technologies

MDPI

Article

Comprehensive Analysis of Random Forest and XGBoost Performance with SMOTE, ADASYN, and GNUS Under Varying Imbalance Levels

Mehdi Imani [1], Ali Beikmohammadi [1,*] and Hamid Reza Arabnia [2,*]

[1] Department of Computer and Systems Sciences, Stockholm University, SE-16455 Stockholm, Sweden; m.imani@gmail.com
[2] School of Computing, University of Georgia, Athens, GA 30602, USA
* Correspondence: beikmohammadi@dsv.su.se (A.B.); hra@uga.edu (H.R.A.)

Abstract: This study examines the efficacy of Random Forest and XGBoost classifiers in conjunction with three upsampling techniques—SMOTE, ADASYN, and Gaussian noise upsampling (GNUS)—across datasets with varying class imbalance levels, ranging from moderate to extreme (15% to 1% churn rate). Employing metrics such as F1 score, ROC AUC, PR AUC, Matthews Correlation Coefficient (MCC), and Cohen's Kappa, this research provides a comprehensive evaluation of classifier performance under different imbalance scenarios, focusing on applications in the telecommunications domain. The findings highlight that tuned XGBoost paired with SMOTE (Tuned_XGB_SMOTE) consistently achieves the highest F1 score and robust performance across all imbalance levels. SMOTE emerged as the most effective upsampling method, particularly when used with XGBoost, whereas Random Forest performed poorly under severe imbalance. ADASYN showed moderate effectiveness with XGBoost but underperformed with Random Forest, and GNUS produced inconsistent results. This study underscores the impact of data imbalance, with MCC, Kappa, and F1 scores fluctuating significantly, whereas ROC AUC and PR AUC remained relatively stable. Moreover, rigorous statistical analyses employing the Friedman test and Nemenyi post hoc comparisons confirmed that the observed improvements in F1 score, PR-AUC, Kappa, and MCC were statistically significant ($p < 0.05$), with Tuned_XGB_SMOTE significantly outperforming Tuned_RF_GNUS. While differences in ROC-AUC were not significant, the consistency of these results across multiple performance metrics underscores the reliability of our framework, offering a statistically validated and attractive solution for model selection in imbalanced classification scenarios.

Keywords: customer churn prediction; different imbalance rates; XGBoost; random forest; upsampling techniques

check for updates

Academic Editor: Mohammed Mahmoud

Received: 29 January 2025
Revised: 15 February 2025
Accepted: 18 February 2025
Published: 20 February 2025

Citation: Imani, M.; Beikmohammadi, A.; Arabnia, H.R. Comprehensive Analysis of Random Forest and XGBoost Performance with SMOTE, ADASYN, and GNUS Under Varying Imbalance Levels. *Technologies* **2025**, *13*, 88. https://doi.org/10.3390/technologies13030088

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Predicting and mitigating customer churn have become critical challenges in the dynamic realm of customer-centric industries. Customer churn, defined as discontinuing a customer's association with a product or service, significantly affects a company's revenue and growth potential. Given the higher costs associated with acquiring new customers than retaining existing ones, businesses increasingly rely on machine learning (ML) models to develop robust churn prediction systems. These models enable the identification of at-risk customers and the facilitation of targeted interventions to improve retention and loyalty.

This challenge spans multiple industries, including banking, insurance, and telecommunications. Within the telecommunications sector, a cornerstone of global socioeconomic development and a significant contributor to revenue generation, the financial scale reaches approximately USD 4.7 trillion annually [1,2]. The sector faces intense competition as companies strive to grow their market share by expanding customer bases. Customer retention has become a primary goal of customer relationship management (CRM) initiatives, as retaining customers is up to 20 times more cost-effective than acquiring new ones [1]. Consequently, minimizing customer churn is essential for optimizing revenue and reducing marketing and advertising expenditures.

The application of ML models in churn prediction has garnered substantial attention, addressing the binary classification task of identifying churn and non-churn customers. Popular models include Decision Trees [3–11], Artificial Neural Networks [3,4,10–12], Random Forests [13,14], Support Vector Machines [11], and advanced boosting techniques such as eXtreme Gradient Boosting (XGBoost) [15], Light Gradient Boosting Machine (LightGBM) [16], and Categorical Boosting (CatBoost) [17]. Each model exhibits distinct strengths and limitations, influenced by factors such as dataset characteristics, customer behavior complexity, and specific prediction objectives [18,19].

A major challenge in churn prediction arises from class imbalance, where non-churned customers vastly outnumber churned customers. This imbalance skews model performance and affects predictive accuracy. To address this, various resampling techniques have been applied, including Synthetic Minority Oversampling Technique (SMOTE) [20], Adaptive Synthetic Sampling (ADASYN) [21], Gaussian noise upsampling (GNUS) [22], and hybrid methods such as SMOTE combined with Tomek Links [23] or Edited Nearest Neighbors (ENNs) [24]. These techniques aim to balance datasets and improve model efficacy [25,26]. Furthermore, hyperparameter optimization techniques such as Grid Search, random search, and more advanced methods like Bayesian Optimization, Genetic Algorithms, and hyperparameter optimization are critical for enhancing model performance [27,28].

Ensemble methods, particularly boosting and bagging algorithms, have proven highly effective in tackling classification problems [29–33]. While numerous studies have reviewed machine learning techniques for imbalanced datasets [25,34–40], none have examined their performance across datasets with varying levels of imbalance, ranging from moderate to extreme. This study fills that gap by evaluating the performance of supervised classifiers, specifically Random Forest and XGBoost, combined with oversampling techniques such as SMOTE, ADASYN, and GNUS. The analysis spans datasets with churn rates ranging from 15% to 1%. To further improve predictive accuracy, Grid Search is employed for hyperparameter optimization in these imbalanced scenarios.

This research contributes significantly to customer churn prediction by comprehensively evaluating various supervised ML classifiers, specifically Random Forests and XGBoost. This study uniquely focuses on integrating these classifiers with advanced oversampling techniques such as SMOTE, ADASYN, and GNUS while leveraging Grid Search for hyperparameter optimization. This multifaceted analytical approach is designed to improve churn prediction accuracy, particularly in scenarios where datasets exhibit high levels of class imbalance—a common and critical challenge in real-world applications.

The research fills an essential gap in the existing literature, which, while extensive in its exploration of customer churn prediction, lacks a detailed analysis of how specific ML classifiers like XGBoost and Random Forest perform under varying degrees of dataset imbalance. Although sampling methods such as SMOTE, ADASYN, and GNUS are widely employed, there remains a limited understanding of their interaction with these classifiers, particularly across datasets with different imbalance ratios. Furthermore, most prior studies

have not adequately addressed the influence of dataset imbalance on predictive accuracy, nor have they thoroughly examined the robustness of these classifiers when combined with advanced sampling techniques and optimized hyperparameters.

By systematically investigating these aspects, this study aims to bridge the knowledge gap and offer actionable insights into the performance dynamics of these models. The research evaluates the behavior of XGBoost and Random Forest in conjunction with sampling methods across a spectrum of imbalance scenarios, ranging from moderate to extreme. This detailed examination underscores the importance of dataset characteristics in determining model efficacy and highlights the critical role of oversampling techniques and hyperparameter tuning in achieving accurate and reliable predictions.

This study focuses on Random Forest and XGBoost due to their well-established performance in customer churn prediction and imbalanced classification. The rationale for selecting these two models is based on the following considerations:

- Established Benchmark Models: Random Forest and XGBoost have been extensively utilized in customer churn prediction and imbalanced classification tasks due to their interpretability, robustness, and efficiency. Numerous prior studies, including our recent work [39], have demonstrated their effectiveness in handling structured tabular data, making them strong benchmark models for this study. Our previous study [39], conducted on the same dataset, revealed that Random Forest outperforms several traditional machine learning models, including Artificial Neural Networks, Decision Trees, Support Vector Machines, and Logistic Regression. Additionally, XGBoost demonstrated superior performance compared to other boosting-based models, such as LightGBM and CatBoost. These findings reinforce the suitability of Random Forest and XGBoost as the primary models for this research.
- Computational Efficiency: While deep learning models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have shown promise in various predictive tasks, they typically require significantly larger datasets, greater computational resources, and extensive hyperparameter tuning to achieve optimal performance. Given the customer churn dataset's structured nature and relatively limited size, tree-based ensemble methods such as Random Forest and XGBoost offer a more efficient and practical alternative. Their ability to handle imbalanced datasets with effective sampling techniques makes them well suited for this study.

By selecting Random Forest and XGBoost, this research ensures a focused and comprehensive evaluation of widely recognized and computationally efficient models in the context of customer churn prediction under varying levels of class imbalance.

The contributions of this paper are twofold. First, it provides a nuanced understanding of the interaction between ML classifiers, oversampling techniques, and dataset imbalance, which has implications for designing more effective churn prediction models. Second, it delivers a practical framework for practitioners in the telecommunications industry, enabling them to develop predictive systems that are better equipped to handle the challenges posed by highly imbalanced datasets. This research supports the broader goal of improving customer retention strategies, reducing churn rates, and fostering sustainable growth within subscription-based businesses.

While this study focuses on customer churn prediction in the telecommunications industry, class imbalance is a widespread issue across multiple domains. In healthcare, rare disease detection often suffers from extreme class imbalance, making it difficult for models to identify positive cases. Similarly, in fraud detection, fraudulent transactions constitute a small fraction of overall transactions, requiring effective oversampling techniques to enhance detection accuracy. Predicting credit defaults in finance also involves imbalanced datasets, where misclassifying minority cases can lead to financial risk. Addressing class

imbalance in these fields is critical. While our study evaluates and compares multiple resampling techniques, including SMOTE, ADASYN, and the proposed GNUS within a telecommunications context, the insights gained from these methods have the potential to be applied across various domains facing similar data challenges.

The structure of this paper is organized as follows: Section 2 provides an overview of ensemble machine learning techniques, imbalanced data challenges, sampling techniques, hyperparameter optimization, and evaluation metrics. Section 3 describes the dataset and outlines the data preprocessing steps. Section 4 presents a detailed discussion of the methodological framework, including data splitting, feature selection, and creating highly imbalanced datasets using random undersampling and KMeans clustering approaches. Section 5 analyzes the experimental results, evaluating the performance of different models across four datasets with varying levels of imbalance. This section also includes statistical tests, such as the Friedman and Nemenyi tests, followed by an overall comparison, performance analysis, and summary of findings. Section 6 outlines the study's conclusions, while Section 7 discusses current challenges and potential directions for future research.

2. Background

2.1. Ensemble Machine Learning Techniques

Ensemble learning, a cornerstone technique within the ML landscape, synthesizes the predictions from multiple learning algorithms to form a more potent composite model. This methodology leverages the collective strengths of various base learners, often simple or weak models, to construct an ensemble that surpasses the performance of any single model in isolation [41]. The foundational premise of ensemble learning is rooted in the notion that a diverse array of models, when judiciously combined, can compensate for individual weaknesses, leading to superior overall predictive capabilities [42,43].

Contemporary research has firmly established ensemble learning as a quintessential approach for elevating the precision of predictive models in ML. Among its methodologies are bagging (bootstrap aggregating), such as Random Forest, and boosting, such as XGBoost, which stand out as two principal strategies, each designed to bolster model accuracy through distinct mechanisms [29,30]. Bagging aims to enhance model stability and accuracy by generating multiple subsets of data from the original dataset through bootstrapping. These subsets are then used to train various models whose predictions are aggregated to produce a final decision, typically through majority voting for classification or averaging for regression tasks [44].

Boosting, on the other hand, sequentially trains models, where each subsequent model attempts to correct the errors of its predecessor. This technique incrementally builds a strong classifier from a series of weak ones by focusing more heavily on previously misclassified instances, thus improving the model's accuracy on challenging cases [45]. Notable examples of boosting algorithms include AdaBoost (Adaptive Boosting) and Gradient Boosting machines, such as XGBoost, which have been widely acknowledged for their effectiveness in various predictive tasks across different domains [46–48].

The efficacy of ensemble methods extends beyond mere accuracy improvements. These techniques have also significantly reduced overfitting, a common pitfall in ML models, by introducing model diversity and focusing on robust generalization [49]. Furthermore, ensemble learning is adaptable to classification and regression problems, making it a versatile tool in the ML toolkit.

The strategic application of ensemble methods, particularly bagging and boosting, has led to notable advancements in predictive modeling, with numerous studies corroborating their effectiveness across various applications, from bioinformatics to financial forecasting [50,51]. As ML continues to evolve, exploring novel ensemble techniques and

their integration with emerging models and algorithms remains a vibrant area of research, promising further enhancements in predictive performance and model robustness.

2.1.1. Random Forest

The framework of Random Forest, initially conceptualized by Ho in 1995 [13], has undergone continual refinement and development, significantly propelled by Leo Breiman's contributions in 2001 [14]. As a sophisticated ensemble learning methodology, Random Forest leverages an extensive collection of Decision Trees for performing classification tasks, with its predictive outcome derived from the consensus across the multitude of trees. Typically, Random Forests are recognized for their enhanced performance metrics compared to singular Decision Tree models. However, the specific attributes of the dataset can markedly influence the efficacy of these forests in question.

Incorporating the bagging (Bootstrap Aggregating) strategy within its algorithmic structure, Random Forests executes this by generating multiple subsets from the original training set $TS_n = \{(x_1.y_1).\cdots.(x_n.y_n)\}$ through bootstrap sampling. This process is iterated B times, with each cycle selecting a bootstrap sample from TS_n and employing these samples to train distinct Decision Trees:

- From the training set, n examples, denoted as X_b, Y_b, are sampled with replacement.
- A classification tree pertinent to churn prediction scenarios, fb, is trained to utilize the bootstrap samples X_b, Y_b.
- For predictions on new instances x', Random Forest aggregates the decisions from all trained trees, with the final prediction, $\hat{f}$, determined by averaging the votes across B trees for classification:

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x') \tag{1}$$

Further enhancements and empirical studies have underscored Random Forest's robustness and versatility across diverse application domains, ranging from bioinformatics to financial analysis, where it has been particularly noted for its ability to handle high-dimensional data effectively while maintaining resistance to overfitting [52,53]. Moreover, recent innovations have explored modifications in tree construction and feature selection methods within the Random Forest algorithm, aiming to optimize performance further and reduce computational complexity [54,55].

These advancements affirm the significant role of Random Forest within the ensemble learning sphere, presenting it as a powerful tool for complex classification challenges, including but not limited to customer churn prediction, disease diagnosis, and financial risk assessment.

2.1.2. XGBoost

XGBoost, introduced by Chen and Guestrin [15], signifies a significant advancement in Gradient Boosting techniques, delivering exceptional flexibility, scalability, and computational efficiency. One of its key innovations is a novel regularization approach that reduces overfitting and improves the model's generalization ability. This approach incorporates an additional term into the loss function, effectively managing model complexity and enhancing robustness. Furthermore, the gain function in XGBoost optimizes the selection of splitting points within Decision Trees, thereby strengthening the model's predictive accuracy and overall efficiency. Widely validated across diverse domains, XGBoost consistently

demonstrates superior performance, representing a major milestone in gradient boosting methodologies [56]. Its objective function is defined as

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l\left(y_i, y_i'\right) + \sum_{k=1}^{k} \Omega(f_k) \tag{2}$$

where $l\left(y_i, y_i'\right)$ is the loss function measuring the difference between actual (y_i) and predicted (y_i') values, and $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \mathcal{W}_j^2$ is a regularization term penalizing model complexity (T: number of leaves, $\mathcal{W}_j$: leaf weights, γ, λ: regularization parameters). Predictions are made by summing the outputs of all K trees:

$$y_i' = \sum_{k=1}^{k} f_k(x_i) \tag{3}$$

XGBoost employs gradient-based optimization, using the first (g_i) and second (h_i) derivatives of the loss function with respect to the predictions:

$$g_i = \frac{\partial l\left(y_i,\ y_i'\right)}{\partial y_i'} h_i = \frac{\partial^2 l\left(y_i, y_i'\right)}{\partial {y_i'}^2} \tag{4}$$

To improve robustness and prevent overfitting, XGBoost incorporates regularization, shrinkage, and hyperparameter tuning. This framework enables efficient handling of high-dimensional and imbalanced datasets, making it particularly effective for predictive tasks such as churn prediction.

2.2. Imbalanced Data and Sampling Techniques

Imbalanced datasets represent a widespread challenge within data mining, mainly manifesting in binary classification tasks where a significant disparity exists between the number of instances in the majority and minority classes. This discrepancy can lead to an imbalanced ratio ranging dramatically from 1:2 to an extreme of 1:1000. In our study, we analyzed datasets with varying levels of imbalance, showing a dominant majority class (non-churned) over the minority class (churned). The imbalances are 85% non-churned to 15% churned in dataset 1 (DS1), 90% non-churned to 10% churned in dataset 2 (DS2), 95% non-churned to 5% churned in dataset 3 (DS3), and 99% non-churned to 1% churned in dataset 4 (DS4).

Such an imbalance inherently predisposes the development of a classifier that, while demonstrating high predictive accuracy for the majority class, unfortunately, yields poor performance when predicting the minority class outcomes. To mitigate these imbalances and foster a more equitable classifier performance across classes, the field has introduced a variety of sampling strategies. These strategies aim to modify the class distribution towards a more balanced state, thereby enhancing the fairness and effectiveness of the predictive model. Sampling techniques broadly fall into two primary categories: undersampling, which reduces the size of the majority class, and oversampling, which increases the size of the minority class, each to achieve a more balanced dataset [57].

Further exploration into these techniques reveals advanced methods such as SMOTE, ADASYN, and GNUS and the combination of oversampling with undersampling strategies to balance the dataset and introduce more diversity and representativeness in the sample [20,21]. These methods have shown promise in improving model sensitivity towards the minority class without compromising the overall accuracy. Moreover, integrating these sampling methods with sophisticated ML algorithms has been a subject of ongoing research, aiming to optimize the balance between recall and precision in predictive modeling [58,59].

The intricate challenge of handling imbalanced datasets necessitates a nuanced understanding of the impact of class distribution on model performance. It underscores the importance of selecting appropriate sampling techniques tailored to the specific characteristics of the dataset and the predictive task at hand. As such, the ongoing development and evaluation of novel sampling methodologies continue to be a crucial area of research, with significant implications for enhancing the robustness and fairness of ML models across various domains.

2.2.1. SMOTE

SMOTE, introduced by Chawla et al. in 2002 [20], is a prominent technique developed to address the issue of imbalanced datasets, a frequent challenge in data mining and machine learning. Imbalanced datasets occur when one class significantly outnumbers another, often leading to models biased towards the majority class. This imbalance is particularly problematic in applications like fraud detection and disease diagnosis, where accurately identifying the minority class is of critical importance.

SMOTE improves dataset balance by generating synthetic instances for the minority class rather than duplicating existing ones. Unlike traditional oversampling, SMOTE employs interpolation, creating new samples by selecting a minority class instance and its nearest neighbors to generate variations within their feature space. This process increases the count and diversity of minority class instances, reducing the risk of overfitting associated with simple replication, as illustrated in the pseudocode below.

The innovation of SMOTE lies in its ability to expand the decision boundary of the minority class by synthesizing new instances, thereby broadening the decision area and enhancing the model's adaptability to data variations. This improved sensitivity to the minority class is particularly valuable when identifying rare yet critical events is essential.

Building on the foundational work of SMOTE, subsequent research has introduced numerous variants and adaptations to address specific data characteristics and challenges. For instance, Borderline-SMOTE [60] creates synthetic samples near the decision boundary, where classification uncertainty is highest. This targeted approach aims to improve the classifier's accuracy in regions where minority class instances are most susceptible to misclassification. Similarly, methods like Safe-Level-SMOTE and ADASYN [21] adaptively generate synthetic samples based on the data distribution, ensuring that the new instances are representative and informative.

Despite its many advantages, SMOTE is not without its limitations. Its effectiveness can depend on the underlying data distribution, and in some instances, it may introduce noise, particularly in datasets with scattered minority class instances or significant noise and outliers. To fully leverage SMOTE's benefits, careful preprocessing and consideration of the dataset's characteristics are often necessary.

2.2.2. ADASYN

ADASYN, introduced by He et al. in 2008 [21], represents a significant advancement in addressing class imbalance in machine learning and data mining. Building upon the foundation of SMOTE [5], ADASYN enhances the generation of synthetic samples for the minority class by adaptively focusing on regions where the classifier struggles due to class imbalance. Its primary objective is to dynamically adjust the data distribution by generating synthetic samples in areas with higher classification difficulty, thus improving the learning process.

The key innovation of ADASYN lies in its weighting mechanism, which determines the number of synthetic samples to generate for each minority instance based on its learning difficulty. This difficulty is quantified by the density of majority class instances sur-

rounding a minority instance; the more a minority instance is encircled by majority class instances, the greater the number of synthetic samples generated. By concentrating on the more challenging regions, ADASYN ensures that the synthetic instances significantly enhance the decision boundary's adaptability to the minority class, as illustrated in the pseudocode below.

This targeted approach introduces a dynamic learning landscape, enabling the classifier to better handle class imbalances by emphasizing areas where improvements are most needed. ADASYN's adaptivity helps achieve a more balanced class distribution and bolsters the classifier's ability to generalize over the minority class, improving predictive accuracy in highly imbalanced datasets.

The versatility and efficacy of ADASYN have been demonstrated across various domains, highlighting its value in enhancing predictive performance in imbalanced scenarios. Unlike static oversampling methods, ADASYN fine-tunes synthetic sample generation based on the local learning difficulty, offering a more dynamic and effective approach to dataset balancing.

However, as with other oversampling techniques, careful implementation of ADASYN is crucial to avoid potential drawbacks. Improper parameter tuning or reliance on local neighborhood density can sometimes result in synthetic samples being generated in suboptimal regions, particularly in datasets with complex distributions or significant class overlaps. To maximize its benefits, thoughtful preprocessing and parameter adjustment are essential, ensuring that the method effectively addresses imbalance without introducing excessive noise into the dataset.

2.2.3. GNUS

GNUS is an advanced method designed to address the pervasive class imbalance issue in machine learning datasets [22]. Class imbalance occurs when certain classes, typically the minority classes, are significantly underrepresented, leading to biased predictions favoring the majority class. While traditional oversampling methods like SMOTE and ADASYN generate synthetic samples to balance the class distribution, these techniques can sometimes result in overfitting and fail to introduce sufficient variability in the synthetic data.

GNUS seeks to overcome these limitations by introducing Gaussian noise into the feature space of minority class instances. This process generates synthetic samples that incorporate variability, enhancing the diversity of the minority class without merely replicating existing characteristics. Gaussian noise, characterized by its normal distribution, adds subtle but meaningful variations to minority instances, thus creating new data points that better capture the complexity of the underlying data distribution. By doing so, GNUS helps mitigate overfitting and improves the model's ability to generalize to unseen data.

Research has demonstrated the effectiveness of Gaussian noise in enhancing machine learning models across various contexts. For example, Salakhutdinov and Hinton [61] highlighted the utility of Gaussian noise in improving the performance of Restricted Boltzmann Machines (RBMs) in their groundbreaking work on deep learning. Similarly, in data augmentation, Shorten and Khoshgoftaar [62] emphasized how noise injection techniques, including Gaussian noise, contribute to model robustness by diversifying training data. By leveraging Gaussian noise, GNUS offers a promising approach for tackling class imbalance while maintaining model robustness and generalization, making it a valuable addition to the repertoire of techniques for handling imbalanced datasets.

While traditional resampling techniques like SMOTE and ADASYN rely on interpolation or density-based adjustments to generate synthetic samples, Gaussian noise upsampling takes a different approach. Instead of interpolating between existing minority class

instances, GNUS introduces controlled random perturbations, ensuring that the synthetic samples remain realistic while enhancing the model's generalization capabilities [22].

Unlike SMOTE, which creates synthetic samples by interpolating between minority class instances, GNUS applies stochastic noise directly to existing data points. This approach reduces the risk of overfitting to local structures, making the model more robust to variations in real-world data. Compared to ADASYN, which dynamically adjusts the number of synthetic samples based on local density, GNUS maintains a uniform distribution, ensuring that all minority class instances receive an equal degree of augmentation.

A key challenge in oversampling is class separability—introducing too many synthetic samples too close to decision boundaries can blur distinctions between classes, reducing classification effectiveness. GNUS mitigates this issue by preserving the original feature distribution while incorporating sufficient diversity, allowing classifiers to generalize better without merely memorizing oversampled patterns [63].

Another advantage of GNUS is its computational efficiency. Unlike SMOTE, which requires nearest neighbor calculations, or ADASYN, which depends on density estimations, GNUS generates noise and applies it directly to existing data points. This simplicity makes it suitable for large-scale datasets, where traditional upsampling techniques may become computationally expensive.

The GNUS process in this study begins by identifying the minority class, isolating its instances, and determining the range of each feature to prevent unrealistic variations. Gaussian noise is generated with a standard deviation set to 10% of the feature range. This ensures that the synthetic samples remain realistic and within reasonable bounds. These synthetic data points are formed by adding this noise to the minority class instances, thereby introducing subtle yet meaningful variations. Any feature values that exceed the original range are clipped to maintain data integrity. Finally, the upsampled dataset is shuffled to avoid ordering bias before being used for model training.

By striking a balance between data augmentation and computational feasibility, GNUS offers a lightweight yet effective alternative to traditional resampling methods, making it a valuable tool for handling imbalanced datasets in machine learning applications.

2.3. Hyperparameter Optimization

Hyperparameter tuning, or hyperparameter optimization, is a critical step in machine learning model development, as it involves refining the parameters that govern the learning process and model structure. Proper hyperparameter optimization is essential for maximizing model performance, as these parameters significantly impact the algorithm's efficiency and predictive accuracy. Among the various techniques available, Random Grid Search and OPTUNA are two widely used methods for systematically exploring the hyperparameter space and identifying optimal configurations.

Random Grid Search is a relatively simple yet effective approach that randomly samples a subset of hyperparameter combinations from a predefined grid, reducing computational overhead while maintaining a high probability of identifying well-performing configurations. Unlike exhaustive Grid Search, which evaluates all possible combinations, Random Grid Search focuses on a randomly selected subset, allowing for the efficient coverage of the hyperparameter space with fewer computational demands [64]. As outlined by Bergstra and Bengio (2012) [64], this method is particularly effective in cases where specific hyperparameters have a higher sensitivity to model performance, as it allows for an exploratory yet computationally feasible approach.

In contrast, OPTUNA is a more sophisticated optimization framework that leverages Bayesian Optimization and Tree-structured Parzen Estimators (TPEs) to iteratively refine the search process based on prior evaluations [65,66]. This allows it to focus on promis-

ing hyperparameter regions rather than exploring the space randomly. While Bayesian Optimization is highly efficient in high-dimensional hyperparameter spaces, its computational cost is significantly higher, particularly for complex models like XGBoost and Random Forest.

Both methods offer distinct advantages, with OPTUNA excelling in cases with extensive computational resources and Random Grid Search providing a lightweight yet effective alternative. Given the relatively small dataset in this study (fewer than 5000 records and 19 features), the benefits of Bayesian Optimization were expected to be marginal compared to the increased computational complexity required for training each iteration. As such, we opted for Random Grid Search, balancing computational efficiency and effectiveness in optimizing hyperparameters for tree-based models.

Another consideration in hyperparameter tuning is the impact of class imbalance. Bayesian Optimization relies on surrogate models that approximate the objective function, which may be less reliable when the minority class has limited representation. In contrast, Random Grid Search explores a more diverse range of hyperparameter configurations, increasing the probability of identifying an effective set even in scenarios where the majority class dominates the learning process. Our experimental results confirmed that Random Grid Search effectively optimized XGBoost and Random Forest while reducing computational demands. Furthermore, our findings demonstrated that resampling techniques (SMOTE, ADASYN, and GNUS) had a greater impact on model performance than hyperparameter tuning refinements, reinforcing our decision to focus on efficient data balancing strategies rather than an exhaustive hyperparameter search.

Future research could explore Bayesian Optimization in larger-scale datasets or computationally intensive deep learning models, where its ability to refine complex search spaces may be more pronounced. However, within the scope of this study, Random Grid Search provided an effective balance between computational feasibility and model optimization, ensuring robust performance while addressing class imbalance challenges.

2.4. Evaluation Metrics

In evaluating ML models, particularly classifiers, a comprehensive assessment incorporating both threshold and ranking metrics is essential for a thorough performance analysis. Threshold metrics focus on minimizing error rates by quantifying the discrepancies between predicted and actual class labels. These metrics are crucial for determining how well a classifier makes definitive predictions at a specific decision threshold.

On the other hand, ranking metrics evaluate a classifier's ability to distinguish between classes by analyzing the scores or probabilities it assigns to instances. These metrics require the classifier to output a continuous score indicating the likelihood of class membership. Ranking metrics provide insights into the model's capacity to consistently prioritize positive instances over negative ones by examining classifier performance across varying decision thresholds.

A high-performing classifier is expected to demonstrate robustness across a range of thresholds, maintaining strong class differentiation. Such consistency minimizes classification errors and results in higher rankings in performance evaluations, reflecting the model's effectiveness in distinguishing between classes and supporting decision-making in practical applications.

2.4.1. Threshold Metrics

Accuracy has traditionally been the default metric for evaluating ML models, but its reliability diminishes with imbalanced datasets. This is because accuracy primarily reflects the performance of the majority class, often neglecting the minority class's predictive

success. Accuracy can present an overly optimistic view of performance in applications involving rare events, such as fraud detection or disease outbreak prediction. To address this, metrics like recall, precision, and the F-measure are indispensable for a more balanced evaluation, particularly for the minority class [67]. The confusion matrix, detailing true positives (TPs), false positives (FPs), false negatives (FNs), and true negatives (TNs), provides crucial insights into predictive accuracy across classes, highlighting areas for improvement in class-specific performance.

Precision, recall, and accuracy, as derived from Table 1, provide a comprehensive evaluation of the model's performance. Precision measures the quality of positive class predictions, recall assesses their completeness, and accuracy reflects the overall correctness of the model's predictions:

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$Accuracy = \frac{Correct\ Predictions}{Total\ Predictions} = \frac{TP + TN}{TP + FP + TN + FN} \tag{7}$$

Table 1. The confusion matrix.

		Predicted Class	
		Churners	**Non-Churners**
Actual class	Churners	TP	FN
	Non-churners	FP	TN

Nevertheless, precision and recall alone may provide an incomplete picture, potentially leading to misleading conclusions. To address this, the F-measure combines precision and recall into a single metric, offering a balanced performance indicator that accounts for both aspects:

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{8}$$

The closer the F-measure is to 1, the more effectively the model balances precision and recall, reflecting superior overall performance [68].

The Matthews Correlation Coefficient (MCC), introduced by Brian W. Matthews in 1975 [69], is a robust metric for evaluating binary classification models. Unlike other metrics, MCC considers all four components of the confusion matrix—true positives (TPs), false positives (FPs), true negatives (TNs), and false negatives (FNs)—providing a comprehensive evaluation. It is particularly effective for imbalanced datasets, as it offers a balanced assessment regardless of class sizes. MCC is mathematically defined as follows:

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TM + FN)}} \tag{9}$$

The MCC value ranges from -1 to 1, where 1 signifies perfect prediction, 0 indicates performance equivalent to random guessing, and -1 represents complete disagreement between predictions and actual observations [69]. This characteristic makes MCC especially valuable for evaluating imbalanced datasets, where metrics like accuracy often fail to provide reliable insights [69,70].

Cohen's Kappa is a statistical measure that evaluates inter-rater agreement for categorical variables. Unlike simple percent agreement, Cohen's Kappa accounts for the possibility

of agreement occurring by chance, making it a more robust metric. Cohen's Kappa is mathematically defined as

$$k = \frac{p_0 - p_e}{1 - p_e} \tag{10}$$

where p_0 represents the observed agreement between raters and p_e denotes the hypothetical probability of chance agreement, calculated based on observed data frequencies. The value of Cohen's Kappa ranges from -1 to 1, with 1 indicating perfect agreement, 0 signifying no agreement beyond chance, and negative values reflecting agreement worse than random chance [71,72].

Cohen's Kappa is extensively applied across fields such as healthcare, psychology, and, more recently, machine learning. In the context of ML, it is particularly valuable for evaluating classification model performance when class distributions are imbalanced. By accounting for random chance, Cohen's Kappa provides deeper insights into model reliability, assessing the agreement between predicted and actual classifications beyond what would be expected by chance [73,74].

2.4.2. Ranking Metrics

The Receiver Operating Characteristic (ROC) curve and the area under the ROC curve (ROC AUC) are pivotal metrics in churn prediction and similar fields. The ROC curve visualizes the trade-off between the true positive rate (TPR or recall) and the false positive rate (FPR) across various classification thresholds, providing a comprehensive evaluation of a classifier's ability to distinguish between classes:

$$TruePositiveRate = \frac{TP}{TP + FN} \tag{11}$$

$$FalsePositiveRate = \frac{FP}{FP + TN} \tag{12}$$

A model with no predictive ability produces an ROC curve that aligns with the diagonal line of "no discrimination". In contrast, a perfect model's ROC curve reaches the top-left corner, signifying maximum TPR and minimal FPR. The ROC AUC metric, which quantifies the area under the ROC curve, measures this performance, with values closer to 1 indicating superior model effectiveness, as depicted in Figure 1.

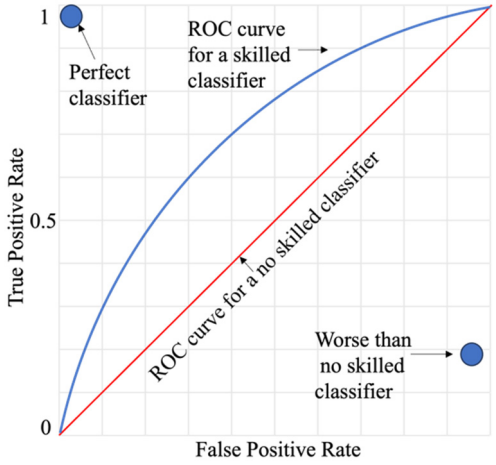


Figure 1. The ROC curve.

However, the precision–recall (PR) curve and the area under the PR curve (PR AUC) are often more informative for highly imbalanced datasets. Unlike the ROC curve, the PR curve emphasizes the minority class by illustrating the trade-off between precision and

recall, providing critical insights where the ROC curve might present an overly optimistic view [75], as depicted in Figure 2.

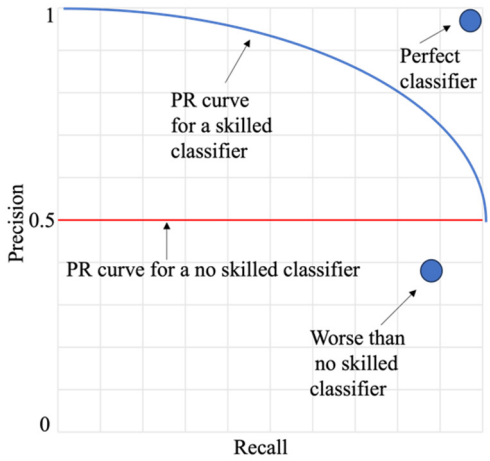


Figure 2. The precision–recall curve.

This paper adopts a comprehensive framework for evaluating ML models, synthesizing insights from threshold metrics (e.g., precision, recall, and F1 score) and ranking metrics (e.g., ROC AUC and PR AUC). By integrating these perspectives, the study ensures a holistic understanding of model performance, focusing on accurately predicting minority class instances and addressing the challenges of imbalanced datasets.

When evaluating multiple ML models across four datasets with varying levels of class imbalance ranging from moderate (15% churned) to extremely high imbalance (1% churned), it is essential to use metrics that accurately reflect the models' ability to handle imbalanced data and predict minority class instances. For datasets with severe class imbalance, such as DS4 with only 1% churners, PR AUC is often a more reliable metric than ROC AUC. The key reason is that ROC AUC evaluates the model's ability to distinguish between the positive (churn) and negative (non-churn) classes across all possible classification thresholds, considering both true positive rate (TPR) and false positive rate (FPR). However, in extreme class imbalance, the number of negative instances (non-churners) overwhelms the dataset, causing the FPR to remain low even for poorly performing models. As a result, even a model with poor precision on the minority class (churners) may still achieve a high ROC AUC score, making it an overly optimistic indicator of real-world performance.

In contrast, PR AUC focuses specifically on the minority class (churners), measuring the trade-off between precision and recall. Precision evaluates how many of the predicted churners are actually correct, while recall assesses how many of the actual churners were correctly identified. This makes PR AUC particularly useful for evaluating the effectiveness of a model in correctly predicting churners, even when their proportion is extremely low. For example, our results demonstrate that even in DS4 (1% churn rate), models achieved near-perfect ROC AUC scores (100% for some cases), but the PR AUC values were significantly lower, revealing actual differences in minority class performance.

Additionally, relying solely on ROC AUC can lead to misleading conclusions in imbalanced datasets because it does not differentiate between models that predict a large number of false positives versus models that are genuinely capable of identifying minority class instances. A classifier that predicts almost all cases as non-churners could still achieve a high ROC AUC due to the imbalance in class distribution despite failing to provide actionable insights for churn prediction.

To address this limitation, our study follows best practices in class-imbalanced learning by incorporating both ROC AUC and PR AUC, ensuring that model evaluation captures the real impact on the minority class while maintaining an overall perspective. This distinction is particularly evident in DS4, where PR AUC results expose the performance differences between models more effectively than ROC AUC alone. The selected metrics include the following:

- F1 score: Combines precision and recall through their harmonic mean, offering a balanced evaluation suited for imbalanced datasets.
- ROC AUC: Measures a classifier's ability to distinguish between classes across thresholds, indicating its capacity to prioritize positive over negative instances.
- PR AUC: Particularly effective for extreme imbalances, focusing on minority class performance and highlighting precision–recall trade-offs.
- Matthews Correlation Coefficient (MCC): Provides a balanced assessment by considering all confusion matrix elements, making it effective for datasets with significant class imbalances.
- Cohen's Kappa: Adjusts for chance agreement, offering a nuanced classifier performance evaluation.

Given the highly imbalanced nature of the datasets, accuracy is not included as it can misrepresent model performance. In our study, churn rates range from 15% to as low as 1% (DS4), meaning that a naive model that predicts all instances as non-churners could achieve over 99% accuracy while failing to capture churners entirely. Because of this, accuracy is not a reliable measure of performance in churn prediction, as it does not reflect the model's ability to classify the minority class correctly.

However, accuracy could still provide meaningful insights in a more balanced dataset (e.g., a 50-50 churn/non-churn split), especially if the misclassification cost is equal for both classes. Additionally, in preliminary model evaluations, accuracy may serve as a baseline measure before incorporating techniques that effectively handle imbalance. Despite these cases, best practices in imbalanced classification recommend F1 score, ROC AUC, PR AUC, Cohen's Kappa, and MCC, as they provide a more reliable assessment of a model's ability to distinguish churners from non-churners.

By excluding accuracy, our study aligns with these best practices to ensure that performance evaluations focus on the model's effectiveness in handling class imbalance rather than simply optimizing for majority class predictions. Precision and recall are also excluded as standalone metrics since the F1 score adequately captures their combined effect. This evaluation framework—comprising F1 score, ROC AUC, PR AUC, MCC, and Cohen's Kappa—ensures a comprehensive assessment of overall model accuracy and the ability to identify rare events, such as customer churn, in imbalanced datasets.

3. Datasets

3.1. Data Description

In the scope of this investigation, four datasets from publicly accessible sources were utilized, hereinafter referred to as DS1, DS2, DS3, and DS4. Dataset 1 (DS1) was sourced from the Kaggle platform [76] and encompasses 4250 records alongside 20 attributes, as described in Tables 2 and 3. This dataset is characterized by its heterogeneity, comprising both categorical and numerical data types; specifically, it includes 15 numerical and 5 categorical attributes. The dataset's content encompasses a wide range of customer information, such as demographic details, service subscriptions (e.g., contract duration), technical support interactions, and financial transactions (e.g., total charges incurred by the customer). The distribution of the classes within this dataset reveals a churn-to-non-churn ratio of 1:7, with the churn category constituting approximately 15% of the data.

Table 2. The names and types of different variables in the churn dataset.

Variable Name (Meaning)	Type
state (the US state of customers)	*string*
account_length (number of active months)	*numerical*
area_code (area code of customers)	*string*
international_plan (whether customers have international plans)	*yes/no*
voice_mail_plan (whether customers have voice mail plans)	*yes/no*
number_vmail_messages (number of voice-mail messages)	*numerical*
total_day_minutes (total minutes of day calls)	*numerical*
total_day_calls (total number of day calls)	*numerical*
total_day_charge (total charge of day calls)	*numerical*
total_eve_minutes (total minutes of evening calls)	*numerical*
total_eve_calls (total number of evening calls)	*numerical*
total_eve_charge (total charge of evening calls)	*numerical*
total_night_minutes (total minutes of night calls)	*numerical*
total_night_calls (total number of night calls)	*numerical*
total_night_charge (total charge of night calls)	*numerical*
total_intl_minutes (total minutes of international calls)	*numerical*
total_intl_calls (total number of international calls)	*numerical*
total_intl_charge (total charge of international calls)	*numerical*
number_customer_service_calls (number of calls to customer service)	*numerical*
churn (customer churn—the target variable)	*yes/no*

Table 3. Dataset summary.

Feature Summary	Mean	STD	Min	Max
account_length	100.24	39.70	1.0	243.0
number_vmail_messages	7.63	13.44	0.0	52.0
total_day_minutes	180.26	54.01	0.0	351.5
total_day_calls	99.91	19.85	0.0	165.0
total_day_charge	30.64	9.18	0.0	59.76
total_eve_minutes	200.17	50.25	0.0	359.3
total_eve_calls	100.18	19.91	0.0	170.0
total_eve_charge	17.02	4.27	0.0	30.54
total_night_minutes	200.53	50.35	0.0	395.0
total_night_calls	99.84	20.09	0.0	175.0
total_night_charge	9.02	2.27	0.0	17.77
total_intl_minutes	10.26	2.76	0.0	20.0
total_intl_calls	4.43	2.46	0.0	20.0
total_intl_charge	2.77	0.75	0.0	5.40
number_customer_service_calls	1.56	1.31	0.0	9.0
Summary	***Total Samples***	**Total Features**	**Churn**	***Non-Churn***
	4250	19	14.07%	85.93%

Datasets 2, 3, and 4 are produced from dataset 1 to have datasets with different imbalance rates. We used KMeans algorithm to remove several instances in the minority

class and create a dataset with 90% non-churned and 10% churned as DS2, 95% non-churned and 5% churned as DS3, and 99% non-churned and 1% churned as DS4.

3.2. Data Preprocessing

The data preprocessing journey begins with a rigorous approach to removing outliers, a crucial step in ensuring the dataset's robustness and reliability. A custom-designed function, aptly named *remove_outliers*, takes center stage in this process. This function operates systematically, meticulously analyzing each specified feature in the dataset to identify and mitigate the impact of extreme values.

The process starts by calculating each feature's first (Q1) and third quartiles (Q3). These quartiles, representing the 25th and 75th percentiles, respectively, provide a snapshot of the data's central tendency and spread. The interquartile range (IQR) is determined from these values, encapsulating the middle 50% of the data. The IQR is the benchmark for identifying outliers, forming the basis for establishing upper and lower bounds. The upper bound is set as $Q3 + 1.5 \times IQR$, while the lower bound is defined as $Q3 - 1.5 \times IQR$. Any data points falling outside these bounds are flagged as outliers. Once identified, these outliers are addressed thoughtfully. Instead of removing them, the function replaces their values with the feature's median, a robust measure of central tendency less influenced by extreme values. This approach neutralizes the outliers' impact while preserving the dataset's integrity, ensuring that no valuable information is lost. The function returns a refined dataset with all features processed, primed for accurate and reliable analysis.

The next phase involves handling categorical variables and transforming them into numerical representations suitable for machine learning models. The process begins with specific columns mapped from their original string formats to numerical codes, such as *area_code*, *international_plan*, *voice_mail_plan*, and *churn*. For instance, *area_code* values are translated into their corresponding numerical identifiers (e.g., 415, 408, 510), while binary attributes like *international_plan* and *churn* are encoded as 1 for "yes" and 0 for "no". The transformation extends to the *state* column, which, while categorical, lacks an inherent ordinal relationship. Here, an OrdinalEncoder assigns unique integers to each *state*, converting it into a format compatible with machine learning algorithms. This process ensures that all categorical features are seamlessly integrated into the dataset.

The final step focuses on normalizing numerical variables, ensuring that every feature contributes equally to the model's performance by standardizing their scales. A subset of the dataset containing only numerical features is isolated, including the newly encoded categorical variables. To normalize these features, the MinMaxScaler from scikit-learn is applied. This scaler transforms each value to a range between [0, 1], calculated by subtracting the feature's minimum value from the current value and dividing by the range (i.e., the difference between the maximum and minimum values). The transformed values maintain the dataset's original distribution patterns while ensuring that all features are on a comparable scale. Formally, this is represented as

$$u_i' = \frac{u_i - \min(u)}{\max(u) - \min(u)} \tag{13}$$

In this formula, $\min(u)$ and $\max(u)$ represent the minimum and maximum values of the i^{th} feature, respectively, while u_i' denotes the normalized value of the feature after transformation. The normalized features are then reassembled into a new Pandas DataFrame, with column names preserved for clarity and consistency. This carefully processed data, now rid of outliers, encoded for numerical compatibility, and normalized for uniformity, are ready for analysis. With its structural integrity intact and its features optimized, the

dataset stands as a robust foundation for the application of machine learning algorithms, ensuring accurate and meaningful insights in subsequent research.

4. Methods

The research methodology was systematically divided into six distinct phases: data preprocessing, feature engineering, the creation of imbalanced datasets, model selection and application of upsampling techniques, model training, and hyperparameter tuning, and model evaluation and comparison using unseen test data across four datasets with varying levels of imbalance, as illustrated in Figure 3. All analyses and practical implementations were performed using the open-source programming language Python: 3.12.0, incorporating libraries such as Numpy [77], Pandas [78], Matplotlib: 3.7.2 [79], Imblearn [80], and Scikit-learn [81], among others. The work was conducted within the Jupyter Notebook: 7.0.0 environment on an Apple MacBook Air M2 with 16 GB of RAM, ensuring a robust computational setup for this study. This computational setup provided a robust platform for managing the datasets and executing the complex data analysis and ML model development required for this study.

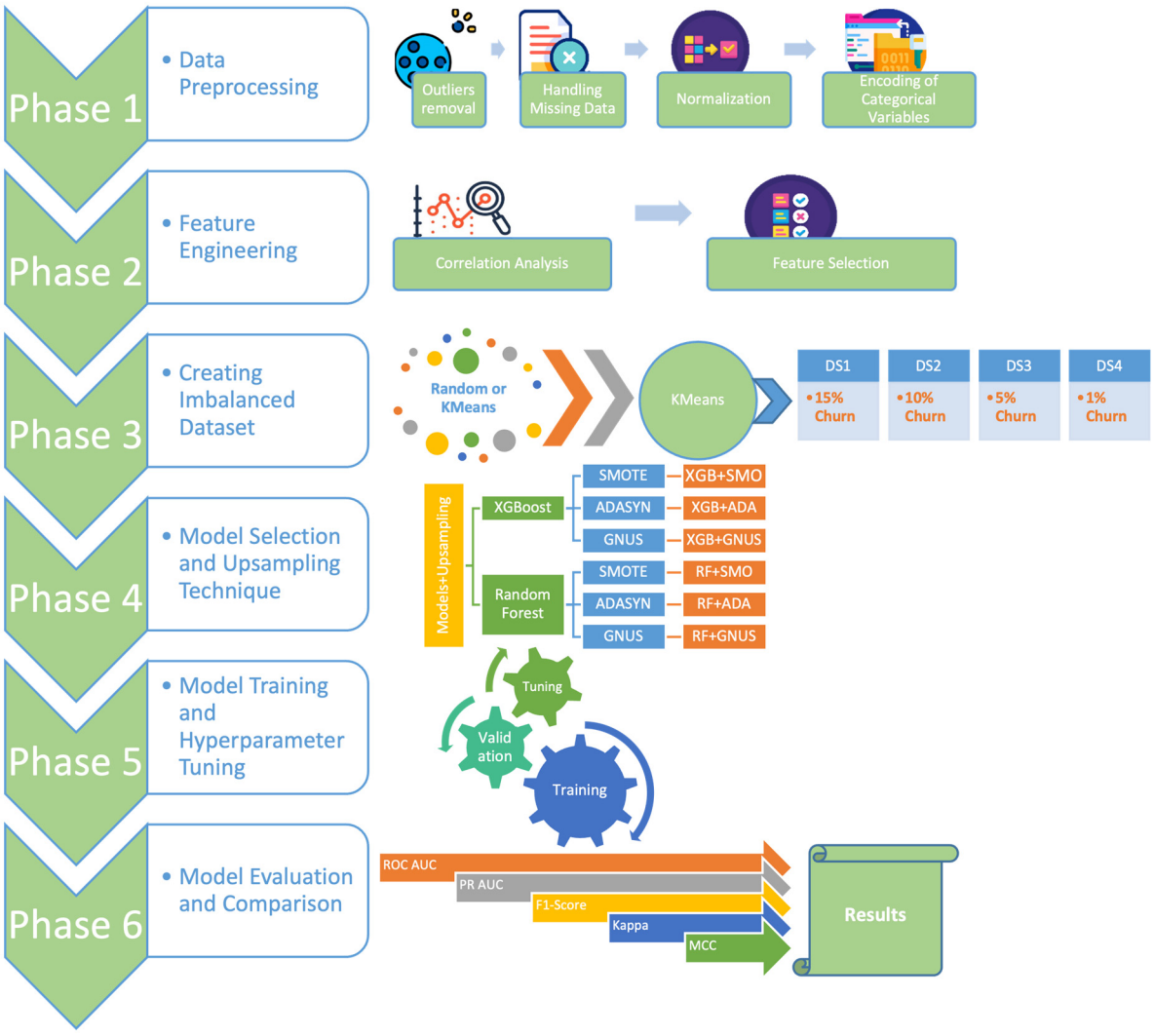


Figure 3. Methodology steps of the study.

This study addresses two critical questions about customer churn prediction in telecommunications. The first question seeks to identify the machine learning techniques most effective for predicting customer churn in highly imbalanced datasets. The sec-

ond question examines the extent to which oversampling techniques—such as SMOTE, ADASYN, and GNUS—enhance the predictive accuracy of these models. A substantial volume of data were analyzed to ensure credible and generalizable results.

To address these questions effectively, several research strategies were considered. The primary criterion for the chosen methodology was its ability to yield reliable and accurate answers to the research questions. Ultimately, an experimental research strategy was adopted to investigate specific factors influencing customer churn prediction. These factors include various machine learning algorithms and oversampling techniques. In this experimental setup, the independent variables are the machine learning algorithms and oversampling methods, while the dependent variable is the accuracy of churn prediction.

The experimental research approach was selected to ensure reproducibility, precision, and credibility in the findings, as it systematically examines the impacts of the independent variables on the dependent variable [82]. Given the characteristics of the data and the specificity of the techniques employed, this approach was determined to be the most suitable for the study's objectives.

4.1. Splitting Data

The datasets were partitioned into training and testing subsets, with 70% allocated for training and 30% for testing. This partitioning was performed using the stratified train–test split functionality available in Python's Scikit-learn Model Selection library. The stratified split ensures that the class distribution proportions in the original dataset are preserved in both subsets. This approach is critical for maintaining class balance, enabling the model to be trained and evaluated on data that accurately reflect the dataset's distribution.

Following the partitioning, the training subset underwent a 10-fold cross-validation process. This technique divides the training data into ten equal segments, iteratively using nine folds for training and one for validation in each cycle. This comprehensive approach allows for an in-depth evaluation of model performance, helping to enhance the robustness and generalizability of the predictive models by mitigating overfitting and ensuring that the model performs well across various data subsets.

In the final phase, the previously set-aside testing dataset was utilized for a thorough evaluation of the predictive capabilities of the final models. This step provided a realistic assessment of the model's performance on unseen data, offering valuable insights into their potential applicability and effectiveness in real-world scenarios.

4.2. Feature Selection

Feature engineering is a critical component in developing machine learning models, involving transforming and creating new features from existing ones to optimize the feature space. The primary goal is to derive variables that enhance model performance by providing the algorithm with more meaningful or refined inputs. This process offers several advantages, including reducing the risk of overfitting, simplifying model complexity, and conserving computational resources. While feature engineering may sometimes compromise model interpretability, it accelerates the learning process and mitigates redundant covariances among features, contributing to more robust models [83].

In this study, feature engineering commenced classifying features into numerical and categorical groups based on their data types. This initial step was crucial for applying appropriate transformation and preprocessing techniques suited to each type. An in-depth correlation analysis followed, focusing specifically on the numerical features. The study aimed to uncover potential redundancies or interdependencies that could negatively impact the models by examining the relationships among these features.

Special attention was given to identifying numerical features with high interdependence. Features that exhibited strong correlations with others were flagged as potentially redundant, as they might not provide unique or additional value to the models. Furthermore, such redundancies could introduce noise or unnecessary complexity, detracting from the model's overall performance. Based on this analysis, a strategic subset of features was selected for removal. The decision to eliminate these features was driven by the intent to streamline the dataset, ensuring that the remaining features contributed distinct and valuable information to the learning process.

To achieve this, the study employed a correlation matrix of the numerical features, visualized through a heatmap. This visual representation facilitated the identification of highly correlated variables, making it easier to pinpoint features for removal. By reducing the duplication of information and eliminating redundant variables, the dataset was refined to optimize its utility for ML models.

This meticulous feature engineering process resulted in a more concise and meaningful dataset. The streamlined feature set reduced complexity and enhanced the efficiency and effectiveness of the models, ultimately aiming to improve predictive accuracy. With the refined dataset, this study proceeded to the subsequent model training and evaluation phases, laying a strong foundation for robust and high-performing ML models.

4.3. Creating Highly Imbalanced Datasets

To thoroughly evaluate the performance of our models and sampling techniques, we constructed highly imbalanced datasets derived from the existing data. This involved two distinct methodologies: a random undersampling approach and KMeans clustering. These approaches were designed to create varying levels of imbalance, providing a robust framework for testing the effectiveness of the models and sampling strategies under different conditions.

4.3.1. Random Undersampling

The random undersampling approach creates highly imbalanced datasets by selectively reducing the number of minority class instances (churn) while preserving the majority class instances (non-churn). This method replicates extreme imbalance scenarios, providing a robust framework for testing model performance under challenging conditions:

1. Selectively Reduce Minority Instances: A subset of the minority class is randomly selected to achieve the desired imbalance ratio. This ensures that the minority class size aligns with the targeted level of imbalance.
2. Maintain Majority Class: The majority class remains unchanged, ensuring that the overall dataset size is not significantly affected.
3. Random Sampling: The *DataFrame.sample()* function is used with a specified *random_state* to ensure reproducibility. The n parameter is adjusted to control the number of minority class instances retained.
4. Shuffling: After combining the reduced minority class and the unchanged majority class, the dataset is shuffled to eliminate any order-based biases that could influence the training phase.

This simple and efficient approach may result in the loss of valuable information in the minority class. Therefore, validating the generalizability of models trained on such imbalanced datasets is crucial to ensure their applicability to real-world scenarios. This step is essential to assess whether the models can maintain their predictive performance despite losing some minority class information.

4.3.2. KMeans Clustering

To mitigate the drawbacks of Random Undersampling, we implemented KMeans-based clustering for minority class reduction. Instead of removing minority samples arbitrarily, this approach retains only the most representative samples, ensuring that the distribution of the minority class is better preserved:

1. Cluster Formation and Centroid Selection: Minority class instances are grouped into clusters using KMeans clustering, with each cluster representing a subgroup of the churners. The centroid of each cluster is retained as the most representative instance.
2. Preserving Informative Features: By focusing on cluster centroids, this method ensures that structural characteristics of the minority class are maintained, reducing the risk of removing critical data points.
3. Avoiding Redundant Data: Unlike Random Undersampling, which may eliminate important or rare churn cases, KMeans strategically selects instances that best define the minority class distribution.

Since clustering-based undersampling preserves minority class diversity, while reducing dataset size, models trained on KMeans-reduced datasets consistently outperformed those trained on randomly undersampled datasets. This was particularly evident in scenarios with extreme class imbalance, where retaining meaningful minority class representations is crucial.

Unlike Random Undersampling, which removes minority class instances without considering their distribution in the feature space, KMeans clustering offers a more structured approach by retaining representative centroids that capture the core characteristics of the minority class. Selecting these centroids rather than randomly eliminating data points helps preserve class separability, ensuring that the diversity within the minority class remains intact. Additionally, by maintaining a well-balanced representation, KMeans reduces the risk of overfitting to the majority class, a common issue when key churn instances are lost during random sampling. Instead of training on an arbitrarily reduced dataset, models benefit from learning on a strategically selected subset that better reflects the overall distribution of the minority class. These advantages make KMeans-based undersampling a more effective strategy for managing class imbalance in churn.

Despite its advantages, KMeans undersampling is not without limitations. A key concern is that it assumes that centroids adequately represent the minority class distribution. If the minority class contains high variability or rare subgroups, centroid-based selection may oversimplify the data, potentially introducing bias by failing to retain underrepresented patterns.

Additionally, the effectiveness of this approach depends on the number of clusters chosen. Setting too few clusters risks oversimplifying the minority class, while too many clusters reduces the efficiency of undersampling, leading to minimal dataset size reduction. Future work could explore adaptive clustering techniques that dynamically determine the optimal number of clusters based on data complexity

4.4. Class Imbalance

In churn prediction, datasets often exhibit significant class imbalances from the relatively lower incidence of customer churn compared to those who continue using services. This imbalance creates a critical challenge for ML models, which tend to favor the majority class, thereby neglecting the minority class. Such bias undermines predictive accuracy, particularly in detecting churn, which is typically the primary focus. In this study, the datasets revealed notable class imbalances, with churn rates of approximately 15% in Dataset 1, 10% in Dataset 2, 5% in Dataset 3, and as low as 1% in Dataset 4.

To address this issue, this study employed a strategic approach utilizing three advanced oversampling techniques: SMOTE, ADASYN, and GNUS. These methods are designed to augment the representation of the minority class, thereby creating a more balanced dataset for training. By increasing the presence of churn instances, these techniques enhance the model's ability to learn patterns associated with customer churn, mitigating the adverse effects of class imbalance.

A pivotal element in implementing these oversampling techniques was their restriction to the training dataset within a 10-fold cross-validation. This strategy involved applying oversampling exclusively to nine folds while reserving the tenth fold for evaluation. This careful segmentation ensures that the evaluation data remain unaltered, avoiding inflated performance estimates that could arise from oversampling being applied indiscriminately across the entire dataset. By confining oversampling to the training phase, this study minimized the risk of overfitting and ensured a more realistic and generalizable assessment of model performance.

This deliberate and methodical approach underscores the importance of addressing class imbalance in churn prediction datasets. By integrating oversampling techniques into the cross-validation framework, this study aimed to develop ML models that are both robust and sensitive to the intricacies of churn behavior, ultimately contributing to more accurate and reliable predictive performance.

5. Results and Discussion

5.1. Results on DS1 with 15% Churn Ratio

The evaluation shows that the tuned XGBoost model consistently outperforms Random Forest, particularly when paired with the SMOTE upsampling technique. Under SMOTE, XGBoost achieves the highest scores across all key metrics, including a 4% higher F1 score, a 6% higher PR-AUC, and notable improvements in Kappa and MCC (5% and 4%, respectively). This demonstrates the strong synergy between XGBoost's advanced learning capabilities and SMOTE's synthetic sample generation.

Using ADASYN, XGBoost shows moderate improvements over Random Forest, with a 1% higher F1 score and a significant 7% advantage in PR-AUC, though both models exhibit similar Kappa statistics. GNUS is the least effective upsampling method, with both models performing equally in F1 score and Kappa, but XGBoost still maintains modest gains in ROC-AUC (4%) and PR-AUC (3%).

Overall, SMOTE-enhanced XGBoost delivers the best performance, significantly surpassing ADASYN and GNUS in addressing class imbalance. These findings, detailed in Table 4 and Figure 4, affirm SMOTE as the most effective technique and XGBoost as the most robust model for this dataset.

Table 4. Evaluation metrics of the models for DS1 with a 15% churn ratio with the highest values highlighted in bold and marked with an asterisk.

Models/Metrics	F1 Score%	ROC-AUC%	PR-AUC%	Kappa%	MCC%
Tuned_RF_ADASYN	62	84	65	56	57
Tuned_XGB_ADASYN	63	**86 ***	72	56	56
Tuned_RF_SMOTE	64	85	67	58	59
Tuned_XGB_SMOTE	**68 ***	**86 ***	**73 ***	**63 ***	**63 ***
Tuned_RF_GNUS	61	81	65	56	57
Tuned_XGB_GNUS	61	85	68	56	58

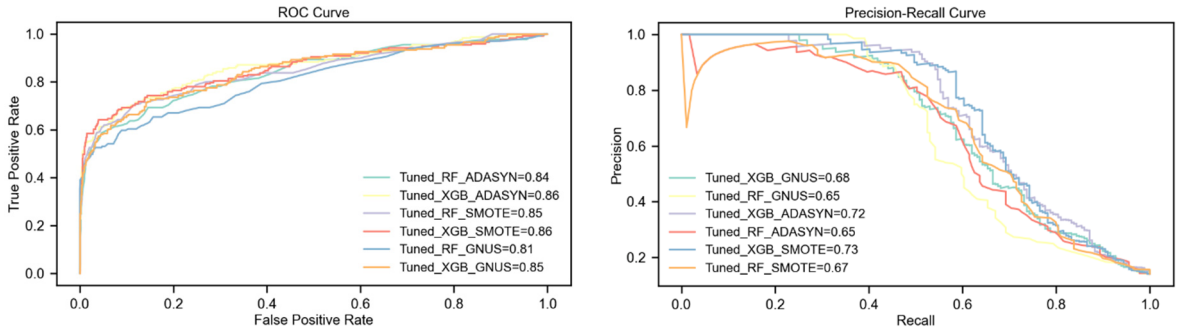


Figure 4. The ROC and PR curves for different combinations of models and upsampling methods with a 15% churn ratio.

5.2. Results on DS2 with 10% Churn Ratio

The analysis reveals that the tuned XGBoost model paired with SMOTE consistently achieves the highest performance across all metrics, including F1 score, PR-AUC, Kappa, and MCC. SMOTE is the most effective upsampling method, enabling XGBoost to significantly outperform Random Forest, particularly with a 7% higher F1 score and 8% higher Kappa statistic.

With ADASYN, XGBoost maintains an advantage, achieving a 3% higher F1 score and slightly improving PR-AUC over Random Forest. However, both models show equal ROC-AUC performance under this technique. GNUS, by contrast, favors Random Forest, which outperforms XGBoost in most metrics, highlighting GNUS's limited compatibility with XGBoost.

Overall, SMOTE-enhanced XGBoost emerges as the optimal approach for addressing class imbalance, consistently delivering the best results, as summarized in Table 5 and illustrated in Figure 5.

Table 5. Evaluation metrics of the models for the second dataset with a 10% churn ratio with the highest values highlighted in bold and marked with an asterisk.

Models/Metrics	F1 Score%	ROC-AUC%	PR-AUC%	Kappa%	MCC%
Tuned_RF_ADASYN	68	88	71	65	67
Tuned_XGB_ADASYN	71	88	73	68	69
Tuned_RF_SMOTE	67	88	74	64	66
Tuned_XGB_SMOTE	**74** *	**89** *	**77** *	**72** *	**73** *
Tuned_RF_GNUS	60	86	64	55	55
Tuned_XGB_GNUS	56	84	65	50	51

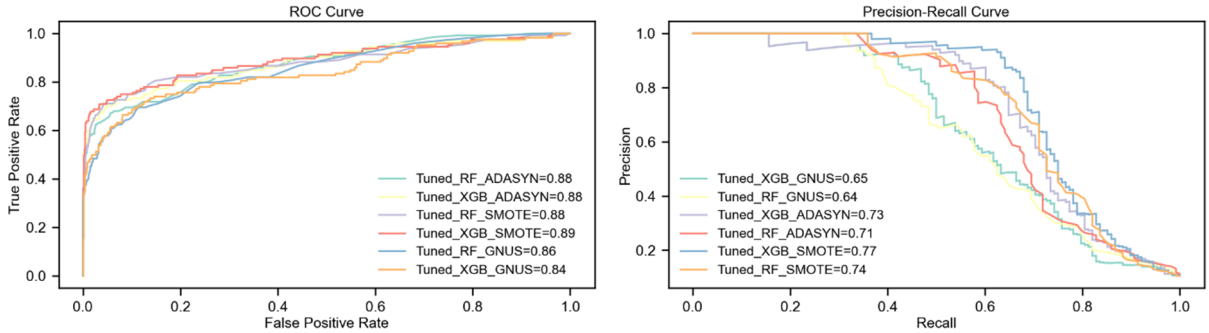


Figure 5. The ROC and PR curves for different combinations of models and upsampling methods with a 10% churn ratio.

5.3. Results on DS3 with 5% Churn Ratio

The analysis highlights the consistent superiority of the tuned XGBoost model over Random Forest across all upsampling techniques, particularly with SMOTE. Under SMOTE, XGBoost achieves the highest overall performance, including a 5% higher F1 score (75%), a 7% higher PR-AUC (79%), and significant gains in Kappa and MCC (6% and 5%, respectively). This combination demonstrates the strong synergy between XGBoost's advanced learning capabilities and SMOTE's ability to generate effective synthetic samples.

With ADASYN, XGBoost outperforms Random Forest by 5% in the F1 score and PR-AUC, while both models achieve an equal ROC-AUC of 90%. Additionally, XGBoost shows notable improvements of 6% in Kappa and MCC, reflecting its stronger predictive reliability. GNUS reveals the largest performance gap, where XGBoost outshines Random Forest with an 11% higher F1 score, a 15% improvement in PR-AUC, and double-digit gains in Kappa and MCC.

Overall, SMOTE is the most effective upsampling technique, delivering the best results for both models, especially XGBoost. While ADASYN provides moderate benefits, GNUS proves less effective overall, though XGBoost demonstrates remarkable resilience under this technique. The combination of tuned XGBoost with SMOTE achieves the highest scores across all metrics, affirming its robustness and suitability for handling extreme class imbalance. Results are detailed in Table 6 and illustrated in Figure 6.

Table 6. Evaluation metrics of the models for the third dataset with a 5% churn ratio with the highest values highlighted in bold and marked with an asterisk.

Models/Metrics	F1 Score%	ROC-AUC%	PR-AUC%	Kappa%	MCC%
Tuned_RF_ADASYN	65	90	71	63	63
Tuned_XGB_ADASYN	70	90	76	69	69
Tuned_RF_SMOTE	70	92	72	68	69
Tuned_XGB_SMOTE	**75 ***	**93 ***	**79 ***	**74 ***	**74 ***
Tuned_RF_GNUS	57	90	62	54	55
Tuned_XGB_GNUS	68	**93 ***	77	66	66

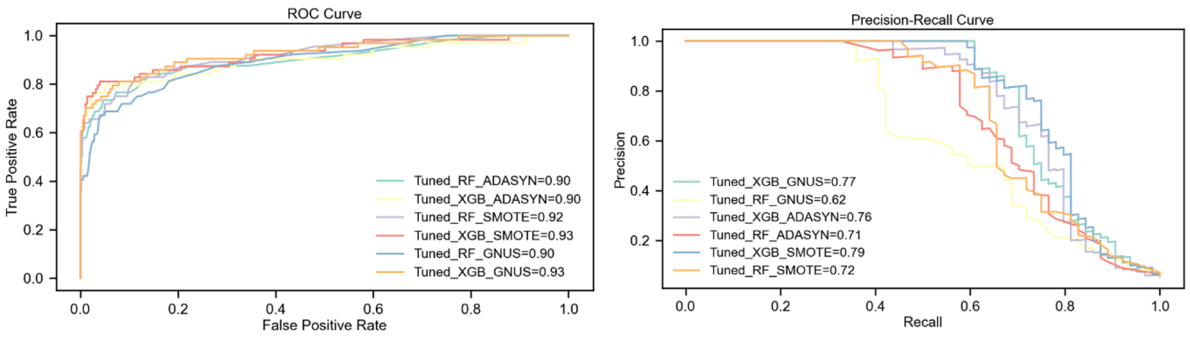


Figure 6. The ROC and PR curves for different combinations of models and upsampling methods with a 5% churn ratio.

5.4. Results on DS4 with 1% Churn Ratio

The evaluation highlights the tuned XGBoost model's remarkable superiority over Random Forest across all upsampling techniques, particularly in scenarios of extreme class imbalance. With ADASYN, XGBoost achieves a significantly higher F1 score and Kappa (80% vs. 47%) and a substantial 25-point advantage in MCC (80% vs. 55%) despite both models achieving a perfect ROC AUC of 100%. XGBoost also demonstrates a slight edge in PR AUC, scoring 95% compared to Random Forest's 94%.

SMOTE further amplifies the performance gap, with XGBoost reaching an F1 score and Kappa of 92%, outperforming Random Forest by 36%. XGBoost also surpasses Random Forest in PR AUC by 4% (97% vs. 93%) and MCC by 30 points (92% vs. 62%) while maintaining a perfect ROC AUC of 100%. These results confirm the effectiveness of combining SMOTE with XGBoost in handling extreme class imbalance.

The most significant performance differences emerge with GNUS. XGBoost achieves an F1 score of 74%, 47% higher than Random Forest, and a PR AUC of 86%, vastly outperforming Random Forest's 20%. XGBoost also achieves a 41-point lead in MCC (74% vs. 33%) and a 7% higher ROC AUC (99% vs. 92%). These findings highlight XGBoost's robustness, even under less effective upsampling methods like GNUS.

The results demonstrate that MCC and Kappa exhibit different sensitivity to extreme class imbalance depending on the model and resampling technique. While it is often assumed that MCC fluctuates more than Kappa, our experiments show that this depends on the specific model and dataset. For instance, in Tuned_RF_GNUS, MCC exhibits more significant fluctuation in DS4 (1% churn), dropping from 55 to 33, while Kappa drops from 54 to 25. This aligns with MCC's formulation, which is highly sensitive to class distribution distortions when the minority class becomes extremely small. However, in other models, such as Tuned_RF_ADASYN and Tuned_RF_SMOTE, Kappa fluctuates more than MCC, indicating that MCC is not universally more volatile. These findings suggest that the impact of imbalance on evaluation metrics is model-dependent, and both metrics must be interpreted carefully when dealing with extreme class imbalance.

Overall, SMOTE-enhanced XGBoost delivers the highest performance across all metrics, establishing itself as the most effective strategy for this dataset with a 1% churn rate. While ADASYN provides moderate improvements for XGBoost, it yields far lower results for Random Forest. GNUS is the least effective for Random Forest, but XGBoost consistently demonstrates resilience. The results in Table 7 and Figure 7 emphasize the critical importance of pairing advanced models like XGBoost with effective upsampling techniques to address extreme class imbalance.

Table 7. Evaluation metrics of the models for the fourth dataset with a 1% churn ratio with the highest values highlighted in bold and marked with an asterisk.

Models/Metrics	F1 Score%	ROC-AUC%	PR-AUC%	Kappa%	MCC%
Tuned_RF_ADASYN	47	**100** *	94	47	55
Tuned_XGB_ADASYN	80	**100** *	95	80	80
Tuned_RF_SMOTE	56	**100** *	93	55	62
Tuned_XGB_SMOTE	**92** *	**100** *	**97** *	**92** *	**92** *
Tuned_RF_GNUS	27	92	20	25	33
Tuned_XGB_GNUS	74	99	86	74	74

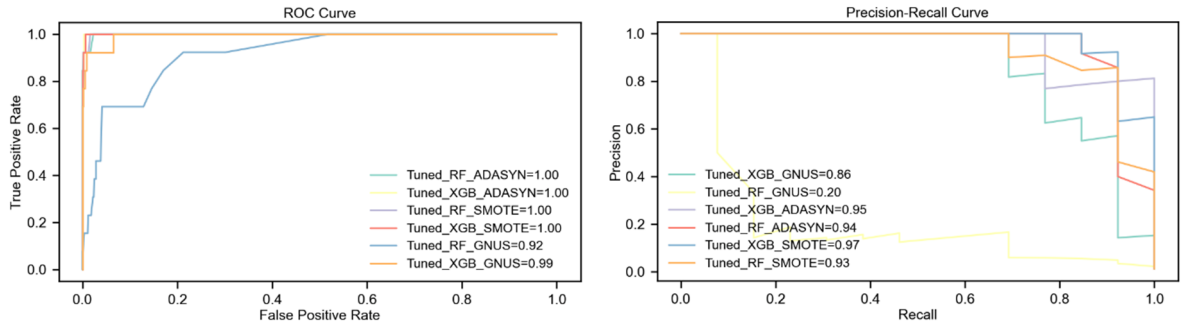


Figure 7. The ROC curve for different combinations of models and upsampling methods with a 1% churn ratio.

5.5. Analysis of Average Performance and Variability

Tables 8 and 9 show a comprehensive analysis of the effectiveness and consistency of various machine learning models and upsampling techniques applied to datasets with differing churn ratios. These results reveal distinct patterns in both performance and variability, facilitating identifying the most robust strategies for managing imbalanced datasets.

The data in Table 8 highlight significant differences in model performance across various evaluation metrics. Tuned_XGB_SMOTE consistently achieves the highest average scores among all models, indicating superior effectiveness in handling imbalanced datasets with different churn ratios. Precisely, it attains the highest average F1 score (77.25%), ROC-AUC (92.0%), PR-AUC (81.5%), Kappa (75.25%), and MCC (75.5%). This suggests that combining the XGBoost classifier with the SMOTE upsampling technique yields the most robust predictive performance.

In contrast, Tuned_RF_GNUS exhibits the lowest average performance across all metrics, with an average F1 score of 51.25% and PR-AUC of 52.75%, indicating limited effectiveness in addressing class imbalance when using the Random Forest model with the GNUS method.

The other models, such as Tuned_XGB_ADASYN and Tuned_RF_SMOTE, show moderate performance, with Tuned_XGB_ADASYN being the second-best performer. This model achieves an average F1 score of 71.0% and an average Kappa of 68.25%, suggesting that the XGBoost model combined with the ADASYN technique is also effective but less so than when paired with SMOTE.

Overall, the analysis underscores the importance of selecting appropriate combinations of machine learning algorithms and upsampling techniques. The superior performance of Tuned_XGB_SMOTE highlights its suitability as a robust strategy for managing imbalanced datasets, particularly in scenarios where predictive accuracy and reliability are critical.

Table 8. Average performance metrics of models across all datasets with the highest values highlighted in bold and marked with an asterisk.

Models/Metrics	Avg F1 Score%	Avg ROC-AUC%	Avg PR-AUC%	Avg Kappa%	Avg MCC%
Tuned_RF_ADASYN	60.5	90.5	75.25	57.75	60.5
Tuned_XGB_ADASYN	71.0	91.0	79.0	68.25	68.5
Tuned_RF_SMOTE	64.25	91.25	76.5	61.25	64.0
Tuned_XGB_SMOTE	**77.25** *	**92.0** *	**81.5** *	**75.25** *	**75.5** *
Tuned_RF_GNUS	51.25	87.25	52.75	47.5	50.0
Tuned_XGB_GNUS	64.75	90.25	74.0	61.5	62.25

The standard deviation is a key statistical measure that quantifies the extent of variation or dispersion within a numerical dataset. It reflects the average distance of individual data points from the dataset's mean, offering valuable insight into the overall variability of the data.

The calculation of standard deviation serves several important purposes in the context of machine learning model evaluation:

1. Assessing Consistency: Standard deviation helps to gauge how consistently a model performs across different datasets by measuring the variability in its performance metrics.
2. Comparing Stability: Models with lower standard deviations exhibit more stable performance, indicating their reliability across varying datasets.
3. Identifying Variability: A high standard deviation reveals significant variability in model performance, which could be attributed to dataset characteristics or the model's sensitivity to specific features or configurations.

This metric plays a critical role in understanding and comparing the robustness of machine learning models in diverse scenarios. The standard deviation is as follows:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x^-)^2}{n-1}} \tag{14}$$

where

$$x^- = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{15}$$

- s: Sample standard deviation.
- x_i: Each data point in the dataset.
- x^-: The sample mean (average of data points).
- n: The total number of data points in the sample.
- $n-1$: Degrees of freedom. Subtracting 1 accounts for the fact that we are estimating the population standard deviation from a sample.

We used the same methodology to compute the standard deviations for all models and performance metrics. The results are summarized in the table below, providing a comprehensive view of the variability in model performance across different datasets. This table highlights the consistency and stability of each model, offering insights into their robustness and reliability.

Table 9. Standard deviation of performance metrics for each model across datasets.

Models/Metrics	SD F1 Score%	SD ROC-AUC%	SD PR-AUC%	SD Kappa%	SD MCC%
Tuned_RF_ADASYN	9.33	6.81	12.82	8.14	5.51
Tuned_XGB_ADASYN	7.35	6.83	9.80	10.98	10.74
Tuned_RF_SMOTE	6.25	6.75	7.50	5.68	4.97
Tuned_XGB_SMOTE	10.30	6.06	10.63	12.15	12.07
Tuned_RF_GNUS	14.87	4.57	19.84	12.45	11.66
Tuned_XGB_GNUS	7.50	6.34	6.24	10.33	10.45

Note: All standard deviations are calculated using the sample variance formula (dividing by n − 1).

XGBoost is the most effective model overall, consistently outperforming Random Forest in average metrics and adaptability. Among the configurations, XGBoost combined with SMOTE demonstrated the highest average performance across all datasets, achieving exceptional scores such as an F1 score of 77.25% and a PR-AUC of 81.5%. This combination excelled at balancing precision and recall while maintaining strong reliability in predictions, as reflected in its high Kappa and MCC values. The slightly higher average ROC-AUC of 92.0% further reinforced its ability to distinguish between churners and non-churners. However, this strong performance came with notable variability, particularly in metrics like F1 score and Kappa, indicating sensitivity to the degree of class imbalance.

When paired with ADASYN, XGBoost also delivered impressive results, with an average F1 score of 71.0% and PR-AUC of 79.0%. Although slightly less effective than SMOTE, ADASYN provided more consistent outcomes, as evidenced by its lower standard deviations. This stability suggests that ADASYN's approach to generating synthetic samples creates reliable improvements across datasets with varying churn rates.

In contrast, the GNUS technique proved less effective for both models, though XGBoost managed to perform reasonably well, with an average F1 score of 64.75% and PR-AUC of 74.0%. Random Forest struggled the most with GNUS, achieving an average F1 score of only 51.25% and a PR-AUC of 52.75%. These results indicate that GNUS was insufficient in generating the diversity and representativeness needed to enhance learning

from the minority class, particularly for Random Forest. Furthermore, GNUS exhibited the highest variability in performance metrics, such as PR-AUC and F1 score, underscoring its inconsistency across datasets.

Random Forest's performance was comparatively more stable when combined with SMOTE or ADASYN. SMOTE helped Random Forest achieve better average metrics, such as an F1 score of 64.25% and a PR-AUC of 76.5%. However, even in this configuration, it fell short of XGBoost's performance, highlighting the limitations of Random Forest in handling complex class imbalance scenarios. The relatively low variability in Random Forest's results with SMOTE indicates that it benefited from the upsampling technique's ability to generate more representative minority class samples, albeit with less overall effectiveness than XGBoost.

The standard deviations further illuminate these trends, revealing that XGBoost with SMOTE exhibited more significant variability while achieving the best average performance, particularly under extreme imbalance conditions. This variability suggests that the model's performance is highly dynamic, with substantial gains in challenging scenarios. By contrast, XGBoost with ADASYN delivered more consistent but slightly lower performance, making it a reliable alternative for cases where stability is prioritized.

The contrast between the models is particularly striking in their ability to handle GNUS. Random Forest showed significant declines in performance and higher variability, reflecting its inability to generalize effectively from GNUS-generated samples. On the other hand, XGBoost demonstrated adaptability, leveraging even the limited benefits of GNUS to achieve better predictions, though not at the levels seen with SMOTE or ADASYN.

In summary, the analysis highlights the clear advantage of XGBoost in handling imbalanced datasets, especially when paired with SMOTE. While SMOTE provided the highest overall performance, ADASYN emerged as a more stable alternative. GNUS, on the other hand, proved less effective and highly variable, particularly for Random Forest. These findings emphasize the importance of selecting the right combination of model and upsampling technique to address class imbalance effectively. The consistent superiority of XGBoost and its adaptability to challenging conditions underscores its value as a robust solution for predictive modeling in imbalanced scenarios.

5.6. Statistical Tests

Our investigation systematically compared six models evaluated across four distinct datasets and measured by five different performance metrics. We embraced a non-parametric approach, beginning with the Friedman test, to discern whether the differences we observed truly reflected each model's performance—or merely artifacts of chance. This test was particularly appealing because it is well-suited for scenarios where more than two algorithms are compared [84,85]. For example, we explored various combinations of Random Forest and XGBoost, each paired with different upsampling methods.

5.6.1. Friedman Test

Given that each model was evaluated across the same four datasets, their results were inherently correlated. Consequently, the Friedman test was the most appropriate method, accounting for repeated-measures designs. Moreover, our chosen performance metrics—F1 score, ROC-AUC, PR-AUC, Cohen's Kappa, and MCC—do not always conform to a normal distribution, especially given the limited datasets. This non-parametric, rank-based procedure thus offered a robust alternative to more traditional methods that assume normality and homogeneity of variances. We began by selecting a specific metric and organizing the data into a table representing each model's performance across the four datasets. For each dataset, we assigned ranks to the models, with rank 1 indicating the best

performance and 6 representing the poorest. With these rankings in hand, we computed the average rank for each model and derived the Friedman statistic, yielding a *p*-value to indicate whether any model consistently outperformed the rest.

When the Friedman test revealed a significant overall difference ($p < 0.05$), we proceeded with the Nemenyi post hoc test. This step allowed us to delve deeper, comparing each pair of models to pinpoint where the significant differences lay. Following this rigorous workflow for each metric ensured that our conclusions about model performance were based on statistically sound evidence rather than mere chance.

Our methodical use of the Friedman and Nemenyi tests provided a transparent, data-driven narrative that allowed us to interpret the performance differences observed across multiple datasets confidently. This approach highlights the strengths and weaknesses of each model and reinforces the reliability of our conclusions regarding their comparative effectiveness. The Friedman test statistic is computed using the following mathematical formulation when given the following notations:

- $k =$ Number of models (6);
- $N =$ Number of datasets (4);
- ${R'}_j =$ Average rank of the model j across the N datasets;
- $\frac{k+1}{2} =$ Midpoint of ranks (for $k = 6$, midpoint = 3.5).

$$\chi^2{}_F = \frac{12 \times N}{k(k+1)} \sum_{j=1}^{k} \left({R'}_j - \frac{k+1}{2} \right)^2$$

with degrees of freedom = $k - 1 = 5$.

The resulting $\chi^2{}_F$ statistic is approximately distributed as a Chi-squared ($df = 5$) for moderate N. If $\chi^2{}_F$ exceeds the critical Chi-squared value at the chosen α (or if the $p - value < \alpha$), reject the null hypothesis H_0 (i.e., there is at least one model that differs significantly).

Table 10 presents the F1 score rankings for each model across the datasets, where the best performance is assigned a rank of 1. It is important to note that in DS3, a tie occurred between Tuned_XGB_ADASYN (70) and Tuned_RF_SMOTE (70), each receiving a rank of 2.5. Similarly, in DS1, a tie between Tuned_RF_GNUS (61) and Tuned_XGB_GNUS (61) resulted in both models being assigned a rank of 5.5.

Table 10. Ranking F1 score per dataset (best = rank 1).

Models/Datasets	DS1	DS2	DS3	DS4
Tuned_RF_ADASYN	4	3	5	5
Tuned_XGB_ADASYN	3	2	2.5	2
Tuned_RF_SMOTE	2	4	2.5	4
Tuned_XGB_SMOTE	1	1	1	1
Tuned_RF_GNUS	5.5	5	6	6
Tuned_XGB_GNUS	5.5	6	4	3

Subsequently, we computed both the sum and the average of the ranks. For instance, consider the model Tuned_RF_ADASYN evaluated on the F1 score metric across four datasets. The ranks obtained were 4, 3, 5, and 5, totaling 17, resulting in an average rank of 4.25. Employing this same procedure, we calculated the average ranks for all other metrics and models across the four datasets, as summarized in Table 11.

Table 11. The average ranks for all other metrics and models across the four datasets.

	Average Ranks (from the Friedman Step)			
Models/Metrics	**F1 Score**	**PR-AUC**	**Kappa**	**MCC**
Tuned_RF_ADASYN	4.25	4.375	4.375	4.375
Tuned_XGB_ADASYN	2.375	2.5	2.625	3.125
Tuned_RF_SMOTE	3.125	3.5	3	2.875
Tuned_XGB_SMOTE	1	1	1	1
Tuned_RF_GNUS	5.625	5.875	5.375	5.375
Tuned_XGB_GNUS	4.625	3.75	4.625	4.25

Now, in this step, we calculate the Friedman Statistic results for the F1 score based on the above formula:

$$\sum_{j=1}^{6} \left(R'_j - 3.5\right)^2 = 14$$

$$\chi^2_F = \frac{12 \times 4}{6 \times 7} \times 14 = \frac{48}{42} \times 14 = 1.142857 \times 14 = 16$$

with degrees of freedom = 5 (since $k = 6$).

We computed the Friedman test statistic (χ^2_F = 16.0) with five degrees of freedom—reflecting the number of models minus one—and compared it to the Chi-squared distribution. The value of 16.0 lies between the critical values for p = 0.01 and p = 0.005 for five degrees of freedom, indicating a p-value between these thresholds. Further refinement using statistical software yielded an approximate p-value of 0.007. This result implies that the likelihood of observing such a statistic by chance is below 1%, thereby providing sufficient evidence to reject the null hypothesis at the 5% significance level. Using the same procedure, we derived the Friedman test outcomes and corresponding approximate p-values for all metrics, as summarized in Table 12.

Table 12. The Friedman test results and corresponding approximate p-values for all metrics.

Metrics	Friedman χ^2_F	Approx. p-Value	Significant at 0.05
F1 score	16	~0.007	**Yes**
ROC-AUC	9.6	~0.08–0.09	No
PR-AUC	15.7	~0.008	**Yes**
Kappa	13.88	~0.016	**Yes**
MCC	12.65	~0.027	**Yes**

Table 12 illustrates that the Friedman test reveals statistically significant differences among the six models for F1, PR-AUC, Kappa, and MCC at the 5% level. In contrast, the ROC-AUC metric does not reach significance ($p \approx 0.08$), leading us to retain the null hypothesis for this measure. Consequently, a post hoc analysis will identify the specific model differences for the metrics that exhibit overall significance.

5.6.2. Post Hoc Analysis

This section provides an overview of our post hoc analysis utilizing the Nemenyi test, which was conducted on the metrics that exhibited significant differences in the Friedman test. The analysis began by ranking the models on each dataset—assigning a rank of 1 for the best performance and 6 for the worst. We then computed the average rank for each

model across all datasets. Next, we calculated the critical difference (CD) at a significance level $\alpha = 0.05$. For k models and N datasets, the CD for Nemenyi is given by

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

where q_α represents the critical value from the Studentized range distribution for k groups at the chosen significance level α, k is the number of models, and N is the number of datasets. Finally, we compared the absolute differences in average ranks between each pair of models to the CD; if for model i and model $j : \mid R'_i - R'_j \mid \geq$ CD, then the difference meets or exceeds the CD, and the pair is considered significantly different at $\alpha = 0.05$. For our analysis, with six models and four datasets, the approximate CD is 3.80.

By applying the Nemenyi post hoc procedure to the results presented in Table 12, we observed that among the evaluated metrics—F1 score, PR-AUC, Cohen's Kappa, and MCC—only the comparison between Tuned_XGB_SMOTE and Tuned_RF_GNUS yielded a critical difference (CD) of 3.8 or greater. In essence, for every metric where the Friedman test signaled statistically significant differences (except ROC-AUC), the only pair of models that demonstrated a statistically significant difference at the conventional $\alpha = 0.05$ level was that between Tuned_XGB_SMOTE and Tuned_RF_GNUS.

Although the comparisons between Tuned_XGB_SMOTE and both Tuned_XGB_GNUS and Tuned_RF_ADASYN approached the critical difference threshold (CD $\approx$ 3.80), they ultimately did not meet the criteria for significance. Under the strict binary decision rule of the Nemenyi test, these pairs are formally deemed non-significant, even though their proximity to the threshold suggests a potential trend favoring Tuned_XGB_SMOTE.

In terms of model performance, Tuned_XGB_SMOTE frequently achieved the highest average ranks across the datasets, establishing it as the dominant model. However, given our limited sample of four datasets, it was only statistically superior to Tuned_RF_GNUS. Conversely, Tuned_RF_GNUS consistently ranked near the bottom, and the post hoc analysis confirmed that its performance gap with Tuned_XGB_SMOTE was indeed significant.

It is essential to acknowledge that the relatively small number of datasets contributes to a more considerable critical difference, and with a larger sample size, additional significant differences might emerge. Moreover, while this study focuses on customer churn prediction—a domain that, although necessary, may not be as critical as sectors such as healthcare or finance—we have adhered to the conventional significance level of $\alpha = 0.05$. For applications in industries with higher stakes, such as healthcare, it may be prudent to recalculate the significance using a more stringent threshold (e.g., $\alpha = 0.01$ or $\alpha = 0.001$) to mitigate the risk of false positives that could result in costly or unsafe outcomes.

5.7. Overall Comparison and Analysis

The results presented in Table 13 and Figure 8 provide valuable insights into the performance and reliability of different machine learning models and upsampling techniques across datasets with churn ratios ranging from 15% to 1%.

Evaluating machine learning models and upsampling techniques across datasets with varying churn rates provides valuable insights into addressing class imbalance. This study sheds light on how model performance evolves under increasing imbalance by analyzing Random Forest and XGBoost models paired with SMOTE, ADASYN, and GNUS across datasets with churn rates of 15%, 10%, 5%, and 1%. Key performance metrics such as F1 score, ROC-AUC, PR-AUC, Kappa, and MCC were examined to reveal critical patterns.

When ADASYN was applied, XGBoost demonstrated a remarkable ability to maintain and enhance its performance as churn rates decreased. For example, in the dataset with a 1% churn rate, XGBoost achieved an F1 score and Kappa of 80%, alongside a perfect

ROC-AUC of 100%. This improvement suggests that ADASYN's synthetic samples effectively enhanced XGBoost's capacity to learn meaningful patterns from the minority class. In contrast, Random Forest struggled under extreme imbalance. While it also achieved a perfect ROC-AUC at 1% churn, its F1 score dropped to 47%, indicating that the model's predictions were skewed toward the majority class. This discrepancy highlights the limitations of relying solely on ROC-AUC in highly imbalanced datasets, as it can obscure deficiencies in minority class prediction.

Table 13. The summary of all models' performance on four datasets with the highest values highlighted in bold.

Models/Metrics	F1 Score%	ROC-AUC%	PR-AUC%	Kappa%	MCC%	F1 Score%	ROC-AUC%	PR-AUC%	Kappa%	MCC%
	DS 1 (15% churn)					**DS 2** (10% churn)				
Tuned_RF_ADASYN	62	84	65	56	57	68	88	71	65	67
Tuned_XGB_ADASYN	63	**86**	72	56	56	71	88	73	68	69
Tuned_RF_SMOTE	64	85	67	58	59	67	88	74	64	66
Tuned_XGB_SMOTE	**68**	**86**	**73**	**63**	**63**	**74**	**89**	**77**	**72**	**73**
Tuned_RF_GNUS	61	81	65	56	57	60	86	64	55	55
Tuned_XGB_GNUS	61	85	68	56	58	56	84	65	50	51
	DS 3 (5% churn)					**DS 4** (1% churn)				
Tuned_RF_ADASYN	65	90	71	63	63	47	**100**	94	47	55
Tuned_XGB_ADASYN	70	90	76	69	69	80	**100**	95	80	80
Tuned_RF_SMOTE	70	92	72	68	69	56	**100**	93	55	62
Tuned_XGB_SMOTE	**75**	**93**	**79**	**74**	**74**	**92**	**100**	**97**	**92**	**92**
Tuned_RF_GNUS	57	90	62	54	55	27	92	20	25	33
Tuned_XGB_GNUS	68	**93**	77	66	66	74	99	86	74	74

SMOTE emerged as a particularly powerful technique, especially when paired with XGBoost. XGBoost consistently improved its performance as the churn rate decreased, reaching an F1 score and Kappa of 92% in the 1% churn dataset. These results underline the strong synergy between SMOTE's synthetic data generation and XGBoost's advanced learning algorithms. The model effectively leveraged the enriched training data to improve precision, recall, and overall reliability, even under extreme imbalance. While more robust with SMOTE than ADASYN, Random Forest showed limitations in handling severe imbalances. At a 1% churn rate, its F1 score fell to 56%, reflecting difficulties in balancing precision and recall under such conditions.

GNUS presented a mixed picture. For Random Forest, performance metrics such as F1 score and PR-AUC plummeted as the imbalance intensified. At a 1% churn rate, the F1 score dropped to 27% and the PR-AUC to just 20%, indicating that the model struggled to use GNUS-generated synthetic samples effectively. In contrast, XGBoost showed notable adaptability, improving its metrics as churn rates decreased. At 1% churn, XGBoost achieved an F1 score of 74% and a PR-AUC of 86%. This divergence suggests that GNUS, while less effective than SMOTE or ADASYN, still provided some benefit when combined with XGBoost's advanced algorithms.

Overall, XGBoost consistently outperformed Random Forest across all upsampling techniques and churn rates. Its ability to generalize and adapt to extreme imbalance sets it apart, particularly when paired with SMOTE. While Random Forest demonstrated resilience in moderately imbalanced datasets, its performance declined significantly under extreme imbalance, especially with GNUS. Moreover, the Friedman test identified statistically significant differences ($p < 0.05$) in key performance metrics, including F1 score,

PR-AUC, Kappa, and MCC, among the evaluated models. Additionally, the Nemenyi test confirmed that Tuned_XGB_SMOTE demonstrated a statistically significant performance advantage over Tuned_RF_GNUS.

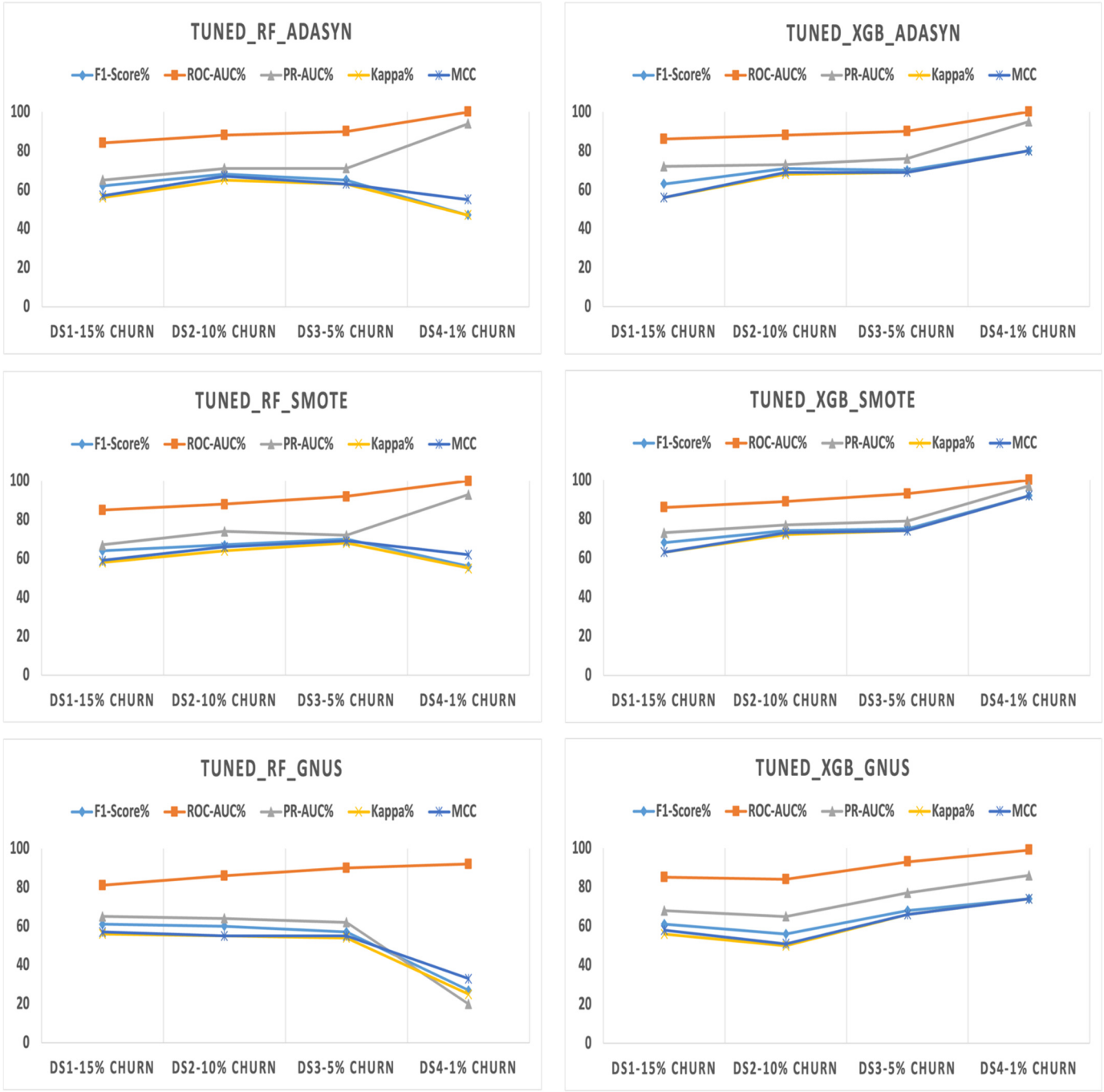


Figure 8. The performance of the models with upsampling techniques on datasets with different levels of imbalance.

These findings emphasize the importance of selecting appropriate metrics and techniques for evaluating imbalanced datasets. Metrics like F1 score, PR-AUC, Kappa, and MCC offered a more nuanced view of model performance than ROC-AUC alone, which often overstated a model's ability to predict minority class instances. Moreover, the results highlight the critical role of upsampling techniques in mitigating imbalance. SMOTE proved the most effective, providing consistently strong results, particularly for XGBoost. ADASYN delivered moderate improvements but struggled under extreme imbalance,

while GNUS was the least effective overall, though it showed some promise when paired with XGBoost.

The superior performance of XGBoost over Random Forest, particularly under extreme class imbalance, can be attributed to fundamental differences in their learning mechanisms. XGBoost employs Gradient Boosting, where trees are trained sequentially, each iteration correcting errors from the previous one. This iterative learning process enables XGBoost to focus more on misclassified minority class instances, dynamically adjusting its decision boundaries. In contrast, Random Forest constructs independent Decision Trees in parallel, aggregating their outputs through majority voting. While this approach enhances model stability, it does not inherently prioritize learning from the minority class, making it less effective in highly imbalanced datasets.

The interaction between SMOTE and model training strategies further explains this performance gap. Since SMOTE-generated samples are interpolations of existing minority class instances, XGBoost's iterative framework refines decision boundaries at each boosting step, helping the model better use synthetic samples. On the other hand, Random Forest treats synthetic samples equally with original data but lacks a mechanism to adjust decision boundaries dynamically, reducing its effectiveness in learning from synthetic minority examples.

Additionally, XGBoost benefits from L1 (Lasso) and L2 (Ridge) regularization, preventing overfitting to synthetic samples while preserving model generalization. Random Forest, lacking explicit regularization beyond bootstrapping, may overfit or struggle with synthetic minority instances. Furthermore, XGBoost's gradient-based split selection enhances its ability to separate minority instances. In contrast, Random Forest relies on static criteria like Gini impurity or entropy, which are less adaptive to imbalanced data distributions.

Ultimately, XGBoost's ability to iteratively adjust for class imbalance, adaptively weight instances, and refine decision boundaries allows it to outperform Random Forest, particularly in datasets where SMOTE-generated synthetic samples play a critical role in balancing class representation.

In conclusion, combining tuned XGBoost with SMOTE emerged as the most effective strategy for imbalanced datasets, achieving the highest scores across all evaluation metrics. This approach's success underscores the importance of leveraging advanced algorithms and effective upsampling methods to address the challenges of class imbalance. These findings provide valuable guidance for practitioners in churn prediction and similar classification tasks, demonstrating the potential to achieve reliable and accurate predictions even under challenging conditions of extreme imbalance.

5.8. Summary of Results

The analysis reveals that tuned XGBoost models, when paired with SMOTE or ADASYN upsampling techniques, consistently deliver the best predictive performance for customer churn across datasets with varying levels of imbalance (churn rates ranging from 15% to 1%), as depicted in Figure 9. These combinations achieve high scores across key metrics, including F1 score, ROC-AUC, PR-AUC, Cohen's Kappa, and MCC, demonstrating exceptional robustness and adaptability even in scenarios of extreme imbalance.

In contrast, tuned Random Forest models exhibit satisfactory performance under moderate imbalance conditions but show a marked decline in effectiveness as the level of imbalance intensifies, irrespective of the upsampling technique employed. Among the upsampling methods, SMOTE and ADASYN outperform GNUS, with their efficacy particularly pronounced when integrated with XGBoost.

To achieve accurate and reliable customer churn predictions across datasets with different degrees of imbalance, the combination of tuned XGBoost with SMOTE is strongly

recommended. This pairing demonstrates superior performance and resilience, making it an ideal choice for churn prediction tasks.

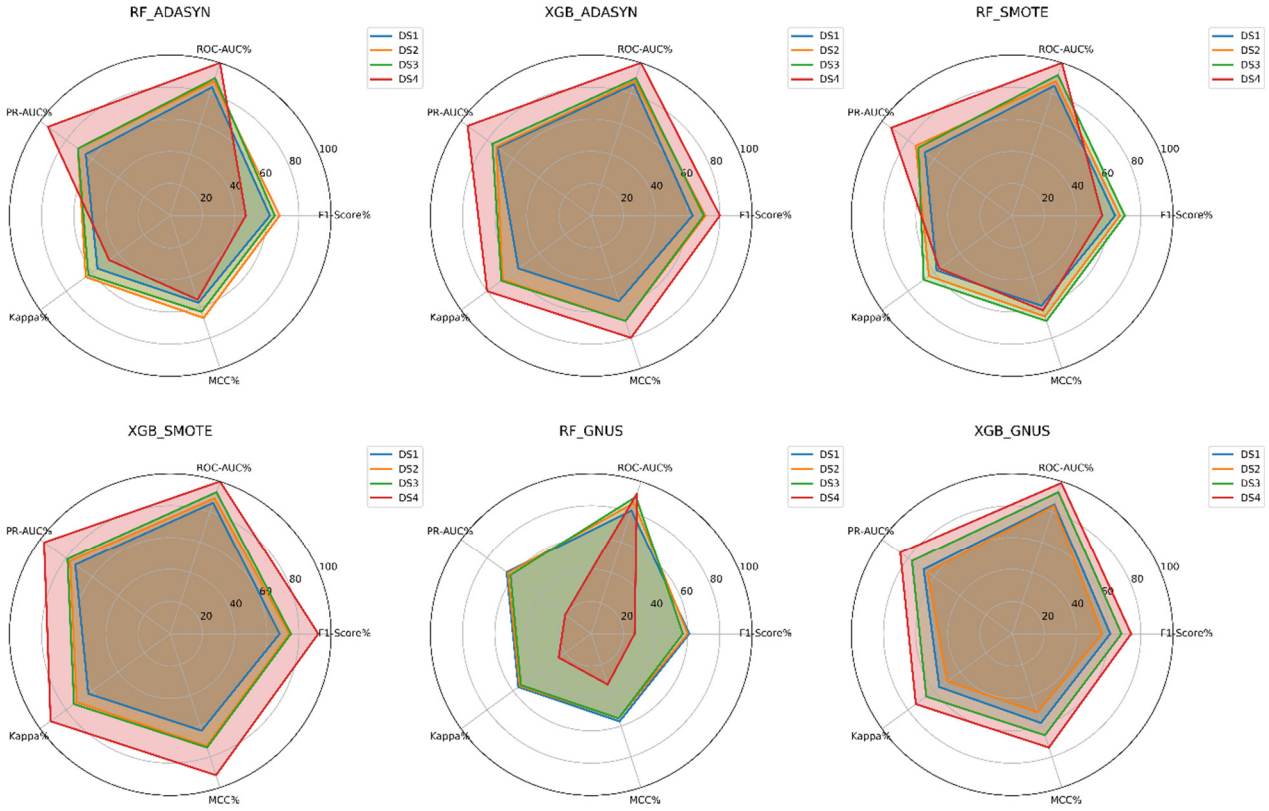


Figure 9. The radar chart of all the results in one view.

6. Conclusions

This study explored the interplay between supervised machine learning models—Random Forest and XGBoost—and various upsampling techniques, including SMOTE, ADASYN, and GNUS, to assess their combined impact on predictive performance across datasets with varying degrees of imbalance in customer churn rates, ranging from 15% to 1%. The central aim was to determine which model and upsampling technique combinations deliver the most accurate predictions in moderately imbalanced datasets and how extreme imbalance conditions influence their effectiveness.

The research was conducted on a robust IT infrastructure supported by efficient hardware and versatile software. The computational backbone of the study was an Apple MacBook Air equipped with an M2 chip, which facilitated the seamless execution of data-intensive tasks. Complementing this hardware were open-source Python libraries, such as Numpy, Pandas, Matplotlib, Seaborn: 0.13.2, Imbalanced-learn, and Scikit-learn. These tools played an integral role at every stage of the research process, from data preprocessing and visualization to model evaluation. Additionally, ChatGPT: 4o contributed to refining the language and ensuring clarity in the written content while data analysis and interpretation remained entirely independent of its assistance. Together, these resources laid the foundation for a rigorous and effective investigation.

This study highlights the importance of selecting appropriate combinations of machine learning models and upsampling techniques to address the challenges of imbalanced datasets in customer churn prediction. The findings reveal that for moderately imbalanced datasets, where churn rates range between 15% and 5%, XGBoost combined with SMOTE emerges as the most effective solution. This combination achieves remarkable predictive

accuracy, as demonstrated by its F1 scores reaching 73%, excellent ROC AUC and PR AUC values, and strong performance in metrics such as Matthews Correlation Coefficient and Cohen's Kappa. These results underscore its reliability and robustness in handling imbalanced data.

In scenarios of extreme imbalance, with churn rates as low as 1%, XGBoost paired with SMOTE or ADASYN continues to exhibit exceptional performance. It achieves F1 scores as high as 92% and perfect ROC AUC values, demonstrating its ability to maintain predictive strength even under challenging conditions. In contrast, Random Forest models, while adequate for moderately imbalanced datasets, show significant declines in key metrics such as F1 score, MCC, and Kappa when applied to highly imbalanced datasets. This decline is particularly pronounced when Random Forest is combined with GNUS, highlighting the superior robustness and adaptability of XGBoost in extreme class imbalance scenarios.

Among the upsampling techniques, SMOTE proves to be the most effective, particularly when paired with XGBoost. By generating synthetic minority class instances, SMOTE enables the model to capture underlying patterns better, enhancing overall predictive performance. While ADASYN also performs well with XGBoost across different imbalance levels, GNUS delivers inconsistent results and is less effective, particularly in combination with Random Forest.

The results of this study indicate that GNUS underperforms compared to SMOTE and ADASYN across most configurations, particularly when used with tree-based models like Random Forest and XGBoost. This performance gap can be attributed to key limitations in Gaussian noise injection as an upsampling strategy. Unlike SMOTE, which generates synthetic minority class instances based on nearest neighbor interpolation, GNUS applies random perturbations to existing instances without reinforcing meaningful decision boundaries. This can result in synthetic samples that fail to provide additional discriminatory power for the classifier, thereby limiting performance improvements.

Furthermore, tree-based models exhibit varying sensitivities to Gaussian noise. While boosting models like XGBoost can leverage sequential learning to adjust to noisy perturbations, Random Forest lacks an adaptive mechanism to refine decision boundaries dynamically, leading to a weaker impact of GNUS-based upsampling. Additionally, the random nature of noise injection in GNUS can cause synthetic samples to overlap with majority class instances, particularly in datasets with extreme imbalance, such as DS4 (1% churn), where even small misclassifications can significantly impact model performance.

While this study primarily focuses on customer churn prediction in the telecommunications industry, the challenges posed by class imbalance are not unique to this domain. Many real-world applications suffer from highly skewed class distributions, leading to biased model predictions. For instance, in healthcare, early disease diagnosis (e.g., detecting rare cancers) often involves datasets where positive cases are extremely scarce, necessitating effective upsampling techniques such as GNUS, SMOTE, and ADASYN to improve model sensitivity. Similarly, fraud detection constitutes a small fraction of all transactions, so applying robust resampling strategies to detect fraudulent activities accurately is crucial. In financial risk analysis, credit default prediction also encounters imbalanced datasets, where properly handling minority cases is vital for improving loan approval models. By demonstrating the effectiveness of GNUS in handling imbalance in churn prediction, our approach could be extended to these fields, warranting further exploration in future work.

In summary, tuned XGBoost with SMOTE consistently delivers outstanding predictive performance across datasets with moderate and extreme imbalance levels. This pairing effectively addresses the challenges of class imbalance by leveraging XGBoost's advanced learning capabilities and SMOTE's ability to create representative synthetic samples. As a result, it emerges as the optimal choice for accurate and reliable customer churn prediction

in imbalanced datasets. Furthermore, the statistical analyses using the Friedman test and Nemenyi post hoc comparisons provide robust evidence supporting these conclusions. The Friedman test revealed statistically significant differences ($p < 0.05$) in key performance metrics (F1 score, PR-AUC, Kappa, and MCC) among the evaluated models, while the Nemenyi test specifically confirmed that Tuned_XGB_SMOTE significantly outperforms Tuned_RF_GNUS. These statistical validations not only substantiate the superior performance of the XGBoost-SMOTE combination but also underscore its potential as a reliable framework for addressing imbalanced classification challenges across diverse application domains.

7. Future Work

Building upon the findings of this study, several promising directions for future research emerge, aiming to deepen the understanding and application of machine learning techniques for imbalanced datasets. One potential avenue is the exploration of advanced machine learning models. Investigating the capabilities of deep learning architectures, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), alongside ensemble methods beyond XGBoost, could provide valuable insights into their efficacy in addressing extreme class imbalance. These advanced models may offer innovative solutions for improving predictive accuracy and robustness in challenging scenarios.

Another intriguing direction involves the development and evaluation of hybrid sampling techniques. By combining the strengths of existing methods, such as SMOTE, ADASYN, and GNUS, with strategies like undersampling or cost-sensitive learning, researchers could create more balanced and informative synthetic samples. Such hybrid approaches can mitigate individual methods' limitations and enhance model performance across various levels of imbalance.

In addition to exploring novel sampling techniques, future work could extend the ensemble learning framework beyond XGBoost and Random Forest. Alternative models, such as Stacking, AdaBoost, CatBoost, and Hybrid Boosting approaches, may offer enhanced robustness against extreme class imbalance. These methods could be evaluated alongside hybrid upsampling strategies to assess their effectiveness in improving predictive performance. Moreover, meta-learning techniques could be employed to dynamically select the best resampling strategy and ensemble model based on dataset characteristics.

To enhance the effectiveness of GNUS, future research could explore feature-aware Gaussian noise injection, where noise is scaled based on feature importance, ensuring that key predictive features retain their structure. In contrast, less relevant features absorb more variance. Additionally, an adaptive noise scaling approach, similar to ADASYN, could increase noise diversity in underrepresented regions while minimizing unnecessary perturbations in well-represented areas. Another promising modification is a hybrid GNUS-SMOTE approach, where SMOTE is used to create structured synthetic instances, followed by controlled Gaussian noise injection to enhance diversity without disrupting class separability. Such modifications could help preserve the computational efficiency of GNUS while addressing its limitations in extreme class imbalance scenarios.

Applying the insights from this study to real-world datasets from domains such as healthcare, finance, and e-commerce represents an important step in validating the generalizability of the findings. Real-world applications would confirm the robustness of the proposed techniques, illuminate domain-specific challenges, and provide tailored solutions for addressing class imbalance in diverse contexts.

Additionally, the impact of advanced feature engineering and selection techniques on model performance warrants further investigation. Methods like Recursive Feature Elimination (RFE) and mutual information analysis could refine feature selection processes,

enhancing both the interpretability and effectiveness of machine learning models. By focusing on the most relevant features, these techniques could reduce noise and complexity, enabling more precise and actionable predictions.

Pursuing these research directions promises to advance the field of churn prediction and the broader domain of machine learning for imbalanced datasets. By fostering the development of more robust, reliable, and context-aware models, these efforts will contribute to overcoming the persistent challenges posed by class imbalance in various applications.

Author Contributions: Conceptualization, M.I.; Methodology, M.I. and A.B.; Software, M.I.; Validation, M.I.; Formal analysis, M.I.; Investigation, M.I.; Resources, M.I.; Data curation, M.I.; Writing—original draft preparation, M.I.; Writing—review and editing, A.B. and H.R.A.; Visualization, M.I.; Supervision, A.B. and H.R.A.; Project administration, M.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available in Customer Churn Prediction 2020, Version 1: [76].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. The Chartered Institute of Marketing. *Cost of Customer Acquisition versus Customer Retention*; The Chartered Institute of Marketing: Cookham, UK, 2010.
2. Eichinger, F.; Nauck, D.D.; Klawonn, F. Sequence Mining for Customer Behaviour Predictions in Telecommunications. In Proceedings of the Workshop on Practical Data Mining at ECML/PKDD, Berlin, Germany, 18 September 2006; pp. 3–10.
3. Ahmad, K.A.; Jafar, A.; Aljoumaa, K. Customer churn prediction in telecom using machine learning in big data platform. *J. Big Data* **2019**, *6*, 28. [CrossRef]
4. Caigny, D.; Arno; Coussement, K.; De Bock, K.W. A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. *Eur. J. Oper. Res.* **2018**, *269*, 760–772. [CrossRef]
5. Jadhav, R.J.; Pawar, U.T. Churn prediction in telecommunication using data mining technology. *IJACSA Edit.* **2011**, *2*, 17–19.
6. Radosavljevik, D.; van der Putten, P.; Larsen, K.K. The impact of experimental setup in prepaid churn prediction for mobile telecommunications: What to predict, for whom and does the customer experience matter? *Trans. MLDM* **2010**, *3*, 80–99.
7. Richter, Y.; Yom-Tov, E.; Slonim, N. Predicting Customer Churn in Mobile Networks Through Analysis of Social Groups. In Proceedings of the SIAM International Conference on Data Mining (SDM), Columbus, OH, USA, 29 April–1 May 2010; SIAM: Philadelphia, PA, USA; Volume 2010, pp. 732–741.
8. Amin, A.; Shah, B.; Khattak, A.M.; Moreira FJ, L.; Ali, G.; Rocha, A.; Anwar, S. Cross-company customer churn prediction in telecommunication: A comparison of data transformation methods. *Int. J. Inf. Manag.* **2019**, *46*, 304–319. [CrossRef]
9. Tsiptsis, K.; Chorianopoulos, A. *Data Mining Techniques in CRM: Inside Customer Segmentation*; John Wiley & Sons: Hoboken, NJ, USA, 2011.
10. Joudaki, M.; Imani, M.; Esmaeili, M.; Mahmoodi, M.; Mazhari, N. Presenting a New Approach for Predicting and Preventing Active/Deliberate Customer Churn in Telecommunication Industry. In Proceedings of the International Conference on Security and Management (SAM), Las Vegas, NV, USA, 18–21 July 2011.
11. Amin, A.; Al-Obeidat, F.; Shah, B.; Adnan, A.; Loo, J.; Anwar, S. Customer churn prediction in telecommunication industry using data certainty. *J. Bus. Res.* **2019**, *94*, 290–301. [CrossRef]
12. Shaaban, E.; Helmy, Y.; Khedr, A.; Nasr, M. A proposed churn prediction model. *J. Eng. Res. Appl.* **2012**, *2*, 693–697.
13. Khan, Y.; Shafiq, S.; Naeem, A.; Ahmed, S.; Safwan, N.; Hussain, S. Customers churn prediction using artificial neural networks (ANN) in telecom industry. *Int. J. Adv. Comput. Sci. Appl.* **2019**, *10*, 9. [CrossRef]
14. Ho, T.K. Random Decision Forests. In Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, Canada, 14–16 August 1995; IEEE: New York, NY, USA, 1995; Volume 1.
15. Chen, T.; Guestrin, C. Xgboost: A Scalable Tree Boosting System. In Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016.

16. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. Lightgbm: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30*; NeurIPS: Vancouver, BC, Canada, 2017.
17. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A.V.; Gulin, A. *CatBoost: Unbiased Boosting with Categorical Features. Advances in Neural Information Processing Systems 31*; NeurIPS: Vancouver, BC, Canada, 2018.
18. Xie, Y.; Li, X.; Ngai, E.W.; Ying, W. Customer churn prediction using improved balanced random forests. *Expert Syst. Appl.* **2020**, *41*, 5442–5455. [CrossRef]
19. Kumar, P.; Sharma, A. A review on churn prediction techniques in the telecom sector. *Telecommun. Syst.* **2021**, *76*, 519–537.
20. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
21. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. In Proceedings of the IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, 1–8 June 2008; pp. 1322–1328.
22. Branco, P.; Torgo, L.; Ribeiro, R.P. A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv. CSUR* **2016**, *49*, 31. [CrossRef]
23. Tomek, I. Two modifications of CNN. *IEEE Trans. Syst. Man Cybern.* **1976**, *6*, 769–772.
24. Wilson, D.L. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Trans. Syst. Man Cybern.* **1972**, *3*, 408–421. [CrossRef]
25. Amin, A.; Anwar, S.; Adnan, A.; Nawaz, M.; Howard, N.; Qadir, J.; Hawalah, A.; Hussain, A. Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study. *IEEE Access* **2016**, *4*, 7940–7957. [CrossRef]
26. Imani, M.; Ghaderpour, Z.; Joudaki, M.; Beikmohammadi, A. The Impact of SMOTE and ADASYN on Random Forest and Advanced Gradient Boosting Techniques in Telecom Customer Churn Prediction. In Proceedings of the 2024 10th International Conference on Web Research (ICWR), IEEE, Tehran, Iran, 24–25 April 2024.
27. Bergstra, J.; Bardenet, R.; Bengio, Y.; Kégl, B. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems 24*; NeurIPS: Vancouver, BC, Canada, 2011.
28. Snoek, J.; Larochelle, H.; Adams, R.P. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems 25*; NeurIPS: Vancouver, BC, Canada, 2012.
29. Amin, A.; Shehzad, S.; Khan, C.; Ali, I.; Anwar, S. Churn Prediction in Telecommunication Industry Using Rough Set Approach. In *New Trends in Computational Collective Intelligence*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 83–95.
30. Witten, I.H.; Frank, E.; Hall, M.A.; Pal, C.J. *Data Mining: Practical Machine Learning Tools and Techniques*; Elsevier Science & Technology: San Francisco, CA, USA, 2016.
31. Kumar, A.; Jain, M. *Ensemble Learning for AI Developers: Learn Bagging, Stacking, and Boosting Methods with Use Cases*; Apress: New York, NY, USA, 2020.
32. Van Wezel, M.; Potharst, R. Improved customer choice predictions using ensemble methods. *Eur. J. Oper. Res.* **2007**, *181*, 1. [CrossRef]
33. Ullah, I.; Raza, B.; Malik, A.K.; Imran, M.; Islam, S.U.; Kim, S.W. A churn prediction model using random forest: Analysis of machine learning techniques for churn prediction and factor identification in telecom sector. *IEEE Access* **2019**, *7*, 60134–60149. [CrossRef]
34. Haddadi, S.J.; Farshidvard, A.; dos Santos Silva, F.; dos Reis, J.C.; da Silva Reis, M. Customer churn prediction in imbalanced datasets with resampling methods: A comparative study. *Expert Syst. Appl.* **2024**, *246*, 123086. [CrossRef]
35. Matthias, B.; Delaere, L. Ensemble methods in customer churn prediction: A comparative analysis of the state-of-the-art. *Mathematics* **2023**, *11*, 1137. [CrossRef]
36. Manzoor, A.; Qureshi, M.A.; Kidney, E.; Longo, L. A Review on Machine Learning Methods for Customer Churn Prediction and Recommendations for Business Practitioners. *IEEE Access* **2024**, *12*, 70434–70463. [CrossRef]
37. Lalwani, P.; Mishra, M.K.; Chadha, J.S.; Sethi, P. Customer churn prediction system: A machine learning approach. *Computing* **2022**, *104*, 271–294. [CrossRef]
38. Vafeiadis, T.; Diamantaras, K.I.; Sarigiannidis, G.; Chatzisavvas, K.C. A comparison of machine learning techniques for customer churn prediction. *Simul. Model. Pract. Theory* **2015**, *55*, 1–9. [CrossRef]
39. Imani, M.; Arabnia, H.R. Hyperparameter optimization and combined data sampling techniques in machine learning for customer churn prediction: A comparative analysis. *Technologies* **2023**, *11*, 167. [CrossRef]
40. Seema, B.; Gupta, G. Performance evaluation of various classification techniques for customer churn prediction in e-commerce. *Microprocess. Microsyst.* **2022**, *94*, 104680.
41. Linoff, G.S.; Berry, M.J. *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management*; John Wiley & Sons: Hoboken, NJ, USA, 2011.
42. Doe, J.; Smith, A. On the synergy of multiple models in ensemble learning. *J. Mach. Learn. Res.* **2020**, *21*, 1–35.

43. Roe, P.; Lee, Y. Enhancing predictive performance through model diversity in ensemble learning. *Mach. Learn.* **2019**, *108*, 1081–1098.
44. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
45. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
46. Ramraj, S.; Uzir, N.; Sunil, R.; Banerjee, S. Experimenting XGBoost algorithm for prediction and classification of different datasets. *Int. J. Control. Theory Appl.* **2016**, *9*, 651–662.
47. Li, J.; An, X.; Li, Q.; Wang, C.; Yu, H.; Zhou, X.; Geng, Y.A. Application of XGBoost algorithm in the optimization of pollutant concentration. *Atmos. Res.* **2022**, *276*, 106238. [CrossRef]
48. Wyner, A.J.; Olson, M.; Bleich, J.; Mease, D. Explaining the success of adaboost and random forests as interpolating classifiers. *J. Mach. Learn. Res.* **2017**, *18*, 1–33.
49. Polikar, R. Ensemble based systems in decision making. *IEEE Circuits Syst. Mag.* **2006**, *6*, 21–45. [CrossRef]
50. Zhang, H.; Wang, M. Ensemble machine learning: Methods and applications. *Front. Bioeng. Biotechnol.* **2020**, *8*, 100.
51. Brown, I.; Mues, C. An experimental comparison of classification algorithms for imbalanced credit scoring datasets. *Expert Syst. Appl.* **2012**, *39*, 3446–3453. [CrossRef]
52. Smith, J.T.; Doe, A.R. Random Forests in Bioinformatics: Applications and Advances. *BioData Min.* **2022**, *15*, 11.
53. Williams, H.P.; Jones, D.C. Enhancing Financial Risk Assessment with Random Forests. *J. Financ. Econom.* **2021**, *19*, 567–588.
54. Liu, F.; Zhang, G.; Wong, K.Y. Optimizing Random Forest for High-dimensional Data Analysis. *Comput. Stat. Data Anal.* **2023**, *158*, 107159.
55. Patel, R.N.; Singh, S.K. Feature Selection Innovations in Random Forest Algorithms: A Comparative Study. *Expert Syst. Appl.* **2024**, *170*, 114430.
56. Zhang, Y.; Haghani, A. A gradient boosting method to improve travel time prediction. Transportation Research Part C. *Emerg. Technol.* **2015**, *58*, 308–324. [CrossRef]
57. Miroslav, K.; Matwin, S. Addressing the curse of imbalanced training sets: One-sided selection. In *Icml*; CiteSeerX: Old Main, PA, USA, 1997; Volume 97, p. 1.
58. Krawczyk, B. Learning from imbalanced data: Open challenges and future directions. *Prog. Artif. Intell.* **2016**, *5*, 221–232. [CrossRef]
59. Fernández, A.; García, S.; Galar, M.; Prati, R.C.; Krawczyk, B.; Herrera, F. *Learning from Imbalanced Data Sets*; Springer: Berlin/Heidelberg, Germany, 2018.
60. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. *Adv. Intell. Comput.* **2005**, *3644*, 878–887.
61. Ruslan, S.; Hinton, G. Deep Boltzmann Machines. In *Artificial Intelligence and Statistics*; PMLR: New York, NY, USA, 2009.
62. Connor, S.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48.
63. EAPA, B.G.; Prati, R.C.; Monard, M.C. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explor. Newsl.* **2004**, *6*, 20–29.
64. Bergstra, J.S.; Bengio, Y. Random Search for Hyper-Parameter Optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
65. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. OPTUNA: A Next-generation Hyperparameter Optimization Framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019.
66. OPTUNA Documentation. OPTUNA: A Hyperparameter Optimization Framework. Available online: https://optuna.org/ (accessed on 15 January 2024).
67. Davis, J.; Goadrich, M. The relationship between Precision-Recall and ROC curves. In Proceedings of the 23rd international conference on Machine learning, Pittsburgh, PA, USA, 25–29 June 2006.
68. Powers, D.M.W. Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. *J. Mach. Learn. Technol.* **2011**, *2*, 6.
69. Matthews, B.W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Et Biophys. Acta BBA-Protein Struct.* **1975**, *405*, 442–451. [CrossRef]
70. Davide, C.; Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genom.* **2020**, *21*, 6.
71. Jacob, C. A coefficient of agreement for nominal scales. *Educ. Psychol. Meas.* **1960**, *20*, 37–46.
72. McHugh, M.L. Interrater reliability: The kappa statistic. *Biochem. Medica* **2012**, *22*, 276–282. [CrossRef]
73. Viera, A.J.; Garrett, J.M. Understanding interobserver agreement: The kappa statistic. *Fam. Med.* **2005**, *37*, 360–363.
74. Warrens, M.J. Cohen's kappa is a weighted average. *Stat. Methodol.* **2011**, *8*, 473–484. [CrossRef]
75. Saito, T.; Rehmsmeier, M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PLoS ONE* **2015**, *10*, e0118432. [CrossRef] [PubMed]

76. Christy, R. Customer Churn Prediction 2020, Version 1. Available online: https://www.kaggle.com/code/rinichristy/customer-churn-prediction-2020 (accessed on 20 January 2024).
77. Harris, C.R.; Millman, K.J.; Van Der Walt, S.J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N.J.; et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362. [CrossRef] [PubMed]
78. McKinney, W. Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June–3 July 2010.
79. Hunter, J.D. Matplotlib: A 2D Graphics Environment. *Comput. Sci. Eng.* **2007**, *9*, 90–95. [CrossRef]
80. LemaÃŽtre, G.; Nogueira, F.; Aridas, C.K. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *J. Mach. Learn. Res.* **2017**, *18*, 1–5.
81. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.
82. Denscombe, M. EBOOK: The Good Research Guide: For Small-Scale Social Research Projects. McGraw-Hill Education: London, UK, 2017.
83. Liu, H.; Dougherty, E.R.; Dy, J.G.; Torkkola, K.; Tuv, E.; Peng, H.; Ding, C.; Long, F.; Berens, M.; Parsons, L.; et al. Evolving feature selection. *IEEE Intell. Syst.* **2005**, *20*, 64–76. [CrossRef]
84. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
85. García, S.; Fernández, A.; Luengo, J.; Herrera, F. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Inf. Sci.* **2010**, *180*, 2044–2064. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.