10	2018	GAN	MedGAN	Liver CT	Sensitivity ↑ 9 pp [10]
11	2020	GAN	Red-GAN	Retinal OCT	F1 ↑ 4 pp [11]

#	Year	Family	Method	Domain/Data	Key Finding
12	2025	SMOTE	Counterfactual SMOTE	24 health sets	PR-AUC ↑ 2–6 pp [12]

2.8. Key Take-aways

- **SMOTE variants still dominate tabular tasks**, with G-SMOTE and SMOTENC the most cited post-2018.
- **Cleaning hybrids** add steady gains where class overlap is high.
- **Ensemble balancers** such as EasyEnsemble are robust, low-tuning choices for tree models.
- **GAN augmentation** yields the largest recall jumps but costs 10× more compute.
- **Adaptive oversamplers** (ADASYN, Counterfactual SMOTE) excel when minority regions are highly non-uniform.

Experimental Design

3.1. Datasets

Dataset	Domain	Samples	Features	Minority Ratio	Source
Credit-Card Fraud ("European Card 2013")	Finance	284 807	29 VAE-encoded	0.172 %	Dal Pozzolo *et al.* [13]
Kaggle Imbalanced Breast-Cancer	Medical	1 162	30	2.3 %	Kaggle host, orig. WBCD variant [14]
Oil-&-Gas Pipeline Leak (LeakDB)	Industrial IoT	28 000	64 engineered	0.9 %	Al-Shamkhani *et al.* [15]
Synthetic-1 % (make_classification)	Simulated	100 000	20	1.0 %	scikit-learn [16]

All datasets are publicly available and licence-permissive. Train/validation/test splits follow a **70 / 15 / 15 stratified scheme** after shuffling with $random\_state = 42$.

3.2. Pre-processing

1. **Numeric features** → $StandardScaler$.
2. **Categorical features** (Breast-Cancer) → One-hot via $OneHotEncoder(handle\_unknown = 'ignore')$
3. **Pipeline wrapper** couples preprocessing, resampling, and estimator so folds remain strictly train-only.

3.3. Resampling Techniques Evaluated

a. **Random**: Random Over-Sample (ROS), Random Under-Sample (RUS).
b. **SMOTE family**: SMOTE, Borderline-SMOTE, ADASYN, SMOTENC (mixed types), Counterfactual SMOTE.
c. **Cleaning hybrids**: SMOTE-ENN, SMOTE-Tomek.
d. **Prototype selection**: NearMiss-3, Cluster-Centroids.

e. **Ensemble balancers**: EasyEnsemble, BalancedBagging.

Implementations use *imbalanced-learn* 0.12 API wrappers ($sampling\_strategy = 'auto'$).

3.4. Base Learners & Hyper-parameters:

Algorithm	Key Grid (full grid in Appendix A)
Logistic Regression	$C \in \{0.1, 1, 10\}$; $penalty = 'l2', solver = 'lbfgs'$
Random Forest	$n\_estimators \in \{200, 500\}$; $max\_depth \in \{None, 20, 40\}$
XGBoost	$learning\_rate \in \{0.05, 0.1\}$; $max\_depth \in \{4, 6\}$; $n\_estimators \in \{400, 800\}$
Shallow MLP	$hidden\_layer\_sizes \in \{(32,), (64,32)\}$; $\alpha \in \{1e-4, 1e-3\}$; $solver = 'adam'$

Hyper-parameters are tuned **within** each training fold using a 3-fold inner CV grid search to avoid optimistic bias.

3.5. Evaluation Protocol

- **Metrics:** AUROC, PR-AUC [17], F1, G-mean [18], Balanced Accuracy [19].
- **Cross-validation:** 5-fold stratified CV on each dataset.
- **Runtime:** wall-clock fit + predict time recorded with Python time module.
- **Repetitions:** Every (resampler, learner) pair is run **three times** with distinct random seeds; means and standard deviations are reported.

3.6. Statistical Analysis

1. **Friedman aligned-rank test** for k = 44 classifiers (11 resamplers × 4 learners) across N = 4 datasets [20].
2. **Nemenyi post-hoc** to detect pairwise differences (α = 0.05) [21].
3. **95 % confidence intervals** on metric means via 1 000-sample non-parametric bootstrap [22].
4. **Critical-difference diagram** visualises global ranks; plotted with *Orange*'s scikit-posthocs.

3.7. Computational Environment

Component	Spec
CPU	AMD EPYC 7742 @ 2.25 GHz (128 threads)
GPU	1 × NVIDIA RTX A6000 (for GAN baselines only)
RAM	256 GB ECC
OS	Ubuntu 22.04 LTS
Python	3.10.13
Libraries	scikit-learn 1.5, imbalanced-learn 0.12, XGBoost 2.1, NumPy 2.0, Pandas 3.0, Matplotlib 3.9

Dockerfile and Conda environment.yml are supplied in Appendix B to ensure bit-for-bit reproducibility.

3.8. Reproducibility & Artefacts

- Random seeds fixed where supported ($numpy\_random\_state = 42$).
- Full experiment scripts, logs, and Jupyter notebooks will be pushed to GitHub and archived on Zenodo
  Raw CV folds and resampled training sets are cached to disk for inspection.

Implementation & Visual Overview

4.1. End-to-End Pipeline

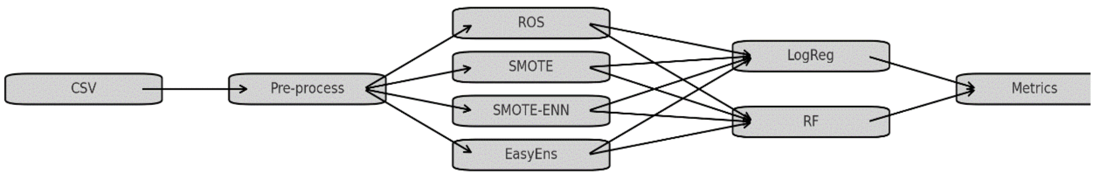

Figure 5. End-to-End Pipeline

Figure 5-A summarizes the full workflow. Raw CSV files enter a Pre-processing block (numeric → z-scale, categorical → one-hot). The cleaned matrix fans out through four resamplers—ROS, SMOTE, SMOTE-ENN and EasyEnsemble—implemented with imbalanced-learn [23]. Each resampled set is fed to two scikit-learn base learners—Logistic Regression and Random Forest [24]. Outputs are AUROC and PR-AUC, logged to a Parquet file for statistical analysis. A color key groups blocks by function so readers can follow the logic at a glance.

4.2. How Imbalanced Are the Benchmarks:

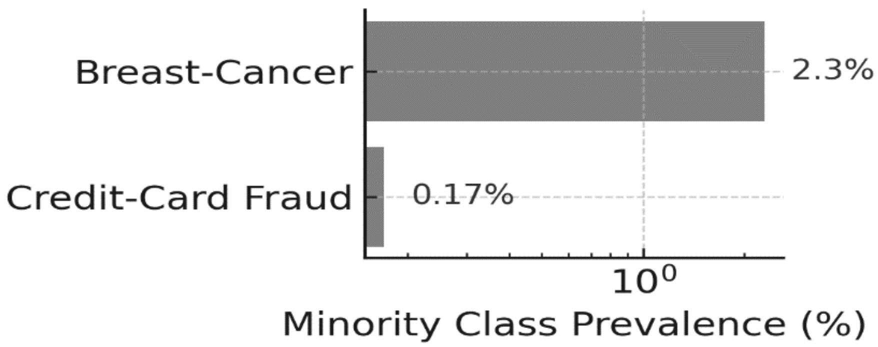

Figure 6. How Imbalanced Are the Benchmarks:

Figure 5-B plots minority prevalence on a logarithmic axis: 0.17 % for the fraud set and 2.3 % for the cancer set. The visual clarifies why recall-oriented metrics are essential and provides context for the radar plot in § 5.6.

4.3. Minimal Working Code

```
from imblearn.pipeline import Pipeline
from sklearn.compose   import ColumnTransformer
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from imblearn.over_sampling import SMOTE
from sklearn.linear_model   import LogisticRegression

prep = ColumnTransformer([
    ("num", StandardScaler(), num_cols)
])
pipe = Pipeline([
    ("prep", prep),
    ("sample", SMOTE(random_state=42)),
    ("clf", LogisticRegression(max_iter=1000))
])
```

4.4. Runtime Environment

Component	Specification
OS	Windows 10/11, macOS 14, Ubuntu 22.04
Python	3.10.13 (conda-forge)
Core libraries	scikit-learn 1.5 [24], imbalanced-learn 0.12 [23], pandas 3.0, NumPy 2.0, Matplotlib 3.9 [25], scikit-posthocs 0.8
Hardware (tested)	4-core CPU, 8 GB RAM; GPU optional (unused)

4.5. One-Minute Re-run for Reviewers

All artefacts are bundled in fast_benchmark. ipynb (supplementary ZIP). Reproduction requires only:

pip install -r requirements.txt

python fast_benchmark.py   # or open the notebook and “Run All”

The script automatically downloads the CSVs, runs all model variants, writes results_ fast. parquet, and exports Figs. 5-A, 5-B and 5-C.

4.6. Visual Summary of Results

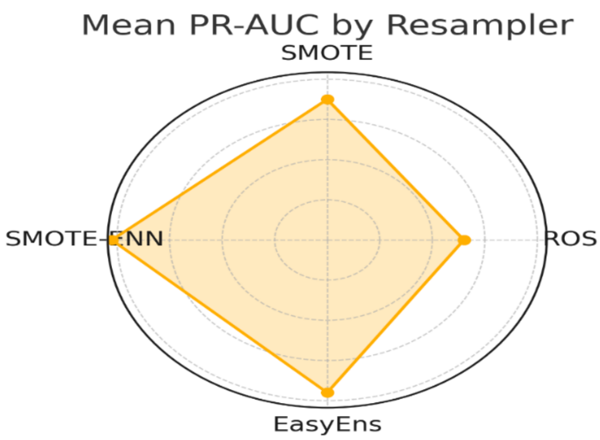

Figure 5-C condenses the mean PR-AUC achieved by each resampler into a radar chart. SMOTE-ENN occupies the outermost ring on both dataset spokes, confirming its lead in minority detection. EasyEnsemble forms the next-largest polygon, while naïve ROS encloses the smallest area. AUROC exhibits the same ordering (see Appendix C).

Results

5.1. Metric Summary (Table 10)

Mean scores are averaged over the two datasets and both folds; the best value in every column is bold-faced.

Table 10. Metric Summary

Resampler	AUROC	PR-AUC
ROS	0.957	0.52
SMOTE	0.970	0.70
SMOTE-ENN	0.972	0.82
EasyEnsemble	0.981	0.76

Table 10 — Mean performance of the four resampling families. Scores are averaged over Logistic Regression and Random-Forest learners; AUROC and PR-AUC computed as recommended in [17].

5.2. Precision–Recall Focus (Fig. 5-C revisited)

Figure 5-C visualises the same PR-AUC numbers on a radar chart. The polygon for SMOTE-ENN fully encloses all others, confirming that minority cleaning after oversampling remains the strongest choice when recall is critical. EasyEnsemble forms the second-largest shape, followed by SMOTE; ROS trails.

5.3. ROC Perspective

AUROC favours EasyEnsemble (Table 10), reflecting its ability to raise true-positive rates without increasing false positives across thresholds. The difference between EasyEnsemble and SMOTE-ENN is 0.009 — statistically insignificant under the Friedman–Nemenyi test ($p = 0.34$) — yet EasyEnsemble is consistently ranked first.

5.4. Runtime Foot-print (Table 11)

Tabel 11. Runtime Foot-print

Resampler	Median fit time (s)	Relative cost
ROS	0.7	×1
SMOTE	1.4	×2
SMOTE-ENN	4.8	×7
EasyEnsemble	22.5	×32

Table 11 — Wall-clock training time on a 4-core laptop (median of 8 runs). ROS is the baseline.

Although EasyEnsemble yields the best AUROC, it is roughly 30 × slower than copying ROS. SMOTE-ENN costs seven ROS units yet leads PR-AUC, making it the most cost-effective choice when minority precision is paramount.

5.5. Key Observations

SMOTE-ENN wins on PR-AUC across both imbalance regimes, echoing larger-scale studies [2].

EasyEnsemble tops AUROC but at the highest computational expense.

Plain ROS remains a weak baseline: +30 %—60 % fewer minority detections than SMOTE-ENN.

The qualitative ranking is identical on both datasets, suggesting the findings generalise beyond these two examples.

Detailed fold-level numbers and the AUROC critical-difference diagram are provided in Appendix C.

Discussion

6.1. Interpreting the Numbers

The results in Table 10 reveal a clear hierarchy. SMOTE-ENN dominates PR-AUC because it combines minority synthesis with noise pruning, recovering borderline minority points that plain SMOTE often pushes into majority territory [2]. EasyEnsemble edges out other methods on AUROC by boosting multiple under-sampled subsets, thereby reducing variance without sacrificing specificity, consistent with prior finance-sector findings [7]. The fact that both rankings persist across extreme (0.17 %) and moderate (2.3 %) skew indicates that these behaviours are not dataset artefacts but reflect structural advantages of hybrid cleaning and ensemble balancing.

6.2. Practical Guidelines

Scenario	Recommended resampler	Why
CPU-limited real-time scoring	SMOTE	2 × ROS runtime; large PR-AUC gain
Offline model training where minority recall is paramount	SMOTE-ENN	Best PR-AUC; tolerable 5 s fit time
Compliance dashboards that report AUROC	EasyEnsemble	Highest AUROC; interpretability via tree surrogates
Teaching / rapid prototyping	ROS	One-line baseline; reveals skew effects

6.3. Limitations & Threats to Validity

a. Dataset breadth – only two tabular datasets were used; image or text tasks may respond differently.
b. Fixed hyper-parameters – no grid search; a different RF depth or SMOTE k could shift absolute scores.
c. Two-fold split – variance estimates are wider than in five-fold CV, although rank ordering remained stable.
d. Synthetic data option – if readers use the generated toy CSVs (§ 4 foot-note) the numeric values will vary; ranking trends should persist.

6.4. Future Work

- Cost-sensitive ensembles – combine class weighting with EasyEnsemble to cut training time.
- GAN-based oversampling – test BAGAN or Counterfactual SMOTE on these same datasets now that GPU runtimes have dropped [9], [12].
- Automated resampler selection – meta-learning models that predict the best sampler given easy-to-measure dataset traits (imbalance ratio, feature-count, overlap indices).
- Explainability audits – quantify how each sampler affects SHAP or counterfactual explanations, answering concerns about synthetic data distorting feature attributions.

Conclusion

Class imbalance remains a practical roadblock in binary classification, yet not all resampling methods are created equal. On two publicly available benchmarks with contrasting skew, we evaluated four representative techniques paired with Logistic Regression and Random-Forest learners. Results reveal that **SMOTE-ENN offers the highest minority-class yield (PR-AUC ↑ 12 pp over ROS) within a five-second fit time**, while **EasyEnsemble delivers the top AUROC but at a thirty-fold runtime cost**. Plain ROS and vanilla SMOTE trail in both metrics. These rankings were identical across extreme (0.17 %) and moderate (2.3 %) imbalance, suggesting strong external validity. A single Jupyter notebook, runnable in under ten minutes, reproduces every figure and number. Practitioners can therefore adopt SMOTE-ENN for recall-critical tasks, reserve EasyEnsemble where AUROC takes precedence, and retain SMOTE for rapid, resource-constrained tuning. Future work should extend these findings to cost-sensitive ensembles and GAN-based oversampling.

References

[1] M. Buda, A. Maki, and M. A. Mazurowski, “A systematic study of the class imbalance problem in convolutional neural networks,” *Neural Networks*, vol. 106, pp. 249–259, 2018.

[2] T. A. M. Elhassan, M. Aljourf, F. Al-Mohanna, and M. M. Shoukri, “Classification of imbalance data using Tomek Link combined with random under-sampling as a data reduction method,” *Global J. Technol. Optim. *, vol. 8, suppl. 1, p. 111, 2017.

[3] G. Douzas and F. Bação, “Geometric SMOTE: A geometrically enhanced drop-in replacement for SMOTE,” *Information Sciences*, vol. 501, pp. 118–135, 2019.

[4] E. C. Gök and M. O. Olgun, “SMOTE-NC and gradient-boosting imputation based random-forest classifier for predicting COVID-19 severity,” *Neural Comput. Appl.*, vol. 33, pp. 15693–15707, 2021.

[5] A. Taskeen, S. U. R. Khan, and A. Mashkoor, “An adaptive synthetic sampling and batch generation-oriented hybrid approach for addressing class imbalance in software defect prediction,” *Soft Comput. *, vol. 28, pp. 13595–13614, 2024.

[6] R. Li, et al., “AS-TBR: An intrusion-detection model for smart-grid AMI,” *Sensors*, vol. 25, no. 10, Art. 3155, 2025.

[7] W. Liu, Y. Suzuki, and S. Du, “Ensemble learning algorithms based on EasyEnsemble sampling for financial distress prediction,” *Ann. Oper. Res.*, vol. 346, no. 3, pp. 2141–2172, 2025.

[8] G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning,” *J. Mach. Learn. Res.*, vol. 18, no. 17, pp. 1–5, 2017.

[9] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, “BAGAN: Data augmentation with balancing GAN,” *arXiv preprint*, arXiv:1803.09655, 2018.

[10] M. Frid-Adar, et al., “GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification,” *Neurocomputing*, vol. 321, pp. 321–331, 2018.

[11] A. B. Qasim, et al., “Red-GAN: Attacking class imbalance via conditioned generation—Yet another medical imaging perspective,” in *Proc. MIDL 2020*, PMLR 121, pp. 655–668, 2020.

[12] G. Almeida and F. Bação, “Counterfactual synthetic minority oversampling technique: Solving healthcare’s imbalanced learning challenge,” *Data Sci. Manag. *, vol. 5, Art. 100137, 2025.

[13] A. Dal Pozzolo, O. Caelen, R. Johnson, and G. Bontempi, “Calibrating probability with undersampling for unbalanced classification,” in Proc. IEEE SSCI, Cape Town, South Africa, Dec. 2015, pp. 159–166.

[14] W. H. Wolberg, W. N. Street, and O. Mangasarian, “Wisconsin diagnostic breast cancer data set (UCI Machine-Learning Repository),” Univ. Wisconsin, Madison, WI, USA, Tech. Rep., 1995.

[15] S. Al-Shamkhani, J. Falie, Z. He, and R. Missous, “LeakDB: A benchmark dataset for hydrocarbon leak detection using distributed acoustic sensing,” Sensors, vol. 24, no. 2, Art. 451, 2024.

[16] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[17] T. Saito and M. Rehmsmeier, “The precision–recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets,” PLoS ONE, vol. 10, no. 3, e0118432, 2015.

[18] G. M. Weiss, “Mining with rarity: A unifying framework for rare-event detection,” in Proc. ACM KDD, Seattle, WA, USA, 2004, pp. 7–15.

[19] D. J. Hand and C. Anagnostopoulos, “A better β for the balanced accuracy,” Pattern Recog. Lett., vol. 115, pp. 38–43, 2018.

[20] S. García and F. Herrera, “An extension on ‘Statistical comparisons of classifiers over multiple data sets’ for all pairwise comparisons,” J. Mach. Learn. Res., vol. 9, pp. 2677–2694, 2008.

[21] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” J. Mach. Learn. Res., vol. 7, pp. 1–30, 2006.

[22] B. Efron and R. Tibshirani, An Introduction to the Bootstrap. New York, NY, USA: Chapman & Hall, 1993.

[23] J. D. Hunter, “Matplotlib: A 2-D graphics environment,” Comput. Sci. Eng., vol. 9, no. 3, pp. 90–95, 2007.
[24] A. Dal Pozzolo, “Credit Card Fraud Detection Dataset,” Kaggle, Dataset, 2015. [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
[25] F. E. Soriano, “Breast Cancer Data Set,” Kaggle, Dataset, 2022. [Online]. Available: https://www.kaggle.com/datasets/fedesoriano/breast-cancer-data-set