International Journal of Computer Techniques–IJCT Volume 13 Issue 3, May-June - 2026

Open Access and Peer Review Journal ISSN 2394-2231 https://ijctjournal.org/

RESEARCH ARTICLE OPEN ACCESS

Empirical Evidence of Posterior Probability Bias Under SMOTE Oversampling in Imbalanced Credit Card Fraud Detection: A Comparative Study of HistGradient Boosting and Logistic Regression

Anthony A. Ogunbor

Computer Science, University of Applied Sciences, Germany

Email: aikhomu-anthony.ogunbor@iu-study.org

----------**********----------

Abstract:

Detecting credit card fraud is a classic imbalanced binary classification problem in which fraudulent transactions belong to the minority class. A typical example, is the Kaggle cardholder's dataset where fraudulent transactions are only 0.173%. This study is directly inspired by earlier work by Dal Pozzolo et al, showing that under-sampling distorts posterior class probabilities, while preserving ranking order, rendering the default classification threshold suboptimal and inflating false positive rates. This paper extends that finding to over-sampling by applying the Synthetic Minority Oversampling Technique (SMOTE) as the sole imbalance intervention to Histogram-based Gradient Boosting (HGB) and Logistic Regression (LR) on the same 284,807-transaction Kaggle Credit Card Fraud dataset used in a previous study. When SMOTE was applied as the only imbalance intervention, LR's false alarms increased by a factor of 165, from 35 to 5,775 across the five cross-validation folds used in the experiment, while ROC-AUC remained almost unchanged at 0.9806. This pattern, i.e., massive false alarm inflation with stable ranking, is the operational fingerprint of posterior probability bias as described in Dal Pozzolo et al. On the other hand, the other classifier in the experiment - HGB, responded differently, with false alarms decreasing and F1 improving by 28.3%, confirming that SMOTE's effect on probability calibration is model-dependent. On the test set, HGB achieved F1 of 0.8342 with 18 false alarms, compared to LR's F1 of 0.1131 with 1,415 false alarms. These results constitute direct empirical evidence that SMOTE over-sampling induces posterior probability bias that is consistent with, and therefore extends earlier established theoretical framework from under-sampling to an over-sampling context

Keywords - **Credit card fraud detection, SMOTE, Logistic Regression, Histogram-based gradient boosting, Posterior probability bias, Class imbalance, Over-sampling, Probability calibration, False positive inflation**

----------**********----------

I. INTRODUCTION

The foundational contribution of Dal Pozzolo, Caelen, Johnson, and Bontempi [1] is of central relevance to this work. Using the same Kaggle Credit Card Fraud Detection dataset employed in this study, the authors demonstrated analytically and through experiments that under-sampling (a popular technique for addressing class imbalance by removing some majority class instances) distorts the posterior probabilities produced by a classifier. Specifically, although the ranking order of predictions (and consequently ROC-AUC) is preserved under under-sampling, the calibration of predicted probability values is found not to be preserved. This makes the default classification threshold of 0.5 suboptimal. They proposed a mathematical correction using Bayes Minimum Risk theory to restore calibrated probabilities after under-sampling. This study extends this inquiry to over-sampling. The research question that arises is: Does SMOTE over-sampling (like under-sampling demonstrated in [1]) introduce an analogous posterior probability bias, and if so, is it

empirically detectable through classifier behaviour observed via experimental isolation on the same dataset?

This research evaluates the performance of two classifiers - HistGradient Boosting and Logistic Regression. They have been selected because they represent almost opposite ends of the model complexity spectrum. While Logistic Regression is simple, linear, and computationally inexpensive and requiring minimal tuning of requirements, HistGradient Boosting is a non-linear ensemble method that requires careful hyperparameter optimization in order to reach optimal performance. This intentional contrast (in the two classifiers chosen) permits the study to isolate the effect of model expressiveness as a performance determinant, without considering the effects of preprocessing and optimization. Three research objectives guide this study: (1) to observe and establish the specific behaviour of both classifiers on a severely imbalanced dataset across multiple experimental stages; (2) to quantify the effect of SMOTE preprocessing on each classifier's performance; and (3) to determine whether observed classifier bahaviour under SMOTE over-sampling is consistent with the probability distortion effects reported in [1] for under-sampling.

The proliferation of digital payment systems has introduced a corresponding rise in financial fraud, and imposed substantial costs on financial institutions and consumers globally. Classification in the Machine Learning (ML) domain is the process of assigning data instances to predefined categories based on learned patterns and has emerged as the primary technical paradigm for automated fraud detection. As a result, most fraud detection problems are treated as classification problems and solutions implemented via classification models. Algorithms including Support Vector Machines (SVM), Random Forests, Logistic Regression, Decision Trees, Neural Networks, and HistGradient Boosting have been widely applied to this task with varied effectiveness [2].

Class imbalance is a fundamental challenge in fraud detection and other outlier-detection domains, where the class of interest, i.e., the positive (minority) class, is always severely under-represented. Formally, class imbalance is defined as the difficulty faced by a trained ML model in effectively learning the patterns of the minority class mn when its frequency of occurrence is negligibly small relative to the majority class mj (i.e., mn << mj). In the benchmark dataset used in this study, this ratio is approximately 577:1, that is, one fraudulent transaction for every 577 legitimate transactions. This challenge has been widely recognized in fraud detection literature such as in [1] – [5]. In practice, this challenge results in the Machine Learning (ML) model becoming biased towards the majority class and leading to misleading high accuracy predictions of the majority class and poor prediction for the minority class, the class of interest.

The remainder of this paper is organized as follows: the materials & method is presented first, followed by the results and discussions, and finally the conclusion where directions for future work is stated.

II. LITERATURE REVIEW

The class imbalance problem has been extensively studied across multiple application domains. Standard ML algorithms that maximize overall accuracy tend to classify all observations as majority class instances when training data is highly skewed, resulting in poor recall on the minority class [1]. Bhattacharyya et al. [3] conducted a comparative study of supervised and unsupervised methods for credit card fraud detection, they found out that the supervised methods generally outperform unsupervised approaches when training data is properly constructed. Sopiyan et al. [4] empirically demonstrated that ensemble models, including Random Forest and HistGradient Boosting, consistently outperform Logistic Regression on the fraud detection task, attributing this to the ensemble's ability to model non-linear decision boundaries in Principal Component Analysis (PCA) - transformed feature spaces. Ali et al. [6] provided a systematic review of data preprocessing methods for class imbalance, identifying over-sampling, under-sampling, and cost-sensitive learning as the three principal intervention categories.

A. Re-sampling Strategies and Probability Calibration

The two dominant resampling strategies for class imbalance are under-sampling (which involves reducing the majority class), and over-sampling (which involves increasing the minority class with synthetic data). Under-sampling is simple and speeds up training, but with the risk that there is loss of information relating to the majority class [1]. SMOTE (Synthetic Minority Over-sampling Technique), introduced by Chawla et al. [7], generates synthetic minority class samples by interpolating between existing instances in feature space, avoiding simple duplication and reducing overfitting risk compared to naive oversampling.

Dal Pozzolo et al. [1] made the theoretically most significant contribution directly relevant to this study. They demonstrated that under-sampling modifies the prior class probabilities in the training set, causing a systematic upward shift in the posterior fraud probabilities produced by a classifier that is trained on the resampled data. Interestingly, this bias introduced does not affect the ranking of predictions as ROC-AUC (Area Under the Receiver Operating Characteristic Curve, an evaluation metric) is preserved. However, it renders the default threshold of 0.5 suboptimal, producing inflated false positive rates at deployment. They derived a correction formula using Bayes Minimum Risk theory and validated it on the Kaggle Credit Card Fraud dataset. This study demonstrates that SMOTE over-sampling produces the same class of distortion on the same dataset, but from an opposite resampling activity. Where under-sampling removes majority class instances to artificially balance the

training set, SMOTE adds synthetic minority class instances. Both interventions (under-sampling and over-sampling), shift the effective training class prior away from the true population prior of 0.173% fraud, and both (as this study shows), produce the same observable effects: which is, inflated false alarms with stable ROC-AUC.

B. *Gradient Boosting Methods for Imbalanced Classification*

Osborne J.W [8] discusses how log transformations reduce dataset skew and help stabilize variance. While this is discussed later under *Feature Engineering*, it is worth mentioning here because it can affect how Gradient Boosting methods behave on skewed datasets. Gradient Boosting (GB) is the family of ensemble methods that sequentially fits decision trees on the residual errors of previous trees and has consistently demonstrated strong performance on tabular, imbalanced datasets. The foundational framework for GB was formalized by Friedman [9]. LightGBM introduced by Ke et al. [10] and developed by Microsoft, employs a histogram-based approximation algorithm for split-finding that dramatically reduces computational cost on large datasets. The scikit-learn HistGradientBoostingClassifier used in this study implements this histogram-based approach. It is a computationally optimized variant of standard Gradient Boosting that reduces split evaluation from O(n) to O(255) per feature per node, making it hyperparameter search controllable on datasets of the scale employed here (284,807 rows) on consumer hardware [11]. Velarde et al. [12] systematically evaluated tree boosting methods across datasets of varying sizes and class distributions, and found that hyperparameter optimization improves detection performance but should be applied carefully, and that oversampling does not consistently improve results across all classifiers. This finding in [12] is empirically tested in this study. Hajek, Abedin, and Sivarajah [13] demonstrated XGBoost-based frameworks achieves strong precision-recall trade-offs in mobile payment fraud detection. This reinforces the suitability of gradient boosting methods for this domain.

C. *Hyperparameter Optimization*

Hastie, Tibshirani, and Friedman [14], in their book, The elements of Statistical Learning, established the theoretical basis for regularized model selection and it would later underpin hyperparameter optimization strategies such as Exhaustive Grid Search and Randomized Search. The work of Bergstra and Bengio [15] demonstrated empirically and in theory, that Randomized Search is more efficient than Exhaustive Grid Search for high-dimensional hyperparameter spaces. This is because many parameters have low sensitivity and random sampling from continuous distributions explores the space more thoroughly than a fixed discrete grid. This study directly compares both methods on both classifiers, providing empirical confirmation of their argument regarding a real-world imbalanced dataset used in the experiment.

III. DESIGN METHODOLOGY

A. *Experimental Setup*

All experiments in this study were implemented and executed on a consumer-grade Intel Core i7 personal laptop using Python 3.9, scikit-learn, imbalanced-learn, scipy, pandas, numpy, and tabulate. The full experimental pipeline completed successfully when executed as a standalone Python script (.py file using the PyCharm IDE), producing all results in approximately 75 minutes. Execution within the Jupyter Notebook environment encountered intermittent MemoryError exceptions during the parallel hyperparameter search phases (n_jobs=-1), attributable to the additional memory overhead of the Jupyter kernel on a machine where available RAM was a constraining factor. This is a hardware and environment constraint rather than algorithmic failure and does not affect the validity of the results, which were obtained from the standalone script execution as already explained above.

B. *Methodology*

The experimental methodology consists of six sequential stages implemented in Python using scikit-learn [11] , [16] and imbalanced-learn. All preprocessing steps are enclosed within pipeline objects as is the standard amongst practitioners in order to prevent data leakage. This ensures that scalers are fitted only on training data and that SMOTE synthetic samples are generated only within training folds, completely isolated from the validation/test data.

C. *Dataset and Data Partitioning*

The Kaggle Credit Card Fraud Detection dataset [17] was manually downloaded from the official Kaggle repository at https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud and saved on a standalone PC used for the experiment. It comprises 284,807 transactions of European cardholders from September 2013. Features V1-V28 are the result of a PCA transformation applied to protect cardholder privacy; The fraud Amount and Time are the only raw features. The target variable Class is binary: That is, 0 (for legitimate, 284,315 instances) and 1 (for fraudulent, 492 instances), yielding an imbalance ratio of approximately 577:1. The dataset contains no missing values.

In the experiment, the dataset is divided into training (80%) and test (20%) partitions using stratified random sampling, which preserves the class ratio (i.e., 577 non-fraudulent:1fraudulent transaction) in both splits. The test set is kept separate from the training set until final evaluation. As a result, it was never used during training or hyperparameter tuning. See Table I below for a summary of the partition.

TABLE I
DATASET PARTITIONING (80 – 20 STRATIFIED SPLIT)

Partition	Total Rows	Fraud Cases	Legitimate	Purpose
Training Set (80%)	227,845	394	227,451	Model training & tuning
Test Set (20%)	56,962	98	56,864	Final evaluation only
Full Dataset	284,807	492	284,315	—

D. Feature Engineering

Two features are engineered from the raw data, Amount and Time. The Amount feature is log-transformed using log(1 + x), following established practice for right-skewed non-negative financial transaction data [8]. The motivation for this transformation arises from the approximately log-normal characteristics commonly observed in payment transaction amounts, where most transactions are relatively small while a small number of very large values (transactions) create extreme outliers. These outliers disproportionately influence gradient updates in Logistic Regression and cause inefficient histogram bin allocation in HistGradient Boosting. The log-transformation is adopted as the theoretically motivated standard for this distribution type and is consistent with prior work on datasets with class imbalance. The Time feature, recorded as elapsed seconds from the first transaction, is converted to hour-of-day by computing modulo 86,400 and dividing by 3,600, capturing potential diurnal fraud patterns. The original Time column is removed after this transformation. The feature set that emerges contains 31 features.

E. Pipeline Construction and Cross-Validation

Two machine learning pipelines are constructed using ImbPipeline from the imbalanced-learn library. This class object correctly restricts SMOTE oversampling to only the training folds. This is a critical design choice as it prevents synthetic samples from leaking into the validation (i.e., test) data [11]. A 5-fold Stratified K-Fold cross-validator (shuffle = True, random_state = 42) is used throughout. With this, each fold preserves the 0.173% fraud rate and ensures that results are fully reproducible independent of this study.

F. Critical Design Decision

During the initial trials in this study, the algorithm-level weighting method, the class_ weight = “Balanced”, was applied during the training for both classifiers. It was however subsequently removed owing to the fact that it alters the model’s learning dynamics and therefore has potential to interfere with the results and preclude the attribution of model behavior to SMOTE alone.

G. Logistic Regression Pipeline

RobustScaler is employed. (It is chosen for its resistance to the outliers present in the Amount feature, using median and IQR rather than mean and standard deviation) → SMOTE oversampling (k_neighbors = 5) → Logistic Regression (solver='lbfgs', max_iter = 1000). The lbfgs solver is selected for its efficiency on medium-to-large datasets.

H. HistGradient Boosting Pipeline

SMOTE oversampling à HistGradientBoostingClassifier. HistGradientBoostingClassifier (early_stopping=True, n_iter_no_change=10). Early stopping prevents open-ended fitting and reduces computational cost. HistGradientBoostingClassifier was selected over the standard GradientBoostingClassifier because histogram-based gradient boosting is more computationally efficient in time and memory, which makes hyperparameter tuning on the full dataset feasible on consumer hardware. Considering the size of dataset (284,807 rows) in this study, it became a practical necessity [8] , [11].

I. Evaluation Metrics

Five metrics are reported: (i) Precision (TP / (TP + FP)), (ii) Recall (TP / (TP + FN)), (iii) F1 (harmonic mean of Precision and Recall), (iv) ROC-AUC, and (v) Average Precision (area under the Precision-Recall curve). Accuracy is deliberately excluded. This is because of the nature of the problem space, i.e., highly imbalanced (skewed) dataset. F1 is used as the optimization criterion in all hyperparameter search procedures. Confusion matrix counts (TP, FP, FN, TN), are arrived at by summation across all five CV folds and juxtaposed with percentage metrics to make the operational fraud detection reality apparent.

J. Hyperparameter Optimization

Here the two classifiers involved are optimized using two strategies (Randomized Search and Exhaustive Grid Search). These strategies are applied and compared in parallel. Randomized Search (20 iterations, log-uniform distributions for continuous parameters) and Exhaustive Grid Search (all combinations) are applied to both classifiers. The parameter spaces are summarized below in Table II.

TABLE II
HYPERPARAMETER SEARCH SPACES

Classifier	Parameter	Randomized Search	Grid Search Values
Logistic Regression	C (regularisation)	log-uniform [0.001, 100]	0.001, 0.01, 0.1, 1, 10
Logistic Regression	Penalty	['l2']	['l2']
Logistic Regression	SMOTE k_neighbors	[3, 5, 7]	[3, 5]
HistGrad. Boosting	learning_rate	log-uniform [0.005, 0.3]	0.01, 0.05, 0.1
HistGrad. Boosting	max_iter	randint [100, 500]	100, 200
HistGrad. Boosting	max_depth	[3, 5, 7, None]	[3, 5]
HistGrad. Boosting	min_samples_leaf	randint [10, 100]	20, 50
HistGrad. Boosting	l2_regularization	uniform [0.0, 1.0]	0.0, 0.1

IV. RESULTS & DISCUSSION

A. Phase 1 - Baseline Performance (No SMOTE, Default Parameter)

In Table III the baseline cross-validated performance of both classifiers without SMOTE and with only the default hyperparameters is presented. Confusion matrix counts are accumulated across all five folds (approximately 394 total fraud cases validated). At baseline, both classifiers demonstrate the characteristic behaviour of models confronting severe class imbalance without any intervention in place. Therefore, this starting point of the experiment shows how each model behaves on the raw imbalanced data.

Here at baseline, Logistic Regression shows a high precision of (0.8795) and average recall (0.6345), and generates only 35 false alarms across all 5-folds. This conservative behaviour is the model's natural response to imbalanced data. This show it predicts fraud only when very confident making it to miss many actual frauds but very few false alarms. On the other hand, HistGradient Boosting at baseline shows more balanced but modest performance (F1 = 0.5574, 243 false alarms).

TABLE III
BASELINE PERFORMANCE: 5 – FOLD CV, NO SMOTE, DEAFULT PARAMETERS (TP/FP/FN/TN SUMMED ACROSS ALL 5 FOLDS, WITH 394 TOTAL FRAUD CASES)

Classifier Name	TP	FN	FP (False	TN (Legit	Precision	Recall	F1	ROC-	Avg	Time
Logistic Regression	250	144	35	227,416	0.8795	0.6345	0.7361	0.9789	0.7588	6s
HistGradient Boosting	249	145	243	227,208	0.5004	0.6322	0.5574	0.7285	0.4721	6s

B. Phase 2 - Effect of SMOTE Preprocessing (Sole Intervention)

Table IV shows the effect of adding of only SMOTE oversampling to both pipelines while retaining default hyperparameters. The introduction of SMOTE produces very different effects on the two classifiers. This is a major finding in this study. For Logistic Regression, False alarms increased by a factor of 165, from 35 to 5,775. Recall records a spike too from 0.6345 to 0.9011. This means the model now catches many more frauds, but at the expense of flagging high numbers of non-fraud as fraud. It is observed that ROC-AUC barely changed as it moved from 0.9789 to 0.9806. For HistGradient Boosting, SMOTE produces a substantial improvement: F1 rises from 0.5574 to 0.7146 (representing a 28.3% improvement). False alarms decreased from 243 to 195 across the five folds.

C. Key Finding

The key findings shown on Table IV indicate that both classifiers achieve high ROC-AUC after SMOTE is applied, recording 0.9806 for Logistic Regression and 0.9573 for HistGradient Boosting. By this metric alone, LR appears to be the stronger model. Yet it generated 5,775 false alarms compared to HistGB's 195 (i.e., increased by a factor of 29.6) being the difference in operational performance. ROC-AUC measured ranking quality, and SMOTE preserved it for both models. What it cannot measure is calibration. That is, how well do the predicted probability values reflect the true likelihood of fraud? The results from Tables III & IV show that LR's probabilities were inflated by SMOTE's rebalancing, elevating many legitimate transactions above the default threshold of 0.5. On the other hand, HistGB's non-linear boundary operated very differently. It absorbed this inflation more selectively. Two models with similarly high ROC-AUC can therefore be very divergent in operational behaviuor when one is mis-calibrated. This is precisely the limitation of ROC-AUC that Dal Pozzolo et al. [1] warned about, and this experiment demonstrates it with clarity using real transaction counts.

TABLE IV
EFFECTS OF SMOTE PREPROCESSING: 5-FOLD CV, DEFAULT PARAMETERS (TP/FP/FN/TN SUMMED ACROSS ALL 5 FOLDS)

Classifier	TP (Caught)	FN (Missed)	FP (False Alarm)	TN (Legit OK)	Precision	Recall	F1	ROC-AUC	Avg Prec	Time
LR + SMOTE	355	39	5,775	221,676	0.0580	0.9011	0.1090	0.9806	0.7476	13s
HistGB + SMOTE	326	68	195	227,256	0.6319	0.8275	0.7146	0.9573	0.7738	80s

TABLE V
SMOTE BIAS EVIDENCE SUMMARY (TABLES III & IV)

LR's False Alarms	35 (@baseline) increased to 5,775 (@SMOTE applied) = 165× increase
HistGB's False Alarms	243 (baseline) increased to 195 (@SMOTE applied) = decrease in false alarms
LR's ROC-AUC	0.9789 increased to 0.9806 (insignificant increase, almost unchanged)
Conclusion	Massive False Positive increase with stable ROC-AUC for LR = probability bias signature (Dal Pozzolo et al. [2])

D. Phase 3 - Hyperparameter Optimization Comparison

Table VI compares Randomized Search and Exhaustive Grid Search for both classifiers. For Logistic Regression, both methods produce almost nearly identical results, i.e., strong regularization (C ≈ 0.001) with SMOTE k_neighbors=5. As shown on Table VI, for HistGradient Boosting, Randomized Search substantially outperforms Grid Search: CV F1 of 0.8293 versus 0.7036. The optimal parameters found by Randomized Search, that is, learning_rate = 0.245approx., max_depth = 7, min_samples_leaf = 70, l2_regularization = 0.374approx., all fell outside the separated ranges tested by Grid Search. This directly demonstrates why continuous distribution sampling outperforms fixed grids for complex parameter spaces [15].

TABLE VI
OPTIMIZATION METHOD COMPARISON (CV F1 – SCORE, F1 AS OPTIMIZATION CRITERION)

Method	Combinations	Best CV	Runtime
LR — Randomized Search	20	0.1160	275s
LR — Grid Search	10	0.1176	33s
HistGB — Randomized Search	20	0.8293	2,678s
HistGB — Grid Search	48	0.7036	455s

E. Phase 4 - Final Evaluation on Isolated Test Dataset

Table VII presents the performance of all four optimized models on the 20% held-out test set. Table VIII shows the corresponding confusion matrices, which make the operational implications directly visible. HistGradient Boosting with Randomized Search is the best-performing model overall, achieving Precision 0.8218, Recall 0.8469, F1 0.8342, and ROC-AUC 0.9870, with only 18 false alarms on 56,962 test transactions shown in Table VIII. Logistic Regression with Grid Search achieves the same recall (0.9184, catching 90 of 98 test frauds) but generates plenty 1,404 - 1,415 false alarms. This represents 78 false alarms by Logistic Regression for every one false alarm by HistGradient Boosting model. See Table VIII.

TABLE VII
FINAL TEST SET PERFORMANCE
(80 – 2- HOLDOUT SPLIT)

Classifier	Precision	Recall	F1	ROC-AUC	Avg Precision
LR — Randomized Search	0.0598	0.9184	0.1123	0.9735	0.7251
LR — Grid Search	0.0602	0.9184	0.1131	0.9738	0.7189
HistGB — Randomized Search	0.8218	0.8469	0.8342	0.9870	0.8838
HistGB — Grid Search	0.4368	0.8469	0.5764	0.9815	0.7739

TABLE VIII
CONFUSION MATRICES:
TEST SET – (98 TOTAL FRAUD CASES, 56,864 LEGITIMATE)

Classifier	TN	FP (False	FN (Missed)	TP (Caught)	Frauds Caught
LR (Randomized Search)	55,449	1,415	8	90	90/98
LR (Grid Search)	55,460	1,404	8	90	90/98
HistGB (Randomized Search)	56,846	18	15	83	83/98
HistGB (Grid Search)	56,757	107	15	83	83/98

Table IX shown below consolidates the results all stages of the experiment from baseline to optimization stages. It makes the progressive specific effect of each intervention visible from a single Table view.

TABLE IX
COMPREHENSIVE SUMMARY
(FROM BASELINE TO SMOTE TO OPTIMIZED)

Classifier	SMOTE	Parameters	Evaluated	Precision	Recall	F1	ROC-AUC
Logistic Regression	No	Default	5-Fold CV	0.8795	0.6345	0.7361	0.9789
Logistic Regression	Yes	Default	5-Fold CV	0.0580	0.9011	0.1090	0.9806
Logistic Regression	Yes	Optimized	Test Set	0.0602	0.9184	0.1131	0.9738
HistGrad. Boosting	No	Default	5-Fold CV	0.5004	0.6322	0.5574	0.7285
HistGrad. Boosting	Yes	Default	5-Fold CV	0.6319	0.8275	0.7146	0.9573
HistGrad. Boosting	Yes	Optimized	Test Set	0.8218	0.8469	0.8342	0.9870

F. The core contribution - The probability bias finding

The main finding of this study can be stated thus: That SMOTE over-sampling induces posterior probability bias in Logistic Regression that is operationally consistent with, and by implication, extends the theoretical framework of Dal Pozzolo et al. [1] for under-sampling. The evidence is clear and unambiguous. Before SMOTE, Logistic Regression generated 35 false alarms across 5-CV folds. After SMOTE being the only change made, the false alarms spiked to 5,775, increasing by a factor of 165. At the same time, ROC-AUC increased marginally from 0.9789 to 0.9806. The model's ability to rank fraud above legitimate transactions remained intact. What SMOTE altered drastically was the probability calibration, that is, the model learned to assign inflated fraud probabilities because it was trained on a dataset where fraud constituted 50% of cases (after SMOTE balancing), and not the true 0.173% that existed pre-SMOTE.

Dal Pozzolo et al. [1] provided analytical proof showing that under-sampling produces this exact signature of 'ranking preserved and calibration broken'. They demonstrated it on this same dataset for under-sampling. This study demonstrates the same signature for over-sampling. The mechanism is identical in direction. That is, that any re-sampling strategy that changes the effective training class prior away from the true population prior will cause a bias posterior probabilities while leaving the ranking order (as demonstrated by marginal increase in ROC-AUC) intact.

G. Why The Two Classifiers (Logistic Regression and HistGradientBoosting) Behaved Differently

The response of HistGradient Boosting to SMOTE was very different in the experiment. The false alarms decreased from 243 to 195 and F1 improved from 0.5574 to 0.7146. This appears to contradict the bias expectation, but it does not really do so. The explanation lies in the nature/structure of the two models. Logistic Regression draws a single linear boundary through the feature space. Therefore, when SMOTE floods the training set with synthetic fraud examples, this boundary would shift towards predicting fraud, and affecting every region of feature space at the same time, including regions containing many legitimate transactions. The direct result is 5,775 false alarms flagged.

On the other hand, the decision boundaries built by HistGradient Boosting are more complex and non-linear. It can learn that synthetic SMOTE samples cluster in certain parts of feature space and weight them differently from real transactions in those regions. It treats and absorbs SMOTE's inflated fraud density more selectively. This helps it to detect real fraud without proportionally inflating false alarms. This model-dependent response to SMOTE is consistent with finding reported by Velarde et al. [12], who found that oversampling does not consistently benefit all classifiers.

H. The Isolated Role of class_weight='balanced'

An important methodological finding concerns the role of class_weight='balanced' as a separate intervention. In the initial version of this experiment, both SMOTE and class_weight='balanced' were applied simultaneously. In that set up. The results (not shown in this study) were almost the same as those seen here in the SMOTE - only experiment with respect to spike in false alarms. This confirms that the probability distortions introduced by class weighting alone are operationally benign. Class weighting adjusts the loss function during training, penalising misclassification of the minority class more heavily, but it does not dramatically shift the prior in the way that SMOTE does by creating entirely new synthetic training examples. When class_weight='balanced' was removed, leaving SMOTE as the sole intervention, the 165x false alarm increase became visible. This isolation experiment confirms that SMOTE, not class weighting, is responsible for the observed probability bias.

I. ROC-AUC (Presents a Misleading Metric on Imbalanced Data)

It is observed that both classifiers in the experiment achieved ROC-AUC above 0.97 throughout. The values that suggest almost the same quality of class discrimination (i.e., ability to detect fraud from non-fraud). Yet the operational performance of both classifiers after SMOTE was applied, differed by a factor of 29.6 in false alarms (5,775 vs 195 in CV), and a difference by a factor of 78.6 in false alarms (1,415 vs 18 on the test set). ROC-AUC measured ranking quality, which was preserved by SMOTE. It could not detect the calibration damage that SMOTE caused. Average Precision (which focuses entirely on the positive class), provided a more balanced expected information, that is, 0.7476 for LR versus 0.7738 for HistGB after SMOTE, which represents a more honest reflection with regards to their operational difference. This confirms Dal Pozzolo et al.'s [1] warning that ROC-AUC can be deceptive when posterior probabilities are miscalibrated.

J. Optimization and Model Expressiveness

The performance gap between the two classifiers widened at every stage. From a 0.43 F1 difference at baseline to a 0.72 F1 difference after full optimization. This progressive widening reflects the fact that every applied technique, whether SMOTE or hyperparameter tuning, did not benefits the models proportionately. It benefited the model that has sufficient capacity to exploit it more than the one that lacked that capacity. Logistic Regression's linear boundary is a structural limitation that no optimization can satisfactorily overcome when fraud patterns in the feature space are inherently non-linear as in datasets with class imbalance.

Randomized Search outperformed Grid Search for HistGradient Boosting (CV F1: 0.8293 vs 0.7036) while evaluating fewer combinations in less time. The optimal hyperparameters, that is, learning_rate = 0.245 approx., l2_regularization = 0.374 approx., both lay outside the separate grid of Grid Search. This is a confirmation of Bergstra and Bengio's [15] argument that continuous distribution sampling is more efficient for complex parameter spaces.

K. Limitations

This study acknowledges some limitations. The train : test partition is drawn from the same two-day transaction period by customers; PCA compression may create near-duplicate feature vectors across the boundary, meaning absolute independence of the test set employed cannot be guaranteed. This limitation arises because transactions occurring within a short temporal window may exhibit similar behavioral and statistical characteristics, which increases the possibility of overlap the PCA transformations.

V. CONCLUSION AND FUTURE WORK

This study set out to determine whether SMOTE oversampling induces posterior probability bias similar to the under-sampling induced bias demonstrated by Dal Pozzolo, Caelen, Johnson, and Bontempi in their work on under- sampling. The answer to this question, supported by clear and unambiguous experimental evidence, is yes. The four conclusions that summarize the findings are given below.

First, SMOTE-induced probability bias is confirmed and demonstrated with evidence. When SMOTE was applied as the sole class imbalance intervention to Logistic Regression (LR), false alarms increased by a factor of 165, from 35 to 5,775 across 5 Cross Validation (CV) folds, while ROC-AUC remained virtually unchanged at 0.9806. On the isolated test set, LR generated 1,415 false alarms at the default threshold of 0.5. This precise pattern of spikes in false alarm inflation with stable ranking (ROC-AUC), is the operational fingerprint (evidence) of posterior probability bias as described by authors of the earlier study. This study demonstrates this fingerprint for SMOTE over-sampling on the same dataset where Dal Pozzolo et al. demonstrated it for under-sampling. Thus, class imbalance not only hinders learning minority patterns but also distorts posterior probabilities when resampling methods are applied, whether by under-sampling (Dal Pozzolo et al., 2015) or by over-sampling (the empirical findings in this study). Second, class_weight='balanced' is not the source of the bias. An initial experiment applied both SMOTE and class_weight='balanced' simultaneously. And then SMOTE was isolated and evidence shows that class_weight='balanced is not the source of the bias. This confirms that class weighting introduces operationally benign distortions.

Third, model expressiveness and SMOTE response are inseparable. HistGradient Boosting responded to SMOTE with improved performance, that is, false alarms decreased

and F1 rose by 28.3%. With Logistic Regression's precision collapsed. Therefore, SMOTE benefits are model-dependent: a classifier must have sufficient capacity to exploit synthetic samples without being miscalibrated by them. Logistic Regression's linear decision boundary lacks this capacity on this dataset. Fourth, Randomized Search is the preferred optimization strategy. Randomized Search outperformed Exhaustive Grid Search for HistGradient Boosting (CV F1: 0.8293 vs 0.7036) while it spent less time evaluating fewer combinations. This empirically confirming the work of Bergstra and Bengio on this real-world imbalanced Kaggle Credit Card dataset.

The most important direction for future work is the formal implementation of Dal Pozzolo et al.'s probability correction formula adapted for SMOTE oversampling. Their formula - $p' = (\beta \times p\_s) / (\beta \times p\_s - p\_s + 1)$, where β = true_fraud_prior / smote_fraud_rate $\approx$ 0.00346 - should correct the upward probability shift that this study has identified as the source of Logistic Regression's false alarm inflation. If this formular is applied successfully, this would dramatically reduce false alarms while preserving recall, and as a result, will provide a practical remedy for the bias demonstrated in this study, and for the first time, formally validate Dal Pozzolo et al.'s correction framework in relation to over-sampling. Additional future directions should explore evaluation on a second dataset to assess generalizability.

REFERENCES

[1] A. Pozzolo, O. Caelen, R. Johnson, et al, "Calibrating probability with Underdamping for Unbalanced Classification", IEEE Symposium Series on Computational Intelligence, pp. 159 – 166, 2015. 10.1109/SSCI.2015.33

[2] S. Chakraborty, L. Dey, "Multi-objective, Multi-class and Multi-label Data Classification with Class Imbalance: Theory and Practices", Springer, Singapore; 2024. 10.1007/978-981-97962-29

[3] S. Bhattacharyya, S. Jha, K. Tharakunnel, & JC. Westland, "Data mining
for Credit Card Fraud: A comparative study", Decision Support Systems, Vol 50, pp 602-613, 2011.
10.1016/j.dss.2010.08.008

[4] M. Sopiyan, K. Fauziah, Y.F. Wijaya, "Fraud Detection Using Random Forest, Logistic Regression and HistGradient Boosting on Credit Cards", JUITA: Jurnal Informatika, Vol 10, pp. 77 – 87, 2022.
10.30595/juita.v10i1.12050

[5] R. J. Bolton, DJ. Hand, "Unsupervised Profiling Methods for Fraud Detection", Proc. Credit Scoring and Credit Control, Vol 2, pp. 5 – 7, 2001.

[6] H. Ali, MNM, Salleh, K. Hussain, et al, "A Review on Data Preprocessing Methods for Class Imbalance Problem", International Journal of Engineering & Technology, Vol 8, pp.390-397, 2019. 10.14419/ijet.v8i3.29508

[7] NV. Chawla, KW. Bowyer, LO. Hall, et al, "Synthetic minority oversampling technique", Journal pf Artificial Intelligence Research, Vol 16, pp.321-357, 2002.
10.1613/jair.953

[8] J.W. Osborne, "Improving your data transformations: Applying the Box-Cox Transformation", Practical Assessment Research and Evaluation, Vol 15, pp. 12, 2010

[9] J.H. Friedman, "Greedy Function Approximation: A gradient Boosting Machine", Annals of Statistics, Vol 29, pp. 1189-1232, 2001

[10] G. Ke. Q. Meng, T. Finley, et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree", Advances in Neural Information, Advances in Neural Information Processing Systems, Vol 30, pp3146 – 3154, 2017.
10.5555/3294996.3295074

[11] F. Pedregosa, G, Varoquaux, A, Gramfort, et al., "Scikit-learn: Machine Learning in Python", Journal of Machine Learning Research, Vol 12, pp.2825-2830, 2011.
10.5555/1953048.2078195

[12] G. Velarde, M, Weichert, A. Deshmunkh, et al, "Tree Boosting Methods for Balanced and Imbalanced Classification and Their Robustness Over Time in Risk Assessment", Intelligent Systems with Applications, Vol 22, 2024.
10.1016/j.iswa.2024.200354

[13] P. Hajek, MZ. Abedin, U Sivarajah, "Fraud Detection in Mobile Payment Systems Using an XGBoost-Based Framework", Information Systems Fronteirs Systems Frontiers, pp 1 – 19, 2022.
10.1007/s10796-022-10301-9

[14] T. Hastie, R. Tibshirani, J. Friedman, "The Elements of Statistical Learning", Springer, New York; 2009.
10.1007/978-0-387-84858-7

[15] J. Bergstra, Y. Bengio, "Random Search for Hyper-parameter Optimization", Journal of Machine Learning Research, Vol 13, pp281-305, 2012.
10.5555/2188385.2188395

[16] "Pipeline and composite estimators", (2026). Accessed: April 1, 2026. https://scikit-learn.org/stable/modules/compose.html

[17] "Credit Card Fraud Detection", (2015). Accessed: April 1, 2026: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.