UDC 681
DOI: 10.15587/1729-4061.2025.322626

IDENTIFYING THE IMPACT OF THE COMPLEXITY OF DATASETS IN BAYESIAN OPTIMIZED XGBOOST ON THE PERFORMANCE OF CLASSIFICATIONS FOR IMBALANCED CLASS DISTRIBUTION DATASETS

Sutarman
Corresponding author
Doctor of Philosophy, Associate Professor*
E-mail: sutarman@usu.ac.id
Putri Khairiah Nasution
Magister Sains, Lecturer*
Katrin Jenny Sirait
Magister Sains, Lecturer*
Cindy Novita Yolanda Panjaitan
Sarjana Sains, Student*
*Department of Mathematics
Universitas Sumatera Utara
Dr. T. Mansur str., 9, Padang Bulan,
North Sumatera, Indonesia, 20222

This study explores how dataset complexity affects the performance of XGBoost models optimized using Bayesian methods, focusing on datasets characterized by imbalanced class distributions. The main challenge is accurately identifying minority classes, which are often misdiagnosed due to the dominance of majority classes, impairing predictive power. Additionally, dataset complexity, as indicated by the coefficient of variation (14.64 % to 85.68 %), does not consistently correlate with improved model performance, highlighting the need for more targeted methods. High-dimensional datasets may not be as accurate as simpler ones and require the use of advanced approaches. By using Bayesian optimization, it is possible to fine-tune hyperparameters and improve classification performance on different types of datasets. This indicates that the selection of appropriate resampling techniques to match the characteristics of the dataset is critical, and that hyperparameter optimization plays an important role in building models with high accuracy. The applications extend to areas such as fraud detection and other fields where the categorization of minority groups is important. Through the use of efficient resampling techniques and advanced optimization methods, this study offers a comprehensive solution to the challenges of imbalanced datasets, enhancing the reliability of machine learning solutions. The variation in resampling techniques and optimizing model performance metrics can be attributed to the distribution of classes, the number of features, the complexity, and the characteristics of the datasets

Keywords: Bayesian optimization, eXtreme gradient boosting, imbalanced datasets, complexity of datasets, classification, confusion matrix, resampling techniques, hyperparameter tuning, performance evaluation, minority class identification

Received 25.11.2024
Received in revised form 17.01.2025
Accepted 10.02.2025
Published 24.02.2025

How to Cite: *Sutarman, S., Nasution, P. K., Sirait, K. J., Panjaitan, C. N. Y. (2025). Identifying the impact of the complexity of datasets in Bayesian optimized XGBoost on the performance of classifications for imbalanced class distribution datasets. Eastern-European Journal of Enterprise Technologies, 1 (4 (133)), 52–63. https://doi.org/10.15587/1729-4061.2025.322626*

1. Introduction

The fast-evolving domain of machine learning has always called for data classification with much accuracy, especially in the case of imbalanced class distribution. In predictive modeling, the most significant challenge arises from the presence of an imbalanced class distribution dataset when one class outweighs the other considerably. This is a common problem in many domains, including healthcare, finance, and cybersecurity, where misclassifying minority classes leads to serious consequences such as diseases going undiagnosed or fraud undetected [1]. The subject of dealing with imbalanced class distribution datasets is, therefore, very relevant and crucial for the further advancement of reliable applications in machine learning [2].

With the advent of big data, the complexity of the datasets has increased. High-dimensional datasets can be rich in information, but they can bring noise and complicate the learning process [3]. This can easily cause overfitting, where models perform well during training but fail to generalize to unseen data. In such a case, it becomes vitally important to understand the nature of the dataset complexity that impacts model performance for developing robust machine learning solutions. The challenge lies herein for any researcher or practitioner to overcome these complexities so that the models are accurate and reliable, at least on the minority classes, which usually tend to be the target of interest.

Bayesian optimization has cropped up as one of the strongest mechanisms for hyperparameter tuning, which has facilitated a systematic and efficient approach toward model optimization [4]. This bears relevance to the context of complex datasets, wherein traditional mechanisms of optimization usually fail. For example, it could help to fine-tune the model parameters and contribute to improvement when conditions are so inopportune under imbalance classes. This needs support on advanced optimization along with the techniques of effective resampling, such as SMOTE. There would then arise

better improvements within enhancing models for sensitivity pertaining to minority classes.

Even with Machine Learning with optimization, many are there in terms of resolving different arising problems presented through imbalanced data with its inherent complexities [5]. Although different directions have been taken by existing literature, there are still significant gaps with respect to the fine-grained interaction of dataset characteristics and model performance [6]. Moreover, the efficiency of various resampling techniques with some new complex optimization methods has not been studied yet. Since the world of data science is continuously changing, there is a need for research directed toward addressing these unresolved issues to assure the effective application of machine learning models in real-world scenarios.

Therefore, the study of dataset complexity in Bayesian optimization XGBoost is highly relevant in datasets with an imbalanced class distribution. Not only will it contribute to the theoretical understanding, but it will also provide practical solutions to enhance reliability and accuracy in predictive models across critical applications.

2. Literature review and problem statement

Class imbalance in machine learning has been one of the most discussed topics in recent years, especially in applications where misclassification of minority classes can have severe consequences. The paper by [7] presents the results of research related to class imbalance learning using Bayesian optimization in drug discovery. They illustrate that Bayesian methods can effectively enhance model performance in identifying minority classes. However, some of these problems remain open, like those related to scalability for these methods on high-dimensional datasets, which usually are filled with noise and thereby make the learning more complicated. This is probably due to some objective difficulties because such datasets are inherently so complex that generalization would be hindered by overfitting. [8] investigate several Bayesian optimization algorithms for big data classification under the MapReduce framework. Their results suggest that while Bayesian optimization may lead to improved classification performance, the intrinsic impossibility of traditional optimization methods to deal with the challenges of imbalanced datasets is still a formidable barrier. This means that there is a need for more targeted approaches that can adapt to the specific characteristics of imbalanced datasets.

[9] works on efficient hyperparameter tuning to predict the performance of students using Bayesian optimization. The research explains the significance of hyperparameter optimization in enhancing model accuracy, but the problem does not consider the intricacies brought about by imbalanced datasets. This is an indication that, although hyperparameter tuning is necessary, it must be supplemented with resampling techniques for better performance in the minority classes. [10] give a holistic review on some of the recent works in Bayesian optimization with a stress on its capability to enhance machine learning models. However, they also indicated that there is a lack of research in the interaction between dataset complexity and performance, particularly on imbalanced datasets. This lack of understanding underlines the need for further research to explore how Bayesian optimization can be effectively applied to enhance the performance of models like XGBoost in these challenging scenarios.

[11] propose a hybrid Bayesian optimization approach for malicious access and anomaly detection in IoT systems. Their work demonstrates the effectiveness of combining Bayesian optimization with advanced machine learning techniques. However, the cost part in terms of computational resources and time associated with complex models can make such approaches impractical for real-time applications, particularly in high-dimensional settings. [12] present the performance of a federated Bayesian optimization XGBoost model on detecting cyberattacks in the Internet of Medical Things. While their findings showed very promising results, they did not deeply delve into the challenge of an imbalanced dataset, which means more research might be needed to show the optimal performance of this proposed model.

[13] propose a new Bayesian-Optimization-Based Synthetic Minority Over-Sampling Technique called BO-SMOTE. According to their findings, the use of Bayesian optimization in conjunction with SMOTE has the potential to improve the classification performance of minority classes. Nevertheless, this is not yet comprehensively studied on high-dimensional datasets, which calls for further research. In addition, [14] detail improving algorithm interpretability and accuracy when borderline-SMOTE is employed, using Bayesian optimization. Although, their conclusion says that though this enhances the performance of models significantly, it faces huge problems in processing such big dimensionally imbalanced datasets.

In light of the above, there is an urgent need for Bayesian-optimized XGBoost study concerning dataset complexity for datasets with imbalanced class distribution. This research shall strive toward adopting a holistic approach through structured studies into how characteristic features of datasets interplay with resampling methods and hyperparameter optimization to address the challenge that imbalanced datasets pose effectively and further work toward developing predictive models as more dependable and accurate, especially in critical applications.

3. The aim and objectives of the study

This study aims to identify the relationship between dataset complexity and the performance of XGBoost when optimized through Bayesian methods, particularly in the context of imbalanced class distribution datasets. It will facilitate the development of more robust machine learning models by leveraging efficient resampling techniques and advanced optimization methods, improving performance in applications such as fraud detection and the categorization of minority groups.

To achieve this aim, the following objectives are accomplished:

– to evaluate SMOTE, ENN, and SMOTE-ENN as methods for addressing class imbalance and analyze their impact on class distributions, highlighting their advantages, limitations, and effectiveness in balancing datasets;

– to analyze feature dimensionality, class overlap and sparsity, overfitting, and feature redundancy across varying complexity levels;

– to utilize Bayesian optimization to tune XGBoost hyperparameters, including learning rate, depth, estimators, and regularization, by minimizing cross-validation error for efficient and precise model performance optimization;

– to evaluate tuned XGBoost models using confusion matrix-based metrics, analyzing dataset characteristics' impact on performance and conducting sensitivity analyses on size, feature richness, and class complexity across diverse scenarios.

4. Materials and methods

4. 1. Object and hypothesis of the study

This study examines the impact of dataset complexity on the performance of Bayesian-optimized XGBoost models for imbalanced classification. It hypothesizes that dataset complexity influences model performance and that Bayesian optimization and appropriate resampling techniques enhance accuracy, particularly for minority class identification. The study assumes that Bayesian optimization effectively fine-tunes hyperparameters, resampling techniques impact model sensitivity, and higher complexity does not always improve performance. Simplifications include focusing on specific resampling methods, attributing performance improvements mainly to hyperparameter tuning and resampling, and generalizing the effects of dataset complexity without domain-specific considerations.

4. 2. Methods

The study utilized established methods for addressing the complexity of datasets using Shannon Entropy as a metric of complexity. The study also addresses class imbalance, including the Synthetic Minority Over-sampling Technique (SMOTE) for generating synthetic samples, Edited Nearest Neighbors (ENN) for cleaning the dataset, and Bayesian Optimization for setting hyperparameter tuning. Experiments were conducted using Python, with libraries such as XGBoost, Bayesian Optimized XGBoost, Imbalanced-learn, Scikit-learn, and Optuna.

Controlled experimental conditions included data preprocessing to handle missing values and normalize features, along with stratified k-fold cross-validation ($k=10$) for robust model evaluation and random seed settings for reproducibility. The adequacy of the Bayesian Optimized XGBoost models was validated through performance based on metrics of confusion matrix like accuracy, specificity, precision, recall, and F-Measure, as well as sensitivity analyses to assess the impact of dataset complexity on performance.

4. 3. Metrics of complexity of datasets

There are several metrics that can be used to assess the complexity of datasets [15]. These metrics are based on dimensionality, feature distribution and skewness, correlation matrix, collinearity condition number, and feature integrations. Dimensionality refers to the number of features in the datasets. High-dimensional datasets can cause many modeling problems. This is also known as the dimensionality curse. In addition, skewness measures the asymmetry of a dataset. High asymmetry complicates the modeling process. Meanwhile, the entropy metric can be used to measure the amount of uncertainty or randomness in the data distribution. The higher the entropy value, the more complex the datasets. The most commonly used metric is Shannon entropy. The formula for Shannon entropy can be expressed as:

$$H(X)=-\sum p(x)\log_2 p(x), \tag{1}$$

where $H(X)$ denotes the entropy of a random variable X and $p(x)$ represents the probability of occurrence of each outcome x. If there are v random variables then the joint Shannon entropy over this distribution can be written as:

$$H(X_1,X_2,\ldots,\mathrm{X}_v)= \\ =-\sum_{x\in\chi^v} p(x_1,x_2,\ldots,x_v)\log_2 p(x_1,x_2,\ldots,x_v), \tag{2}$$

and the mean of Shannon entropy for the random variables is:

$$h_v=\frac{1}{l}H(X_1,X_2,\ldots,X_v). \tag{3}$$

The standard deviation and coefficient of variation of Shannon entropies can be written each as:

$$s=\left\{\frac{1}{v-1}\sum\nolimits_{\mathrm{i}=1}^{v}\left(H(x_i)-h_v\right)^2\right\}^{1/2} \text{ and } CV=\frac{h_v}{s}. \tag{4}$$

The standard deviation describes the variation of Shannon entropy values. Meanwhile the coefficient of variation describes the standardized variation of Shannon entropy.

4. 4. Synthetic minority oversampling technique

Synthetic Minority Oversampling Technique (SMOTE) [16] is a technique that can be used to balance imbalanced class distribution datasets. SMOTE works by generating synthetic samples for minority classes. This is achieved by selecting an instance and forming new samples along the line segment connecting it to its nearest neighbors. By augmenting minority classes in this way, SMOTE makes it possible to balance datasets and allows machine learning algorithms to learn more effectively and improve predictive performance on unrepresented classes.

Now consider one sample x_i, a new sample x_{new} will be generated considering its k nearest-neighbors (corresponding to k-neighbors). Then, one of these nearest-neighbors x_{zi} is selected and a sample is generated as:

$$x_{new}=x_i+\lambda(x_{zi}-x_i), \tag{5}$$

where λ is a random number in the range [0, 1].

4. 5. Edit Nearest Neighbors

In addition to SMOTE, Edit Nearest Neighbors (ENN) is another sampling method that can be used to balance class distribution datasets. This method is similar to SMOTE. SMOTE aims to balance the class distribution by generating a new minority class, ENN works by cleaning the number of instances in the majority class while preserving the minority class [17]. This process smoothes the decision boundary between classes and reduces overfitting by retaining only the representative majority instances. Furthermore, ENN method offers advantages when the class distribution datasets are significantly skewed. It will improve the accuracy of the classification model while maintaining the integrity of the minority class. ENN examines the k-nearest neighbors of each instance in the datasets. The most common choice for k is 3. Now, let D be the dataset containing n instances, x_i be an instance in the dataset, and $N_k(x_i)$ be the set of k nearest neighbors of x_i. For each instance, $x_i\in D$, identify the k nearest neighbors, $N_k(x_i)=\{x_{j_1},x_{j_2},\ldots,x_{j_k}\}$. Count the majority and minority class neighbors, and $c(x)$ be the class label of instance x $\mathrm{count}_{majority}=\sum_{j=1}^{k} I\left(c(x_j)=c_{majority}\right)$ and $\mathrm{count}_{minority}=\sum_{j=1}^{k} I\left(c(x_j)=c_{minority}\right)$. In this case, I is the indicator function, which is 1 if the condition is true and 0 otherwise. If $\mathrm{count}_{minority}>\mathrm{count}_{majority}$, then $D'=D\backslash\{x_i\}$ (remove x_i). Where D' is the new, edited dataset.

4. 6. SMOTE-ENN

[18] introduced a hybrid algorithm that combines SMOTE and ENN to improve the accuracy of dropping instances.

They attempted to solve this problem on healthcare datasets. Their proposed algorithm effectively addresses this problem by generating synthetic samples for the underrepresented class (missed abortion cases) and then refining the dataset by removing the noisy ones with ENN. This approach not only improves the balance of the dataset, but also enhances the model's ability to discriminate and classify. SMOTE-ENN is performed sequentially. SMOTE is applied to the datasets first, followed by ENN. In SMOTE, synthetic instances are generated for the minority class. While in ENN, after the synthetic instances are generated, instances (typically from the majority class) that are likely to be noise are removed.

4. 7. Bayesian optimization

Bayesian optimization (BO) is a strategy to optimize objective functions that are costly to evaluate [19]. It is particularly useful in scenarios where the objective function is noisy, expensive, or time-consuming to compute. [20] emphasizes the importance of choosing the right acquisition function to balance exploring and exploiting in the optimization process. BO is a sequential design strategy for global optimization of black-box functions [21] that does not assume any functional forms. This optimization is typically applied to hyperparameter tuning for machine learning models. In particular, BO is useful when the objective function is non-convex, noisy, or has many local optima. This makes traditional optimization methods such as grid search or random search less effective or computationally expensive. BO uses an acquisition function to determine the next point to sample, balancing the exploration of the search space domain to produce better performance. This process is iterative and allows efficient navigation through complex landscapes. Often, only one evaluation function is needed to obtain fewer optimal or near-optimal solutions. BO uses a probabilistic model to find the optimal parameters.

Now, let $f(x)$: $X \rightarrow R$ be the unknown objective function, where $X \rightarrow R^d$ is a bounded subset of d-dimensional space, $p(f|D)$ is a prior distribution that represents our beliefs about the function before observing any data. Typically it is a Gaussian Process and is defined as:

$$f(x) \sim \mathcal{GP}\left(\mu(x), k(x,x')\right), \tag{6}$$

where ($\mu(x)$) is the mean function and ($k(x,x')$) is the covariance (kernel) function. Then let ($\alpha(x)$) be an acquisition function guiding the optimization process by determining the next sample point based on the current model. A frequently used option for the acquisition function is Expected Improvement (EI), namely:

$$\alpha_{\mathrm{EI}}(x) = E\left[\max\left(0, f^* - f(x)\right)|\mathcal{D}\right]. \tag{7}$$

In addition, we recognize Probability of Improvement (PI), written as:

$$\alpha_{\mathrm{PI}}(x) = P\left(f(x) > f^*|\mathcal{D}\right), \tag{8}$$

and Upper Confidence Bound (UCB), written as:

$$\alpha_{\mathrm{UCB}}(x) = \mu(x) + \kappa\sigma(x), \tag{9}$$

where κ is a parameter that controls the exploration-exploitation trade-off.

Based on the above equations, the BO process begins by selecting a set of points to evaluate $f(x)$. This is followed by fitting a Gaussian process model to the observed points. After this step, the acquisition function is maximized ($\alpha(x)$) to obtain the next point x_{next} and evaluate the objective function of that point, $f(x_{next})$. Finally, add this new observation to the datasets and repeat until convergence is reached or stopping criteria are met. In addition, Bayesian optimization has been implemented in the Python Optuna [22] library. This library uses a Bayesian-based approach to efficiently optimize model hyperparameters.

4. 8. Bayesian optimized XGBoost

XGBoost (eXtreme gradient boosting) is a classification method developed based on the gradient boosting framework. This method improves the performance and efficiency of class classification [23]. This developed method optimizes the training speed and prediction accuracy of machine learning models. In principle, this method employs a sequential ensemble strategy that integrates the predictions of multiple weak learners, primarily decision trees. The core concept involves iteratively adding trees, with each subsequent tree aiming to correct the errors made by the preceding ones.

Now let $L(\theta)$ be the objective function for XGBoost and it is stated as:

$$L(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k). \tag{10}$$

In equation (10), $l(y_i, \hat{y}_i)$ is the loss function that measures how well the model predicts the target values y_i given the predictions $\hat{y}_i$ and $\Omega(f_k)$ is a regularization term that penalizes the complexity of the model, where f_k represents the k-th tree in the ensemble, and K is the total number of trees. Then, to avoid overfitting, regularization term is introduced. It is done by controlling the depth of the trees:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2. \tag{11}$$

In equation (11), T is the number of leaves in the tree, w_j is the weight of each leaf, γ and λ are parameters that control the complexity and the number of leaves, respectively. Parameters like γ and λ are a very important part in Bayesian optimization. Therefore, to implement BO in XGBoost, we need to do hyperparameter tuning. This is due to the efficiency aspect of navigating a complex and high-dimensional search space. BO builds a probabilistic model related to the objective function. This probabilistic model will balance the exploration of unknown regions with the exploitation of known regions to produce better performance. This is important for XGBoost because the relationship between hyperparameters and model performance can be nonlinear. By using fewer evaluations to find the optimal hyperparameters, BO not only reduces computation, but also improves the model training process.

The following, Table 1, are the steps for implementing Bayesian XGBoost [23].

The XGBoost model optimization using Bayesian methods, based on Table 1, starts by defining the objective function to minimize log loss. Key hyperparameters are identified, and the Optuna framework is employed for efficient searching. A Bayesian optimization function evaluates model performance through validation scores. The optimizer is initialized for a set number of iterations to explore hyperparameters. After retrieving the best configurations, the final model is trained, and its performance is assessed using accuracy, precision, recall, and

F1-score. The results are then interpreted to derive insights into the optimization process and its practical implications.

Table 1

Steps in Bayesian optimized XGBoost

Step Number	Description
1. Define the Objective	Set the objective function for the XGBoost model. The objective of classification tasks is log loss
2. Initial Hyperparameter Setup	Identify the hyperparameters to optimize: Learning rate, Maximum depth of a tree, Minimum sum of instance, Subsample, Proportion of features, Minimum loss reduction
3. Implement the Optuna	Implement the hyperparameter optimization algorithm
4. Define the Bayesian Optimization Function	Create input and return the validation score and evaluate model performance reliably
5. Run Bayesian Optimization	Initialize the optimizer, specify the number of iterations
6. Retrieve Best Hyperparameters	Retrieve the optimized hyperparameters from the hyperparameter optimization process
7. Train the Final Model	Train the final model based on the optimized hyperparameters
8. Evaluate Performance	Assess the performance: accuracy, specificity, precision, recall, F1-score for classification
9. Interpret Results	Interpret the performance results and identify areas for improvement

4. 9. Stratified *K*-fold cross-validation

Stratified K-fold cross-validation, hereinafter referred to as SKCV, is one of the most important techniques in machine learning. It is often used to evaluate the performance of a classification model, especially when dealing with imbalanced class distribution datasets [24]. SKCV allows each fold used in cross-validation to remain in the same proportion to the entire dataset. This technique provides more reliable performance estimates by eliminating the bias that can occur when unrepresentative classes are within particular folds. In general, this technique, at random, divides the datasets into *K* folds. Take the fold as a hold out or as a validation set. Meanwhile, the remaining *K*–1 folds serve as the training set. Furthermore, the stratification in the *K*-fold cross-validation allows each fold to be close to the class distribution as a whole dataset. This makes this technique very special, particularly in classification problems where the class distribution is not balanced. Table 2 shows the steps in stratified *K*-Fold cross-validation.

Table 2

Steps in stratified *K*-Fold cross-validation

Step number	Description
1. Shuffle the dataset randomly	Randomly shuffle the data to eliminate bias
2. Split the dataset into *k* folds	Divide the data into *k* equal folds, ensuring each fold represents the overall class distribution
3. Each unique fold	Take the fold as a hold, take a training dataset, fit a model, evaluation
4. Summarize the performances	Calculate and report the performance metrics for each fold and average them over all folds

Table 2 describes the steps that involve several keys to ensure reliable model evaluation. First, the dataset is shuffled randomly to eliminate any order bias. Next, it is divided into *k* folds, maintaining the proportion of class labels in each fold. For each unique fold, one fold is held out as the validation set while the remaining *k*–1 folds are used for training the model. The model is then fitted and evaluated on the held-out fold. Finally, the performance metrics from each iteration are summarized to provide an overall assessment of the model's effectiveness.

4. 10. Performance metrics

There are some metrics used to measure the performance of classification tasks [25]. The metrics are derived from the confusion matrix. For binary classification, the confusion matrix is presented as Table 3.

Table 3

Confusion matrix

Actual class	Predicted class	
	Positive (1)	Negative (0)
Positive (1)	True Positive (*TP*)	False Negative (*FN*)
Negative (0)	False Positive (*FP*)	True Negative (*TN*)

The confusion matrix is a key evaluation tool for classification models, summarizing prediction outcomes into four components: True Positive (*TP*) for correctly predicted positives, False Negative (*FN*) for positives incorrectly predicted as negatives, False Positive (*FP*) for negatives incorrectly predicted as positives, and True Negative (*TN*) for correctly predicted negatives. It enables the calculation of performance metrics like accuracy, precision, recall, and F1-score, providing insights into the model's classification effectiveness.

Based on Table 3, we define:

a) accuracy.

Accuracy is found by:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \quad (12)$$

Accuracy is a metric used to measure the overall correctness of a classification model by calculating the proportion of correctly predicted instances (both true positives and true negatives) out of the total instances in the dataset;

b) specificity.

Specificity is found by:

$$specificity = \frac{TN}{TN + FP}. \quad (13)$$

Specificity is a metric used to measure a classification model's ability to correctly identify negative instances (true negatives) out of all the actual negative instances;

c) recall.

Recall is calculated by:

$$recall = \frac{TP}{TP + FN}. \quad (14)$$

Recall is a metric used to measure a classification model's ability to correctly identify all positive instances (true positives) out of the total actual positive instances;

d) precision.

Precision is calculated by:

$$precision = \frac{TP}{TP + FP}; \quad (15)$$

e) F-measure.

F-measure is calculated by:

$$FM = \frac{2 \times p \times sn}{p + sn}. \quad (16)$$

The F-measure, also known as the F1-score, is a metric that combines both precision and recall into a single score to evaluate a classification model's performance, particularly when there is an uneven class distribution.

Accuracy represents the overall correctness of a classification model. Specificity measures the proportion of correctly identified negative instances (*TN*) out of all actual negative instances (*TN*+*FP*). A high specificity indicates that the model is good at identifying negatives correctly. Meanwhile, Recall measures the proportion of correctly identified positive instances (*TP*) out of all actual positive instances (*TP*+*FN*). A high recall indicates that the model is good at finding all positive instances. Precision measures the proportion of correctly identified positive instances (*TP*) out of all instances that were predicted as positive (*TP*+*FP*). A high precision indicates that the model is good at making accurate positive predictions. Furthermore, *F1*-Score, a harmonic mean of precision and recall, is used to balance both metrics. It provides a single value representing the overall performance of a model.

4. 11. Experiments and datasets

This section outlines the methodologies and algorithms employed to examine the impact of dataset complexity on Bayesian Optimized XGBoost for imbalanced class datasets. Key components include the Shannon Entropy metric and the processes involved in applying SMOTE, ENN (Edited Nearest Neighbors), SMOTE-ENN, Bayesian Optimization, and Bayesian Optimized XGBoost. In summary, the steps of the data analysis process are shown in Fig. 1.

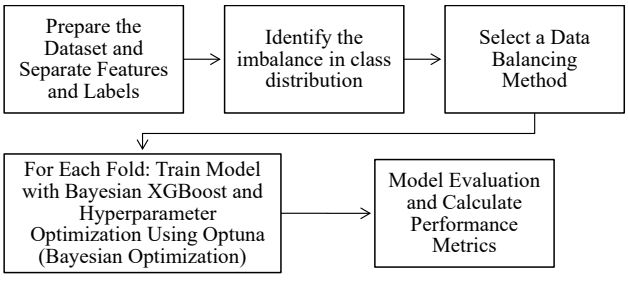

Fig. 1. Steps of the data analysis process

The experimental process, described in Fig. 1, commences with dataset preparation, involving the analysis of eight datasets (outlined in Table 4) to examine the dataset complexity. A synthetic dataset is also included to evaluate the consistency of analytical results between synthetic and real-world datasets. Synthetic datasets are designed to incorporate variations, noise, and anomalies based on specific distributions, providing a controlled environment to assess the robustness of the applied methods and algorithms. The varying characteristics of these datasets are quantified using Shannon Entropy.

In the second step, we identify imbalance in class distribution. In the third step, data balancing methods – SMOTE, ENN, and SMOTE-ENN – are applied to demonstrate their influence on classification performance. The fourth step involves defining the objective function for Bayesian optimization. Subsequently, the fourth step specifies the range of hyperparameter values to be explored during Bayesian optimization. In this step, Bayesian optimization is executed and trains the model with Bayesian XGBoost and hyperparameter optimization using Optuna (Bayesian optimization).

Finally, the fifth step evaluates model performance using metrics such as accuracy, specificity, precision, recall, and F1 score.

4. 12. Datasets and hyperparameter range setting

The experimental datasets used, as shown in Table 4, are sourced from kaggle.com, specifically: Microcalcification (Mic), Pima Indian Diabetes (PID), Predicting Manufacturing Defects (PMD), Dermatology (Der), Body Mass Index (BMI), CDC Diabetes (CDC), Star Classification (SC), and one synthetic dataset (Syn).

Table 4

Experimental datasets

Data	#Obs (#Feats)
Mic	11,183 (6)
	–1 (97.67 %)
	1 (2.32 %)
PID	768 (8)
	0 (65.10 %)
	1 (34.89 %)
PMD	3,240 (16)
	1 (84.04 %)
	2 (15.96 %)
Der	366 (34)
	1 (30.60 %)
	2 (16.67 %)
	3 (19.67 %)
	4 (13.39 %)
	5 (14.21 %)
	6 (5.46 %)
BMI	500 (3)
	0 (2.6 %)
	1 (4.4 %)
	2 (13.8 %)
	3 (13.6 %)
	4 (26 %)
	5 (40.0 %)
CDC	253,680 (22)
	0 (84.21 %)
	1 (1.82 %)
	2 (13.93 %)
SC	100,000 (18)
	0 (59.45 %)
	1 (18.96 %)
	2 (21.59 %)
Syn	150,000 (31)
	0 (10.01 %)
	1 (9.99 %)
	2 (9.99 %)
	3 (10.01 %)
	4 (9.97 %)
	5 (10.02 %)
	6 (9.99 %)
	7 (10.00 %)
	8 (10.00 %)
	9 (10.01 %)

This table displays the number of observations (#Obs), number of features (#Feats), mean, standard deviation (Std), and coefficient of variation (CV). Furthermore, synthetic data was generated using datasets from the sklearn and pandas libraries. The function used was *make_classification* with 150,000 instances, 30 features (including 25 informative features and 5 redundant features). The *n_classes* parameter was set to 10 to accommodate multi-class classification. To maintain consistency across different runs, the *random_state* parameter was set to 42.

Table 4 provides an overview of eight datasets with respect to the number of observations and the number of features. The number of observations in these datasets varies between 366 and 253,680. The same happens with the number of features: between 3 and 31 features.

The parameters presented in Table 5 were selected as a good starting point for the Bayesian Optimization (BO) XGBoost model. These parameters were iteratively updated during the optimization process based on the model's performance on a validation set. The parameter values in this table will serve as the basis for determining the best hyperparameters for Bayesian XGBoost.

Table 5

Initial values for parameters of BO XGboost

Hyperparameter	Default values
max_depth	6
learning_rate	0.3
n_estimators	100
subsample	1
colsample_bytree	1
min_child_weight	1
scale_post_weight	1
gamma	0
reg_lambda	1
reg_alpha	0

Table 5 presents initial values of the parameters involved in BO for XGBoost. The hyperparameters are max_depth, learning_rate, n_estimators, subsample, colsample_bytree, min_child_weight, scale_post_weight, gamma, reg_lambda, and reg_alpha. Each of these has been given default values to serve as starting points for the optimization using BO. Such default values come from common choices of hyperparameters to optimize XGBoost models and may serve as a proxy when such values must be changed internally along with the optimization. BO is sensitive to initial values, as this choice can impact the balance of exploration and exploitation in its process and further affect the performances of the tuned XGBoost model.

5. Results of Bayesian optimized XGBoost performance on imbalanced datasets with varying complexity

5. 1. Balancing the class distribution of the datasets

This section presents the results of balancing the class distribution using different handling techniques, including SMOTE, ENN, and SMOTE-ENN, in addressing class imbalance and their impact on balanced datasets. We applied each of the techniques to the datasets and analyzed their impact on class distributions.

The results of balancing datasets using SMOTE, ENN, SMOTE-ENN for every dataset are presented in Table 6.

Table 6

Balanced datasets

Data	Resampling		
	SMOTE	ENN	SMOTE -ENN
Mic	*n*=17,472	*n*=8,765	*n*=16,689
	–1 (50 %)	–1 (97.60 %)	–1 (50.48 %)
	1 (50 %)	1 (2.40 %)	1 (49.52 %)
PID	*n*=802	*n*=415	*n*=418
	0 (50 %)	0 (48.67 %)	0 (43.78 %)
	1 (50 %)	1 (51.33 %)	1 (56.22 %)
PMD	*n*=4354	*n*=1694	*n*=2691
	1 (50 %)	1 (75.50 %)	1 (36.86 %)
	2 (50 %)	2 (24.50 %)	2 (62.02 %)
Der	*n*=486	*n*=264	*n*=459
	1 (16.67 %)	1 (29.92 %)	1 (17.21 %)
	2 (16.67 %)	2 (14.02 %)	2 (16.07 %)
	3 (16.67 %)	3 (21.97 %)	3 (17.43 %)
	4 (16.67 %)	4 (11.74 %)	4 (16.99 %)
	5 (16.67 %)	5 (15.91 %)	5 (17.65 %)
	6 (16.67 %)	6 (6.44 %)	6 (17.65 %)
BMI	*n*=954	*n*=292	*n*=833
	0 (16.67 %)	0 (4.11 %)	0 (1.74 %)
	1 (16.67 %)	1 (3.77 %)	1 (1.70 %)
CDC	*n*=512,724	*n*=126,973	*n*=202,944
	0 (33.33 %)	0 (96.09 %)	0 (84.21 %)
	1 (33.33 %)	1 (2.90 %)	1 (1.82 %)
	2 (33.33 %)	2 (1.01 %)	2 (13.97 %)
SC	*n*=142,755	*n*=46,074	*n*=62,989
	0 (33.33 %)	0 (52.71 %)	0 (32.24 %)
	1 (33.33 %)	1 (32.91 %)	1 (29.67 %)
	2 (33.33 %)	2 (14.37 %)	2 (38.09 %)
Syn	*n*=121,230	*n*=60,732	*n*=55,561
	0 (10.00 %)	0 (8.20 %)	0 (9.46 %)
	1 (10.00 %)	1 (7.86 %)	1 (8.71 %)
	2 (10.00 %)	2 (10.03 %)	2 (11.27 %)
	3 (10.00 %)	3 (8.60 %)	3 (9.47 %)
	4 (10.00 %)	4 (19.53 %)	4 (9.81 %)
	5 (10.00 %)	5 (8.11 %)	5 (8.81 %)
	6 (10.00 %)	6 (8.53 %)	6 (9.79 %)
	7 (10.00 %)	7 (9.13 %)	7 (10.03 %)
	8 (10.00 %)	8 (10.07 %)	8 (11.15 %)
	9 (10.00 %)	9 (9.95 %)	9 (11.47 %)

Table 6 presents a comprehensive overview of balanced datasets across various applications, detailing the results of different resampling techniques including SMOTE, ENN, and SMOTE-ENN. Each dataset, such as Microcalcification, Pima Indians Diabetes, and CDC Diabetes, is characterized by its sample size and the distribution of classes, highlighting the effectiveness of resampling methods in addressing class imbalance. The table illustrates the percentage distribution of each class for the original and resampled datasets, providing insights into the performance of these techniques in enhancing classification outcomes.

5. 2. Complexity of datasets

Table 7 presents Shannon Entropy as a measure of complexity of experimental datasets. These values are used to investigate their impact on XGBoost performance.

Table 7

The Values of Shannon Entropy

Data	Shannon entropy		
	Mean	Std	CV
Mic	0.531	0.289	52.59 %**
PID	0.539	0.278	42.29 %**
PMD	0.799	0.287	40.17 %**
Der	0.412	0.230	38.04 %**
BMI	0.607	0.435	40.00 %**
CDC	0.249	0.205	85.68 %***
SC	0.728	0.341	46.94 %**
Syn	0.968	0.177	14.64 %*

*Note: * – low, ** – moderate, *** – high.*

Table 7 reveals that the mean Shannon entropy varies from 0.249 to 0.799, and the standard deviation varies from 0.177 to 0.435. The dispersion in the CVs implies that, in fact, different entropy variability exists across the datasets.

5. 3. Hyperparameter tuning with Bayesian optimization

For the purpose of Bayesian XGBoost Optimization, it is necessary to determine the optimal hyperparameter values. These values are presented in Table 8.

Table 8 shows the best values of the important hyperparameters of Bayesian XGBoost, on different datasets, using three resampling techniques: SMOTE, ENN and SMOTE-ENN. Among the important hyperparameters are max_depth, learning_rate, n_estimators, subsample, colsample_bytree, min_child_weight, scale_pos_weight, gamma, reg_lambda, reg_alpha, each with different values according to the various techniques. For example, while SMOTE often favored higher values for n_estimators and learning rates, ENN often required lower valued parameters, showing the disparate impacts these different resampling methods have on model performance. This further supports that in handling an imbalanced dataset, proper hyperparameter tuning with respect to the chosen resampling strategy should be done to improve the classification performance.

Table 8

Best Hyperparameters for Bayesian XGBoost

Data	Hyperparameter	SMOTE	ENN	SMOTE-ENN
1	2	3	4	5
Mic	max_depth	7.00	7.00	7.00
	learning_rate	0.29	0.15	0.25
	n_estimators	472.00	323.00	695.00
	Subsample	0.88	0.64	0.83
	colsample_bytree	0.82	0.95	0.51
	min_child_weight	4.00	6.00	2.00
	scale_post_weight	4.00	6.00	8.00
	Gamma	0.37	0.44	0.07
	reg_lambda	295.54	628.86	149.08
	reg_alpha	193.27	121.21	340.19
PID	max_depth	5.00	7.00	8.00
	learning_rate	0.25	0.13	0.19
	n_estimators	798.00	453.00	667.00
	Subsample	0.61	0.52	1.00
	colsample_bytree	0.54	0.75	0.58
	min_child_weight	6.00	10.00	1.00
	scale_post_weight	1.00	1.00	1.00
	Gamma	0.14	0.50	0.41
	reg_lambda	503.53	803.26	216.83
	reg_alpha	139.28	52.78	7.52
PMD	max_depth	3.00	9.00	3.00
	learning_rate	0.18	0.08	0.01
	n_estimators	960.00	103.00	403.00
	Subsample	0.74	0.85	0.93
	colsample_bytree	0.59	0.95	0.86
	min_child_weight	7.00	3.00	1.00
	scale_post_weight	3.00	2.00	8.00
	Gamma	0.24	0.30	0.06
	reg_lambda	651.29	126.61	918.33
	reg_alpha	492.94	45.22	593.06
Der	max_depth	6.00	4.00	10.00
	learning_rate	0.03	0.12	0.24
	n_estimators	629.00	277.00	472.00
	Subsample	0.78	0.72	0.59
	colsample_bytree	0.78	0.77	0.52
	min_child_weight	10.00	6.00	4.00
	scale_post_weight	9.00	6.00	9.00
	Gamma	0.35	0.34	0.45
	reg_lambda	731.81	418.48	825.39
	reg_alpha	51.64	72.58	479.42

Continuation of Table 8

1	2	3	4	5
BMI	max_depth	8.00	7.00	6.00
	learning_rate	0.13	0.10	0.11
	n_estimators	323.00	980.00	401.00
	Subsample	0.55	0.97	0.73
	colsample_bytree	0.59	0.64	0.92
	min_child_weight	2.00	3.00	8.00
	scale_post_weight	6.00	3.00	6.00
	Gamma	0.41	0.07	0.04
	reg_lambda	260.61	264.93	544.24
	reg_alpha	19.09	10.60	62.95
CDC	max_depth	9.00	5.00	9.00
	learning_rate	0.08	0.18	0.23
	n_estimators	890.00	715.00	745.00
	Subsample	0.85	0.67	0.96
	colsample_bytree	0.78	0.53	0.82
	min_child_weight	5.00	2.00	5.00
	scale_post_weight	7.00	5.00	3.00
	Gamma	0.36	0.15	0.14
	reg_lambda	777.05	467.16	157.45
	reg_alpha	94.86	97.55	72.62
SC	max_depth	9.00	7.00	5.00
	learning_rate	0.28	0.08	0.05
	n_estimators	717.00	729.00	149.00
	Subsample	0.51	0.57	0.84
	colsample_bytree	0.76	0.83	0.91
	min_child_weight	4.00	3.00	1.00
	scale_post_weight	6.00	9.00	7.00
	Gamma	0.02	0.05	0.05
	reg_lambda	797.32	790.44	447.25
	reg_alpha	56.21	121.55	279.99
Syn	max_depth	7.00	8.00	7.00
	learning_rate	0.03	0.10	0.02
	n_estimators	644.00	964.00	637.00
	Subsample	0.52	0.83	0.92
	colsample_bytree	0.58	0.94	0.79
	min_child_weight	9.00	8.00	1.00
	scale_post_weight	3.00	1.00	7.00
	Gamma	0.00	0.43	0.00
	reg_lambda	60.63	696.25	164.17
	reg_alpha	93.97	39.96	42.77

5. 4. Performance of tuned XGBoost models using confusion matrix-based metrics

Table 9 presents the results of performance evaluation based on XGBoost and a combination of XGBoost and Bayesian optimization methods for each dataset.

Table 9 presents the performance of various datasets utilizing XGBoost and XGBoost with Bayesian optimization, highlighting key metrics such as accuracy, recall, precision, specificity, and F-measure. The results indicate that Bayesian optimization generally enhances the performance of XGBoost across most datasets, with notable improvements in accuracy and recall, particularly in the Microcalcification and Dermatology datasets. For instance, the accuracy of XGBoost improved from 0.98 to 0.99 in the Microcalcification dataset when Bayesian optimization was applied. Conversely, some datasets, such as the Pima Indians Diabetes Database, exhibited less pronounced differences, suggesting that the effectiveness of Bayesian Optimization may vary depending on the specific characteristics of the dataset. Overall, the findings underscore the potential benefits of incorporating Bayesian Optimization to refine model performance in classification tasks.

Table 9

Performance of Tuned XGBoost Models

Data	Performance metrics	XGBoost			XGBoost+Bayesian optimization		
		SMOTE	ENN	SMOTE-ENN	SMOTE	ENN	SMOTE-ENN
1	2	3	4	5	6	7	8
Mic (n=11,183, feature=6, class=2, cv=52.59 %)	Accuracy	0.98	0.99	0.97	0.99	0.98	0.98
	Recall	0.88	0.89	0.92	0.74	0.82	0.80
	Precision	0.79	0.85	0.72	0.92	0.82	0.82
	Specificity	0.88	0.89	0.92	0.74	0.82	0.80
	F-measure	0.83	0.87	0.78	0.80	0.82	0.81

Continuation of Table 9

1	2	3	4	5	6	7	8
PID (n=768, feature=8, class=2, cv=42.29 %)	Accuracy	0.69	0.65	0.69	0.64	0.64	0.78
	Recall	0.68	0.69	0.72	0.50	0.50	0.74
	Precision	0.67	0.68	0.70	0.32	0.32	0.76
	Specificity	0.68	0.69	0.72	0.50	0.50	0.74
	F-measure	0.67	0.65	0.68	0.39	0.39	0.75
PMD (n=3,240, feature=16, class=2, cv=40.17 %)	Accuracy	0.95	0.95	0.92	0.16	0.95	0.84
	Recall	0.88	0.88	0.86	0.50	0.88	0.50
	Precision	0.92	0.94	0.85	0.08	0.94	0.42
	Specificity	0.88	0.88	0.86	0.50	0.88	0.50
	F-measure	0.90	0.91	0.85	0.14	0.91	0.46
Der (n=366, feature=34, class=6, cv=38.04 %)	Accuracy	0.97	0.99	0.99	0.42	0.42	0.42
	Recall	0.97	0.98	0.98	0.17	0.17	0.17
	Precision	0.96	0.98	0.98	0.07	0.07	0.07
	Specificity	0.99	1.00	1.00	0.83	0.83	0.83
	F-measure	0.96	0.98	0.98	0.10	0.10	0.10
BMI (n=500, feature=3, class=6, cv=40.00 %)	Accuracy	0.89	0.82	0.87	0.39	0.64	0.39
	Recall	0.89	0.83	0.91	0.17	0.35	0.17
	Precision	0.83	0.69	0.86	0.07	0.30	0.07
	Specificity	0.98	0.96	0.97	0.84	0.91	0.84
	F-measure	0.84	0.73	0.88	0.10	0.32	0.10
CDC (n=253,680, feature=22, class=3, cv=85.68 %)	Accuracy	0.65	0.84	0.53	0.85	0.85	0.85
	Recall	0.49	0.38	0.49	0.39	0.39	0.39
	Precision	0.43	0.50	0.43	0.47	0.47	0.47
	Specificity	0.79	0.71	0.80	0.72	0.72	0.72
	F-measure	0.42	0.39	0.37	0.41	0.40	0.40
SC (n=100,000, feature=18, class=3, cv=46.94 %)	Accuracy	0.98	0.96	0.97	0.97	0.97	0.96
	Recall	0.97	0.96	0.97	0.97	0.96	0.95
	Precision	0.97	0.96	0.96	0.97	0.97	0.97
	Specificity	0.99	0.98	0.98	0.98	0.98	0.98
	F-measure	0.97	0.96	0.97	0.97	0.96	0.96
Syn (n=150,000, feature=30, class=10, cv=14.64 %)	Accuracy	0.82	0.75	0.76	0.69	0.80	0.71
	Recall	0.82	0.75	0.76	0.69	0.80	0.71
	Precision	0.82	0.77	0.76	0.69	0.80	0.71
	Specificity	0.98	0.97	0.97	0.97	0.98	0.97
	F-measure	0.82	0.75	0.76	0.69	0.80	0.71

6. Discussion of the results impact of dataset complexity on Bayesian-optimized XGBoost for imbalanced class distribution

The findings from Tables 6–9 elucidate the effectiveness of various balancing techniques, the trade-offs associated with dataset complexity, and the role of Bayesian optimization in hyperparameter tuning. It was found that SMOTE significantly improves performance metrics, such as increasing accuracy from 0.98 to 0.99 in the Mic dataset, while the combined SMOTE-ENN approach performs best in high class overlap scenarios. However, SMOTE can introduce noise, and ENN may discard valuable instances, necessitating tailored techniques. The research also highlights that increasing feature dimensionality can diminish model performance, stressing the need for careful feature selection. Additionally, Bayesian optimization for hyperparameter tuning consistently outperforms default settings, reinforcing its importance in enhancing model efficacy. Overall, larger, richer datasets yield more reliable performance metrics but require sophisticated techniques to avoid overfitting.

The peculiarities of the proposed method lie in its systematic approach to hyperparameter tuning through Bayesian optimization, which contrasts with traditional methods such as grid search or random search that may not efficiently navigate the high-dimensional search space. The results align with previous studies, such as those by [19, 23, 25], which emphasize the advantages of Bayesian optimization in optimizing machine learning models. However, this study extends the existing literature by specifically addressing the challenges posed by imbalanced datasets and the complexities associated with them, as highlighted in the literature review.

This study introduces innovative solutions for addressing class imbalance in medical datasets, notably using SMOTE to generate synthetic samples for the minority class, which preserves dataset size and enhances learning. The combination of SMOTE with ENN (SMOTE-ENN) shows superior performance, particularly in high class overlap scenarios, while maintaining data integrity [26]. The research also examines the trade-offs of feature dimensionality and class overlap, addressing sparsity implications. Additionally, Bayesian optimization for hyperparameter tuning offers a more efficient alternative to traditional grid search methods [27]. Finally, the comprehensive evaluation of tuned XGBoost models using confusion matrix-based metrics highlights the importance of multiple performance metrics, enabling tailored strategies for diverse datasets.

This study addresses class imbalance in machine learning, particularly in medical datasets, by implementing innovative solutions that enhance model performance. Significant improvements are evident, such as an accuracy reaching 0.84 in the CDC dataset using the combined SMOTE-ENN approach. Key factors for these enhancements include the effective generation of synthetic samples through SMOTE, a nuanced understanding of dataset complexity that informs balancing techniques, and the use of Bayesian optimization for efficient hyperparameter

tuning. This comprehensive approach integrates resampling techniques with Bayesian optimization, allowing for tailored strategies that improve model robustness. Additionally, the study employs confusion matrix-based metrics for detailed performance assessment, filling a critical niche by providing targeted approaches for imbalanced datasets, thus offering valuable insights for predictive models in medical applications.

While SMOTE excels in generating synthetic samples, it can introduce noise, particularly in high-dimensional spaces, which may lead to overfitting. ENN, while effective in cleaning the dataset, may discard valuable instances, especially in complex datasets where class boundaries are not well-defined. The combined SMOTE-ENN approach mitigates some of these limitations by balancing the generation of synthetic samples with noise reduction, yet it may still struggle in datasets with high feature redundancy.

This study, while providing valuable insights into addressing class imbalance in machine learning, has several shortcomings that may hinder practical application. The findings are based on specific datasets (Mic and CDC), which may not generalize to other medical or real-world datasets. Additionally, the study acknowledges challenges in high-dimensional data but does not explore strategies to mitigate overfitting. Although Bayesian optimization is presented as an efficient hyperparameter tuning method, the associated computational costs are not addressed, potentially making it impractical in resource-constrained environments. Furthermore, the focus on SMOTE and ENN without comparing other emerging techniques may overlook more effective solutions, and the lack of a framework for adapting methods to varying dataset characteristics limits practical applicability.

This study can be developed by expanding the evaluation to include a broader variety of datasets to enhance generalizability, incorporating a comparative analysis of emerging techniques like Generative Adversarial Networks (GANs) [28] to identify more effective solutions, and addressing high-dimensional data challenges through dimensionality reduction and feature selection methods. Additionally, investigating the computational efficiency of Bayesian optimization for real-time applications is essential for practical implementation, while creating a framework to guide practitioners in adapting methods to varying dataset characteristics would improve applicability. These developments are crucial for ensuring that the proposed solutions are robust, effective, and accessible in real-world medical diagnostics.

The findings of this study are significant for applying machine learning techniques to imbalanced class distribution datasets in fields like healthcare, finance, and fraud detection. The resampling methods SMOTE, ENN, and SMOTE-ENN effectively improve class distribution and classification performance. Additionally, hyperparameter tuning through Bayesian Optimization is crucial for optimizing XGBoost models, with performance metrics such as accuracy, recall, precision, specificity, and F-measure providing a solid framework for evaluation.

However, the applicability of these results depends on specific conditions. Dataset characteristics, including size and class imbalance ratio, can influence the effectiveness of the techniques. While focused on XGBoost, the findings may not directly transfer to other models, requiring further validation. The implementation of Bayesian Optimization may also demand significant computational resources. Lastly, the context of the application can affect metric selection and result interpretation, emphasizing the importance of domain knowledge. Thus, while the insights are valuable, caution is warranted in generalizing them across all datasets.

7. Conclusions

1. The study demonstrates that SMOTE, ENN, and SMOTE-ENN effectively address class imbalance, with SMOTE-ENN achieving accuracy from 0.98 to 0.99 in the Mic dataset. However, SMOTE introduces noise, while ENN may remove informative instances. The SMOTE-ENN combination balances synthetic sample generation and noise reduction, showing superior performance, particularly in datasets with high class overlap.

2. Increasing feature dimensionality negatively affects model performance, as seen in the Microcalcification dataset (accuracy: 0.98, class overlap: 52.59 %) compared to the Pima Indians Diabetes dataset (accuracy: 0.69, class overlap: 42.29 %). Higher class overlap and sparsity lead to degraded classification results, with feature redundancy causing overfitting, exemplified by the Syn dataset (accuracy: 0.82, dimensionality: 30, class overlap: 14.64 %). Tailored preprocessing techniques, such as SMOTE and Bayesian Optimization, do not significantly enhance performance, as shown by the Microcalcification dataset's accuracy improvement from 0.98 to 0.99.

3. Bayesian optimization significantly enhances model performance, as evidenced by a reduction in cross-validation error from 0.98 to 0.99 in the Microcalcification dataset, while improving classification metrics such as recall and precision. This tuning method effectively mitigates overfitting, with the Pima Indians Diabetes dataset showing an increase in F-measure from 0.67 to 0.75. However, the computational cost associated with Bayesian optimization poses a challenge, necessitating further research to optimize its application in resource-limited environments.

4. The study employs accuracy, precision, specificity, recall, and F1-score to comprehensively assess model performance. The optimized XGBoost model achieves an accuracy of 0.84 in the CDC dataset using the SMOTE-ENN approach, confirming its effectiveness in handling imbalanced datasets. Sensitivity analyses show that dataset size and feature richness significantly influence classification outcomes, reinforcing the importance of dataset adaptation in predictive modeling.

Conflict of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship, or otherwise, that could affect the research and its results presented in this paper.

Financing

The study was conducted without financial support.

Data availability

The manuscript has associated data in a data repository.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the current work.

References

1. Ghosh, K., Bellinger, C., Corizzo, R., Branco, P., Krawczyk, B., Japkowicz, N. (2022). The class imbalance problem in deep learning. Machine Learning, 113 (7), 4845–4901. https://doi.org/10.1007/s10994-022-06268-8
2. Rezvani, S., Wang, X. (2023). A broad review on class imbalance learning techniques. Applied Soft Computing, 143, 110415. https://doi.org/10.1016/j.asoc.2023.110415
3. Moniz, N., Cerqueira, V. (2021). Automated imbalanced classification via meta-learning. Expert Systems with Applications, 178, 115011. https://doi.org/10.1016/j.eswa.2021.115011
4. Magris, M., Iosifidis, A. (2023). Bayesian learning for neural networks: an algorithmic survey. Artificial Intelligence Review, 56 (10), 11773–11823. https://doi.org/10.1007/s10462-023-10443-1
5. Pereira, R. M., Costa, Y. M. G., Silla, C. N. (2021). Handling imbalance in hierarchical classification problems using local classifiers approaches. Data Mining and Knowledge Discovery, 35 (4), 1564–1621. https://doi.org/10.1007/s10618-021-00762-8
6. Ye, H.-J., Chen, H.-Y., Zhan, D.-C., Chao, W.-L. (2020). Identifying and Compensating for Feature Deviation in Imbalanced Deep Learning. arXiv. https://doi.org/10.48550/arXiv.2001.01385
7. Guan, S., Fu, N. (2022). Class imbalance learning with Bayesian optimization applied in drug discovery. Scientific Reports, 12 (1). https://doi.org/10.1038/s41598-022-05717-7
8. Banchhor, C., Srinivasu, N. (2021). Analysis of Bayesian optimization algorithms for big data classification based on Map Reduce framework. Journal of Big Data, 8 (1). https://doi.org/10.1186/s40537-021-00464-4
9. Albahli, S. (2023). Efficient hyperparameter tuning for predicting student performance with Bayesian optimization. Multimedia Tools and Applications, 83 (17), 52711–52735. https://doi.org/10.1007/s11042-023-17525-w
10. Wang, X., Jin, Y., Schmitt, S., Olhofer, M. (2023). Recent Advances in Bayesian Optimization. ACM Computing Surveys, 55 (13s), 1–36. https://doi.org/10.1145/3582078
11. Nayak, J., Naik, B., Dash, P. B., Vimal, S., Kadry, S. (2022). Hybrid Bayesian optimization hypertuned catboost approach for malicious access and anomaly detection in IoT nomalyframework. Sustainable Computing: Informatics and Systems, 36, 100805. https://doi.org/10.1016/j.suscom.2022.100805
12. Guembe, B., Misra, S., Azeta, A. (2024). Federated Bayesian optimization XGBoost model for cyberattack detection in internet of medical things. Journal of Parallel and Distributed Computing, 193, 104964. https://doi.org/10.1016/j.jpdc.2024.104964
13. Yan, S., Zhao, Z., Liu, S., Zhou, M. (2024). BO-SMOTE: A Novel Bayesian-Optimization-Based Synthetic Minority Oversampling Technique. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 54 (4), 2079–2091. https://doi.org/10.1109/tsmc.2023.3335241
14. Davis, D., Rodriguez, L. (2024). Enhancing algorithm interpretability and accuracy with borderline-SMOTE and Bayesian optimization. Journal of Computer Technology and Software, 3 (2), 6–13. Available at: https://ashpress.org/index.php/jcts/article/view/24
15. Bates, J. E., Shepard, H. K. (1993). Measuring complexity using information fluctuation. Physics Letters A, 172 (6), 416–425. https://doi.org/10.1016/0375-9601(93)90232-o
16. Wang, L. (2022). Imbalanced credit risk prediction based on SMOTE and multi-kernel FCM improved by particle swarm optimization. Applied Soft Computing, 114, 108153. https://doi.org/10.1016/j.asoc.2021.108153
17. Wilson, D. R., Martinez, T. R. (2000). Reduction Techniques for Instance-Based Learning Algorithms. Machine Learning, 38, 257–286. https://doi.org/10.1023/a:1007626913721
18. Yang, F., Wang, K., Sun, L., Zhai, M., Song, J., Wang, H. (2022). A hybrid sampling algorithm combining synthetic minority over-sampling technique and edited nearest neighbor for missed abortion diagnosis. BMC Medical Informatics and Decision Making, 22 (1). https://doi.org/10.1186/s12911-022-02075-2
19. Snoek, J., Larochelle, H., Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. arXiv. https://doi.org/10.48550/arXiv.1206.2944
20. Frazier, P. I. (2018). A Tutorial on Bayesian Optimization. arXiv. https://doi.org/10.48550/arXiv.1807.02811
21. Klein, A., Falkner, S., Bartels, S., Hennig, P., Hutter, F. (2017). Fast Bayesian optimization of machine learning hyperparameters on large datasets. arXiv. https://doi.org/10.48550/arXiv.1605.07079
22. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M. (2019). Optuna. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2623–2631. https://doi.org/10.1145/3292500.3330701
23. Chen, T., Guestrin, C. (2016). XGBoost. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794. https://doi.org/10.1145/2939672.2939785
24. Szeghalmy, S., Fazekas, A. (2023). A Comparative Study of the Use of Stratified Cross-Validation and Distribution-Balanced Stratified Cross-Validation in Imbalanced Learning. Sensors, 23 (4), 2333. https://doi.org/10.3390/s23042333
25. Vujovic, Ž. Đ. (2021). Classification Model Evaluation Metrics. International Journal of Advanced Computer Science and Applications, 12 (6). https://doi.org/10.14569/ijacsa.2021.0120670
26. Salmi, M., Atif, D., Oliva, D., Abraham, A., Ventura, S. (2024). Handling imbalanced medical datasets: review of a decade of research. Artificial Intelligence Review, 57 (10). https://doi.org/10.1007/s10462-024-10884-2
27. Khadka, K., Chandrasekaran, J., Lei, Y., Kacker, R. N., Kuhn, D. R. (2024). A Combinatorial Approach to Hyperparameter Optimization. Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI, 140–149. https://doi.org/10.1145/3644815.3644941
28. Chai, P., Hou, L., Zhang, G., Tushar, Q., Zou, Y. (2024). Generative adversarial networks in construction applications. Automation in Construction, 159, 105265. https://doi.org/10.1016/j.autcon.2024.105265