Journal of Big Data

RESEARCH **Open Access**

CTGAN-ENN: a tabular GAN-based hybrid sampling method for imbalanced and overlapped data in customer churn prediction

I Nyoman Mahayasa Adiputra[1] and Paweena Wanchai[1*]

*Correspondence:
wpaweena@kku.ac.th

[1] College of Computing, Khon Kaen University, Khon Kaen, Thailand

Abstract
Class imbalance is one of many problems of customer churn datasets. One of the common problems is class overlap, where the data have a similar instance between classes. The prediction task of customer churn becomes more challenging when there is class overlap in the data training. In this research, we suggested a hybrid method based on tabular GANs, called CTGAN-ENN, to address class overlap and imbalanced data in datasets of customers that churn. We used five different customer churn datasets from an open platform. CTGAN is a tabular GAN-based oversampling to address class imbalance but has a class overlap problem. We combined CTGAN with the ENN undersampling technique to overcome the class overlap. CTGAN-ENN reduced the number of class overlaps by each feature in all datasets. We investigated how effective CTGAN-ENN is in each machine learning technique. Based on our experiments, CTGAN-ENN achieved satisfactory results in KNN, GBM, XGB and LGB machine learning performance for customer churn predictions. We compared CTGAN-ENN with common over-sampling and hybrid sampling methods, and CTGAN-ENN achieved outperform results compared with other sampling methods and algorithm-level methods with cost-sensitive learning in several machine learning algorithms. We provide a time consumption algorithm between CTGAN and CTGAN-ENN. CTGAN-ENN achieved less time consumption than CTGAN. Our research work provides a new framework to handle customer churn prediction problems with several types of imbalanced datasets and can be useful in real-world data from customer churn prediction.

Keywords: Hybrid sampling method, Over-sampling, Under-sampling, CTGAN, ENN, Machine learning, Customer churn prediction

Introduction

Customer churn prediction uses data analysis and predictive algorithms to determine which clients are most likely to quit a company or cease utilizing its goods or services. By anticipating customer turnover, the business may take proactive measures to retain key clients and optimize its marketing and retention strategy. Businesses in a corporation can increase revenue by developing an accurate prediction of client turnover habits and providing retention solutions [1].

The task of classifying customers presents significant hurdles due to class imbalance. Class imbalance denotes a situation where one or more classes are much more prevalent than the others [2]. Imbalanced classes can cause misleading performance metrics and increase false negative predictions. False negative occurs when the model fails to identify a customer who is likely to churn. That condition can have a significant impact on the implementation of retention strategies because customers who churn are predicted as not churn.

The data level-solving approach to the imbalance class problem is one of a solution to address the issue [3]. The data level method focuses on the preprocessing stage and is independent of the machine learning prediction method. A type of generative model based on a neural network called Generative Adversarial Networks, or GAN for short, is intended to generate realistic samples of entities [4].

Class imbalance issues have been addressed with GAN-based oversampling; nevertheless, class imbalance is not the only difficulty with customer churn data; class overlap is another issue that GAN is unable to resolve. The degree of similarity between instances of distinct classes is known as class overlap. Training a classifier that can distinguish between the classes with accuracy is challenging due to the class overlap requirement. Poor conditions for training data can result in machine learning performance factors [5].

The latest study works on GAN-based hybrid sampling [2], that works overcome class overlap and achieve the best result compared to other oversampling and hybrid sampling methods. We proposed a research framework using tabular type of GAN called CTGAN, in tabular data, category variables, numerical values, and specific relationships between columns are frequently present along with other organizational features. Tabular data has a specific structure and limitations that traditional GANs are not well adapted to handle. Tabular GANs can be configured to meet the difficulties and limitations posed by structured data, making them an invaluable tool for tabular data creation tasks. We did an experiment focused on customer churn problems and tried the result of the method in both classical and ensemble machine learning. This experiment's objective is to observe the effectiveness difference in evaluation metrics and time execution of all algorithms. Beside the data-level solutions, we compared CTGAN-ENN with algorithm-level solution called cost-sensitive learning.

Our research objectives are highlighted as follows:

- Customer churn prediction datasets that have high dimensional features and different imbalance ratios.
- How CTGAN-ENN can handle class imbalance and class overlap in customer churn datasets.
- The effectiveness of CTGAN-ENN in different machine learning algorithms within several evaluation metrics in customer churn prediction.
- Comparison between data-level and algorithm-level solutions with CTGAN-ENN performance.
- Algorithm time consumption between original CTGAN and CTGAN-ENN in customer churn prediction.

The novelty of this paper is this is the first work that investigates a combination of tabular GAN (CTGAN) and an under-sampling technique (ENN). This work shows a framework for handling high dimensional features in customer churn prediction with class imbalance and class overlap. Moreover, this work evaluates how effective CTGAN-ENN is both in evaluation metrics and algorithm time consumption. This paper scope is only on data-level solution, because this study wants to prove a tabular GAN hybrid sampling method achieved better performance than non-tabular GAN hybrid sampling method and another classical hybrid sampling method in customer churn predictions data. Furthermore, we provided a comparison between CTGAN-ENN and cost-sensitive learning in several machine-learning algorithms.

Related works

Customer churn prediction

Machine learning algorithms such as K-nearest Neighbor, naïve Bayes, and decision trees have been widely used on customer churn predictions as supervised learning in recent years. Classical machine learning methods commonly have a not satisfied result in performance. Ensemble approaches are used for classical machine learning problems, such as Random Forest, AdaBoost, Gradient Boosting, and XGBoost [6]. The main problem with customer churn predictions is imbalanced data. A deep learning algorithm is proposed using Deep & Cross Network (DCN) to learn latent features from customer churn prediction and Asymmetric Loss Function (ASL) to handle imbalanced data [1]. Based on recent studies, machine learning and deep learning for customer churn prediction have not reached outperforming results; the other approach is the data-level solution. This method works on the data preprocessing stage, using traditional oversampling techniques such as SMOTE and ADAYSN or hybrid sampling techniques using SMOTE + under-sampling methods [6, 7]. Traditional sampling methods have a problem in representing synthetic data. A deep learning-based sampling method can be used to tackle this problem. One of the common methods is Generative Adversarial Network (GAN); GAN produces synthetic data based on deep learning algorithms and represents real data better than traditional oversampling methods [8]. The latest technology of customer churn predictions uses a GAN-based hybrid sampling method to overcome a class overlap problem that GAN produced; this method resulted in an outperform in AUC, F1-Score, and G-mean metrics compared to other sampling methods [2, 9]. Class overlaps problem and hybrid sampling method are explained empirically in the next subsects, Class overlap problem and Hybrid sampling methods below.

Class overlap problem

Class overlap in machine learning makes it challenging to develop a reliable classifier that can differentiate between the classes. Factors contributing to class overlap include similar data, sparse distribution, and noise. Figure 1 shows oversampling results using CTGAN in all datasets. It's worth noting that all datasets in this research have overlap issues, evident from the lack of clear distance between classes and stacked data between classes.

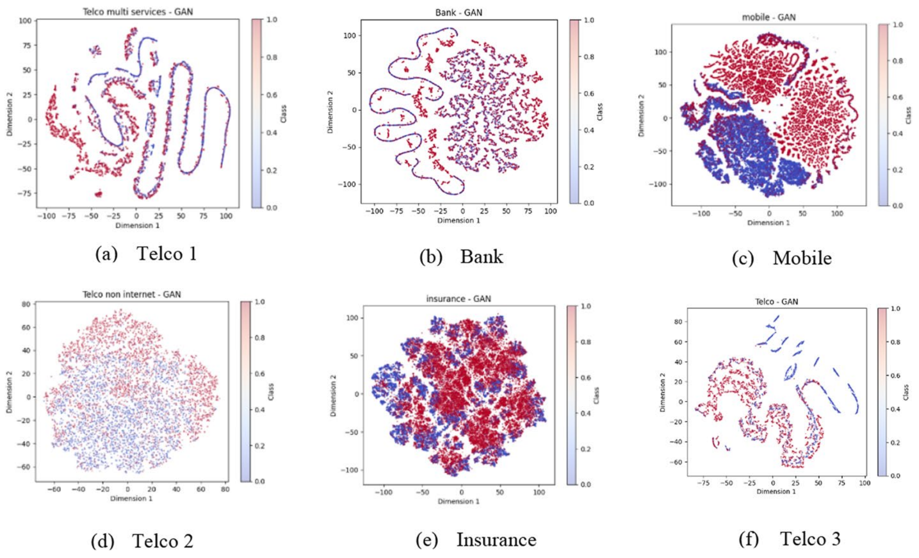

Fig. 1 Class overlap in customer churn datasets

Hybrid sampling methods

In the last few years, there has been research on customer churn prediction and several research for managing class imbalance. Sampling approaches, which are further classified as under-sampling, over-sampling, and hybrid sampling are well-known strategies for addressing class imbalance. Some studies have tried hybrid sampling. Sáez et al. [10] proposed the extension of SMOTE with a noise data filter called an Iterative-Partitioning Filter (IPF) for handling not only class imbalanced problems but also one of the data distribution problems, which is noisy data. The results proved their method performs better than existing techniques. Besides class imbalance, class overlap is another problem in the customer churn dataset. Vuttipittayamongkol [11] proposed an NCR-based under-sampling architecture to eliminate any possible overlapping data to address class imbalance in binary datasets. Their approach is centered on the under-sampling combination, and the experimental results show significant improvements in sensitivity.

Geiler et al. [6] investigated an effective strategy for churn prediction with SMOTE hybrid sampling in several machine learning algorithms, including ensemble machine learning. They compared SMOTE—Random under-sampling, SMOTE—NCR, and SMOTE—Tomek links. Based on their experiment, SMOTE—NCR outperformed in 2 datasets and SMOTE—Tomek links achieved the best AUC in one dataset. Another SMOTE hybrid method proposed by Zhaozhao Xu et al. [12] proposed M-SMOTE with ENN combination and experimented on a RF algorithm. Main objective of this work is to handle class imbalance from medical data with M-SMOTE, while ENN is used to handle the misclassified data. Compared to other comparable approaches, the outcome of this work in ten medical datasets is more encouraging, and they achieved 99.5% in the F-1 score metric.

Besides the traditional oversampling method, there is a neural network-based oversampling method called GAN (Generative Adversarial Network). Ding et al. [9]

proposed RGAN-EL to train the GAN to prioritize the class overlapping region during sample distribution fitting. They experimented with 41 imbalanced datasets. They compared the proposed research framework with several other oversampling methods, and the result showed a promising result from RGAN-EL. The limitation of this work is they are not using high-dimensional data in complex fields. Most of the datasets consist of less than ten dimensions.

Zhu et al. [2] developed a new hybrid sampling method with GAN and adaptive neighborhood-based weighted under-sampling (ANWU) in customer classification datasets. The ANWU method is used for handling class overlaps by removing generated instances and the original majority of class instances. They used KNN, DT, RF and GBM for their experiment. Compared to existing benchmark approaches, the GAN-based hybrid sampling method performs better in accuracy and profit-based evaluation criteria.

Cost-sensitive learning

Cost-sensitive learning is a paradigm within machine learning that focuses on incorporating the varying costs associated with different types of errors or misclassifications into the learning process. In traditional machine learning algorithms, the objective is typically to minimize overall classification error without considering the potential differences in the consequences or costs of different types of misclassifications.

Cost in cost-sensitive learning represented as a cost matrix shown in Fig. 2 [13]. The cost of false positives is C_{10} and the cost of false negative is C_{01}, in classification problem adjusting the cost based on objective of prediction is important. For example, if we want to build a machine learning that predicts customer churn, avoid customer that predicted not churn but churn or false negative is the priority of classification objective. Therefore, give C_{01} more cost than the other matrix can be one of the solutions in algorithm-level.

Machine learning algorithm

We used two types of machine learning to experiment with the classification task in customer churn prediction after the CTGAN-ENN result. The first type is classical machine learning, known as KNN, DT, and NB. The second type is ensemble machine learning, such as XGB, RF, and GBM. These algorithms are explained in this section.

K-nearest neighbor (KNN)

A case-based learning approach called K-Nearest Neighbor retains all the training data for categorization [14]. KNN is a well-known machine learning method that may be used for both regression and classification applications. Since the approach is instance-based and non-parametric, it makes no underlying assumptions about how the data are distributed. The idea behind KNN is that similar data points should lead to similar outcomes. In other words, a new data point that must be classified or predicted, you may

	actual negative	actual positive
predict negative	$C(0,0) = c_{00}$	$C(0,1) = c_{01}$
predict positive	$C(1,0) = c_{10}$	$C(1,1) = c_{11}$

Fig. 2 Cost-sensitive learning cost matrix

utilize the labels (for classification) or values (for regression) of the K nearest data points (neighbors) in the training dataset to create predictions.

Decision tree (DT)

A recursive partition of the instance space is represented by a decision tree classifier. It consists of nodes that divide the instance space into two or more subspaces based on a discrete function over the input attributes. When the root, or root node of the tree, occupies the entire area, the first split occurs.

The nodes that come after are either leaf nodes, which show the final categorization, or internal nodes, which have a predecessor and multiple successors [15]. When it comes to classification jobs, any new data point that travels along the path to a leaf node will be projected to belong to the majority class in that leaf. The mean or median of the target variable in that node is usually the leaf node value for regression tasks.

Naïve bayes (NB)

The Naïve Bayes algorithm relies on a probabilistic approach to perform classification tasks. It operates under the premise that a feature's existence in a class is independent of the existence of another feature in the same class [16]. The "naïve" assumption of feature independence, which simplifies the computations but might not hold true in all real-world situations, is the foundation of the Bayes theorem.

XGBoost (XGB)

XGboost is a weighted quantile sketch and sparsity-aware algorithm for approximate tree learning. To create a scalable tree-boosting system, XGBoost offers information on cache usage patterns and data compression [17]. XGBoost starts by creating a weak learner, which is a simple model that can make some predictions about the data, and then creates a new tree that is trained to correct the errors of the weak learner.

Random forest (RF)

As a kind of bagging technique, random forest constructs several models using various data subsets and then aggregates the forecasts from each model to produce a final prediction. In order for a random forest to function, a set of decision trees that grow in a number of randomly selected data subspaces are combined to create a prediction ensemble. Every tree in the collection is made by first selecting a subset of input coordinates at each node at random (which are there after referred to as features or variables). Based on these features, the training set determines the proper split [18].

Gradient boosting machine (GBM)

GBM is an ensemble forward learning model works by strongest predictor is chosen once all the weaker ones have been eliminated. This revised decision tree method compares each successor to the others, using the structure score, gain computations, and ever-finer approximations to produce a set of the tree's most satisfying structures [19]. To achieve a more accurate and complete model, GBM combines the predictions of several inaccurate models, typically decision trees. It accomplishes this by gradually training a series of decision trees, each one intended to fix the mistakes produced by the one before it.

Light gradient-boosting machine (LGB)

Light Gradient-Boosting Machine in short LightGBM is a novel algorithm from GBDT (Gradient Boosting Decision Tree), the purpose of LightGBM is reducing features by its information gain [20]. LightGBM works by parallel voting decision tree algorithm. It is designed to maximize parallel learning by reducing memory usage, accelerating the training process, and combining sophisticated network connectivity. Partition the training data among several machines, then carry out the local voting to determine the top-k attributes and the global voting to determine the top-two-k attributes for each iteration [20].

Summary

We discovered several kinds of research about hybrid sampling methods. Most recent work has used traditional oversampling techniques like SMOTE [6, 10, 12]. This technique can cause less data diversity in oversampling results. An oversampling technique that is based on a neural network can be the solution to sampling diversity. A common neural network technique for oversampling is the Generative Adversarial Network (GAN). A recent study suggested using GAN-based hybrid sampling to get around the GAN's class overlap problem. The latest work on GAN-based hybrid sampling has some limitations. The first limitation is that they do not use the tabular type of GAN for tabular data problems [2, 9]. The second limitation is that they are not measuring the time execution of the algorithms, which can be a good insight for real-world implementation. The remaining problem of the latest study about the GAN-based hybrid sampling method is that it does not involve a tabular-based GAN on tabular data cases.

In this research, we proposed a research framework that includes a GAN-based hybrid sampling scheme and used an ENN under-sampling combination to address the class overlap and tabular GAN type (CTGAN) to handle tabular data. Our work is focused on customer churn prediction problems with time execution insight in experiments. Hopefully, this method can be useful in real-world cases and used for any company that wants to implement churn prediction in their campaign strategies.

Methodology

Data preprocessing using the CTGAN-ENN approach is the first of two phases in our proposed research framework, as seen in Fig. 3 The input of imbalanced dataset divided by minority and majority classes. The minority class is oversampled by GAN using CTGAN [21], and it produces generated data within a balanced number as the majority class. The next process is concatenating the majority and generated data.

CTGAN can handle several challenges that are different from the traditional GAN model. First is a mixed data type because real-world data consists of a mixed data type between discrete and continuous. The second is highly imbalanced categorical columns. Customer churn prediction datasets used in this research have a high imbalanced ratio [21].

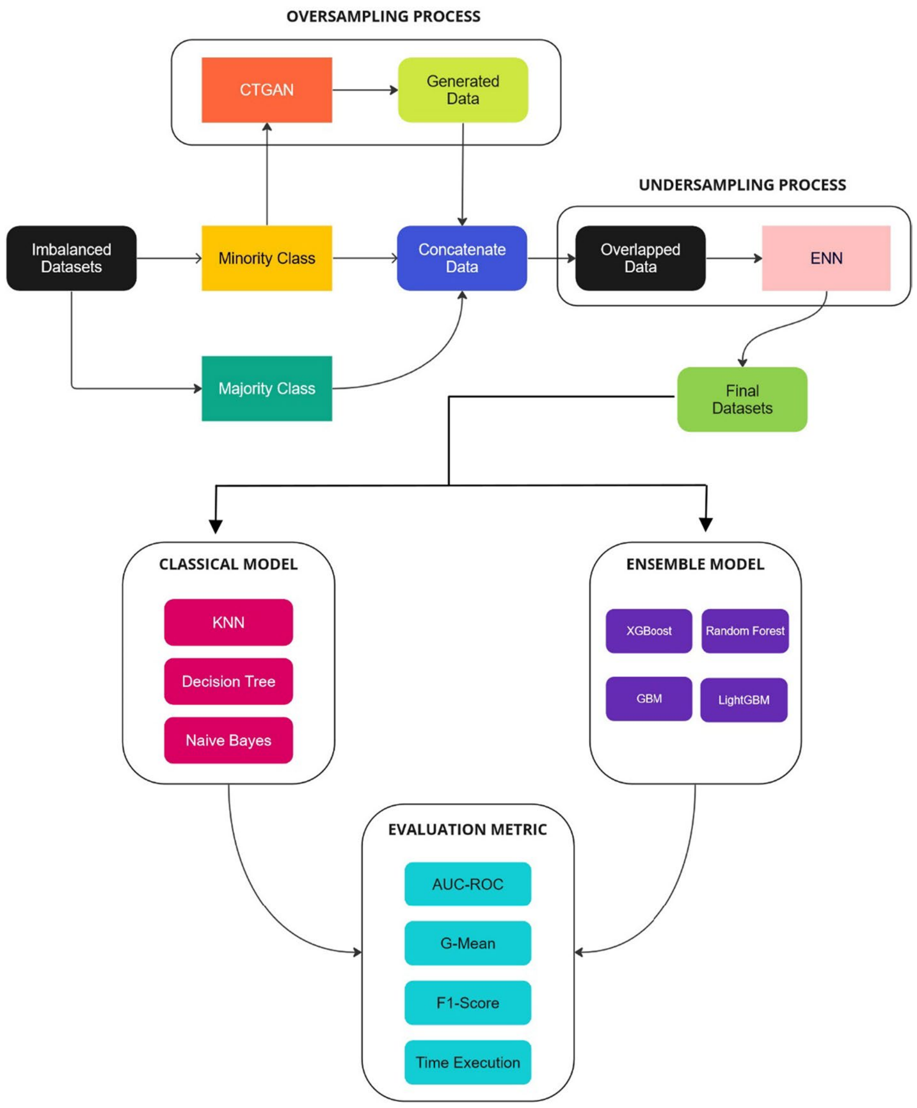

Fig. 3 Proposed research framework

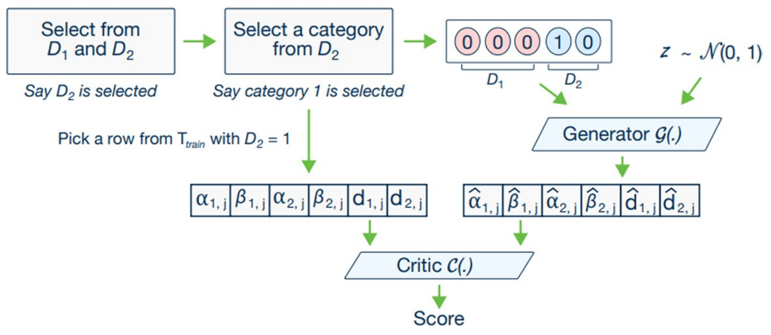
Select from D_1 and D_2
Say D_2 is selected
Select a category from D_2
Say category 1 is selected
0 0 0 1 0
D_1
D_2
$z \sim \mathcal{N}(0, 1)$
Pick a row from T_{train} with $D_2 = 1$
Generator $\mathcal{G}(.)$
$\alpha_{1,j}$ $\beta_{1,j}$ $\alpha_{2,j}$ $\beta_{2,j}$ $d_{1,j}$ $d_{2,j}$
$\hat{\alpha}_{1,j}$ $\hat{\beta}_{1,j}$ $\hat{\alpha}_{2,j}$ $\hat{\beta}_{2,j}$ $\hat{d}_{1,j}$ $\hat{d}_{2,j}$
Critic $\mathcal{C}(.)$
Score

Fig. 4 CTGAN original framework

Figure 4 is the CTGAN workflow, assuming data has two features D_1 and D_2, and we want to oversample the D_2 feature and pick $D_2 = 1$. Select all the row from D_2 that has 1 in value and let it be train data T_{train}. The result of generator compared with T_{train} resulted a critic score; The critic score estimated the distance between the learned conditional distribution and the conditional distribution of real data. That process is called training-by-sampling, which is better than a random value from traditional GAN.

The ENN technique is effective for removing instances so that it can handle class overlap. Through the consideration of its k-nearest neighbors who are members of the other class, ENN selectively eliminates data that do not belong to majority class [11]. The details of ENN algorithm are presented in Algorithm 1. The input is concatenated data from CTGAN and original data$D$, the number of nearest neighbors is $k = 3$, the majority samples D_{maj} is dependent on the datasets and the initial under-sampling rate is $R = 1$.

Algorithm 1 Edited Nearest Neighbor (ENN)

```
Input: Concatenated data $D$, number of nearest neighbors $k$, majority samples $D_{maj}$, Initial under-sampling rate $R$
Output: Final dataset $D$’
1: Initialize the under-sampling rate $R$, the value of $R_{del} = R * D_{min}$
2: Set the number of deleted samples $C_{del} = 0$
3: Search the KNN samples of $D_{maj}$ and save to indexes in $K_{maj}$[ ]
4: Select $k$ samples closest to $D_{maj}$
5: Compare the label of $D_{maj}$ with the class labels of its neighbor samples. Remove the label if
$D_{maj}$ does not match the labels of at least two of its nearest neighbor samples.
6: Increment $C_{del}$ by 1. If $C_{del}$ is less than $R_{del}$, return to Step 3 to process the next neighbor samples.
Otherwise, the operation for $Di_{maj}$ is completed.
7: To get the first balanced dataset $D'$, remove the deleted samples from dataset $D$.
```

The under-sampling method removed overlap instances from concatenate data. This stage result is the final customer churn dataset and is ready to process in machine learning prediction. The result of the effectiveness of the CTGAN-ENN method in this research is shown in the experiment result section.

After forming the final datasets, the second phase of our proposed research framework experimented with two types of machine learning on the customer churn datasets. The first type of model used KNN, Decision Trees, and Naïve Bayes, while the second type was an ensemble model that used XGBoost, Random Forest, GBM and LightGBM. To evaluate the models, we used four different criteria: AUC-ROC, G-Mean, F-1 Score, and algorithm execution time.

Algorithm 2 Pseudo-code of Proposed Framework

```
Input: $Datasets_m$, $classicalML_m$, $ensembleML_m$
Output: $finalDataset_m, AUC_m, Fmeassure_m, Gmean_m, Time_m$
1: Split the $Datasets_m$ into majority and minority data
2: Process CTGAN on minority data
3: Concatenate the majority, minority, and generated data by CTGAN
4: Process ENN as mentioned in algorithm 1, resulted $finalDataset_m$
5: Process $classicalML_m$ and $ensembleML_m$ on $finalDataset_m$
6: Calculate the $AUC_m, Fmeassure_m, Gmean_m, Time_m$ on $classicalML_m$ and $ensembleML_m$
```

Algorithm 2 is pseudo-code of proposed framework, attempts to address the problem of unbalanced datasets in machine learning by putting forth a thorough strategy that includes phases for data pretreatment, augmentation, training, and evaluation. To resolve the underlying class imbalance, the dataset is first split into majority and minority classes. The creation of artificial minority instances via the CTGAN technique which mimics the minority class's distribution by utilizing generative adversarial networks is a crucial next step. This augmentation technique gives the model a more balanced representation of the data, which attempts to alleviate the problem of class imbalance. After augmentation, the original majority and minority subsets are combined with the augmented minority data to create a consolidated dataset. The approach uses the Edited Nearest Neighbors (ENN) algorithm to further refine the dataset and improve its quality. By removing instances that are noisy or borderline, ENN enhances the final dataset's discriminative power. This cleaned dataset (called *finalDataset*$_m$) is used as the basis for the model training and assessment that comes after.

The next step balanced final dataset ready to apply ensemble learning (*ensembleML*$_m$) and classical machine learning (*classicalML*$_m$) approaches. To verify our proposed method performance, a variety of modeling paradigms can be explored thanks to this dual approach. Finally, performance measurements like Area Under the Curve (AUC), F-measure, and G-mean are used to assess how effective the trained models are. With AUC representing overall discriminative power, F-measure reflecting the trade-off between precision and recall, and G-mean evaluating the model's ability to manage class imbalance, these metrics shed light on the models' capacity to discriminate between various classes. The last metric is time execution of algorithms. This metric considers computational efficiency to provide insights into the algorithm's scalability and practical applicability.

Experiment design and result analysis

Data description

This research uses six datasets that are published in the Kaggle platform, which are telecommunication 1 (telco 1) dataset [22], bank dataset [23], mobile dataset [24], telecommunication 2 (telco 2) dataset [25], telecommunication 3 (telco 3) [26] and insurance

Table 1 Datasets overview

Dataset	Features	Data number	Imbalance ratio
Bank	13 (5 cat, 5 cont)	10,000	3.9
Mobile	65 (4 cat, 58 cont)	66,469	3.7
Telco 1	19 (16 cat,3 cont)	7044	2.7
Telco 2	19 (3 cat, 15 cont)	4250	6.1
Telco 3	15	3150	5.3
Insurance	16	33,909	7.5

Table 2 Packages and parameters for the sampling method

Technique	Approach	Package	Parameter	Source
Over	SMOTE	SMOTE	Default	[28]
	ADAYSN	ADAYSN	Default	[29]
	WGAN-GP	WGANGP	Epoch = 100	[30]
	CTGAN	CTGAN	Epoch = 100	[31]
Under	ENN	EditedNearestNeighbours	Default	[32]
Hybrid	SMOTE-ENN	SMOTEENN	Default	[33]
Hybrid	ADAYSN-ENN	ADAYSN EditedNearestNeighbours	Default	[29, 32]
	WGAN-GP+ENN	WGANGP EditedNearestNeighbours	Epoch = 100	[30, 32]
	CTGAN-ENN	CTGAN EditedNearestNeighbours	Epoch = 100	[31, 32]

dataset [27]. The imbalance ratio shows all datasets have imbalanced problems from 2.7 until extremely imbalanced on 7.5 imbalance ratio. This condition is one of the factor classifiers that hardly achieve stratifying results. The datasets overview shown in Table 1 below.

Experiment settings

All experiments in this research were performed using available Python packages. Table 2 shows several types of techniques, packages, and parameters that we used in the experiment. We compared the results of CTGAN-ENN (CE) with those of the conventional hybrid sampling techniques SMOTE-ENN (SE), ADAYSN-ENN (AE), and another GAN-based hybrid sampling technique known as WGAN-GP+ENN (WE) to demonstrate the efficacy of our proposed research framework.

We apply the filling missing value preprocessing technique, wherein null or missing values in our experiment are replaced with mean values computed from the data. Since the mean replaces continuous data without introducing outliers, it yields a better result when used in place of missing or null values. The experiments of this study use fivefold cross validation for data training and data testing, k-fold cross validation gives a stable value of evaluation, because it is used all subsets of data.

The next stage after preprocessing customer churn data using the hybrid sampling method is to do a classification task. Table 3 displays the models, packages, and parameters that we employed for both classical and ensemble machine learning.

Table 3 Packages, functions, and parameters for machine learning method in data-level experiment

Algorithm	Package	Parameter	Source
KNN	KNeighborsClassifier	Default	[34]
DT	DecisionTreeClassifier	Default	[34]
NB	GaussianNB	Default	[34]
XGB	XGBClassifier	Default	[35]
RF	RandomForestClassifier	Default	[36]
GBM	GradientBoostingClassifier	Default	[34]
LGB	LGBMClassifier	Default	[37]

Table 4 Packages, functions, and parameters for machine learning method in algorithm-level experiment

Algorithm	Package	Parameter	Source
DT	DecisionTreeClassifier	Class_weight = {0:1;1:10}	[34]
XGB	XGBClassifier	SCALE_pos_weight = 1	[35]
RF	RandomForestClassifier	Class_weight = {0:1;1:10}	[36]
LGB	LGBMClassifier	Class_weight = {0:1;1:10}	[37]

In the experiment we provided cost-sensitive learning approached on Decision Tree, XGBoost, Random Forest and Light Gradient Boosting algorithms. The aim is to provide a comparison of CTGAN-ENN not only on data-level solutions but also on algorithm-level solutions. Table 4 shown the packages and parameter in the experiment that implemented using the scikit-learn library.

Evaluation metrics

Three separate evaluation techniques were used in the experiment, each with a distinct set of goals. The precision and recall harmonic means are used to calculate the first metric, known as the F1 score. It provides an equitable assessment of a model's efficacy by considering both false positives and false negatives.

$$F1 - Score = 2 * \frac{(PR * RC)}{(PR + RC)} \tag{1}$$

RC stand for recall, the ratio of accurately predicted positive observations to the total number of real positives. It assesses the model's capacity to recognize and accurately classify every occurrence of the positive class. *PR* is precision, the ratio of accurately predicted positive observations to the total number of predicted positives is known as precision. It gauges how well the model predicts the favorable outcomes.

The second metric is AUC-ROC, plotting the true positive rate against the false positive rate yields the second figure of Area Under the Receiver Operating Characteristic Curve. The area under the ROC curve is denoted by AUC. It offers a model's overall performance across different thresholds.

The third is G-mean, which uses a geometric mean of recall and specificity to provide a balanced assessment of a model's performance [2]. The G-mean measure can identify positive examples and prevent false positives, regardless of the distribution between sample classes.

$$G - mean = \sqrt{RC * SP} \tag{2}$$

Specificity, represented by *SP*, is a gauge of how well the model can identify negative cases. Rather than dividing the training and testing data at random, we employed the k-fold cross-validation approach with the number of k = 5.

F1-score result

Tables 5, 6, 7, 8 and 9 give the result on each dataset with different evaluation metric. The best value of the experiment is marked in bold. Table 5 shows that CTGAN-ENN (CE) achieved the best performance in 21 scenarios out of 42 scenarios in F1-Score.

The performance of WE and SE both achieved rank 1 in 6 scenarios of F1-Score. WE performed well in the Decision Tree and Random Forest algorithm in two datasets, while SE performed well in all machine learning algorithms but only in the telco1 dataset. Our proposed research framework outperformed all machine learning algorithms, especially in KNN, GBM, XGB and LGB.

AUC-ROC result

Table 6 shows that CE obtained 29 of the best results out of 42 scenarios in terms of AUC. It outperformed the other sampling method most in all datasets except with the KNN and NB algorithms. Table 5 confirms that our proposed research framework performs well by reaching 29 the best performance out of 42 scenarios in the G-mean metric. Interestingly, WE performed well in the DT algorithm consistently in all evaluation metrics.

We found an interesting result that CE worked well on ensemble machine learning in all scenarios. Our technique performed exceptionally well in GBM, XGB, RF and LGB. Table 7 shows the mean ranking of all scenarios in this experiment and proves that CE achieved the best mean ranking in the ensemble machine learning algorithm.

G-Mean result

Machine learning algorithms are divided into two types: KNN, DT, NB is the classical model; otherwise, GBM, XGB, RF, LGB is ensemble model. From the experimental result, we can see the effectiveness of our proposed research framework. The ensemble model given a few improvements from the original CTGAN compared to the combination model with ENN (CE), but in the classical model, KNN, DT, and NB, the improvement was more significant than the ensemble model. It happened because the ensemble model has a robust classification ability. Even without the sampling method, the ensemble model achieved satisfying results.

Accuracy of minority class

Minority class accuracy refers to the accuracy metric specifically calculated for the minority class; in this study case all datasets have the same minority class (churn). The accuracy of minority class in this experiment was measured by recall metric because the minority class is positive class, as shown in formula 3 below.

Table 5 Experimental result on F1-Score

Alg	Dataset	NONE	ADA	AE	SM	SE	WG	WE	CT	CE
KNN	Bank	0.133	0.697	0.698	0.712	0.907	0.342	0.520	0.679	**0.913**
	Mobile	0.699	0.845	0.962	0.864	0.967	0.873	0.891	0.854	**0.992**
	Telco1	0.495	0.771	0.792	0.782	**0.962**	0.607	0.760	0.768	0.907
	Telco2	0.439	0.851	0.850	0.855	0.953	0.890	0.908	0.838	**0.960**
	Telco3	0.497	0.869	**0.965**	0.876	0.962	0.863	0.944	0.891	0.962
	Insurance	0.425	0.917	**0.985**	0.922	0.977	0.911	0.941	0.844	0.952
DT	Bank	0.481	0.798	0.796	0.798	0.838	0.887	0.894	0.775	**0.906**
	Mobile	0.620	0.859	0.927	0.868	0.931	0.926	0.933	0.894	**0.992**
	Telco1	0.484	0.784	0.775	0.793	**0.938**	0.855	0.879	0.791	0.928
	Telco2	0.720	0.868	0.860	0.882	0.910	0.982	**0.987**	0.874	0.934
	Telco3	0.790	0.950	0.979	0.951	0.971	0.892	**0.992**	0.965	0.982
	Insurance	0.468	0.898	0.949	0.901	0.947	0.969	**0.981**	0.924	0.970
NB	Bank	0.119	0.734	0.450	0.735	0.764	0.276	0.361	0.697	**0.870**
	Mobile	0.446	0.710	0.804	0.752	0.793	**0.830**	0.760	0.722	0.731
	Telco1	0.613	0.781	0.727	0.792	**0.904**	0.593	0.670	0.808	0.789
	Telco2	0.543	0.739	0.738	0.750	0.807	0.833	0.835	0.874	**0.934**
	Telco3	0.466	0.803	0.883	0.799	0.863	0.787	0.845	**0.882**	0.826
	Insurance	0.400	0.755	0.839	0.763	0.839	0.864	**0.876**	0.737	0.719
GBM	Bank	0.575	0.837	0.840	0.841	0.861	0.587	0.628	0.828	**0.934**
	Mobile	0.699	0.855	0.925	0.879	0.934	0.869	0.868	0.921	**0.992**
	Telco1	0.575	0.834	0.800	0.836	**0.949**	0.583	0.707	0.851	0.933
	Telco2	0.834	0.839	0.835	0.860	0.903	0.957	0.961	0.959	**0.983**
	Telco3	0.816	0.941	0.970	0.944	0.970	0.903	0.968	0.971	**0.985**
	Insurance	0.490	0.899	0.946	0.905	0.947	0.900	0.923	0.942	**0.978**
XGB	Bank	0.583	0.859	0.863	0.855	0.899	0.742	0.773	0.833	**0.937**
	Mobile	0.695	0.898	0.957	0.904	0.960	0.876	0.887	0.921	**0.991**
	Telco1	0.566	0.844	0.830	0.846	**0.964**	0.705	0.810	0.846	0.949
	Telco2	0.841	0.934	0.935	0.936	0.957	0.992	**0.993**	0.973	0.986
	Telco3	0.869	0.976	0.988	0.976	0.989	0.929	**0.997**	0.983	0.992
	Insurance	0.536	0.945	0.979	0.945	0.978	0.930	0.953	0.946	**0.981**
RF	Bank	0.564	0.858	0.861	0.859	0.882	0.921	0.928	0.828	**0.933**
	Mobile	0.676	0.904	0.959	0.902	0.958	0.858	0.960	0.918	**0.994**
	Telco1	0.554	0.845	0.836	0.849	**0.960**	0.887	0.914	0.846	0.942
	Telco2	0.830	0.938	0.938	0.939	0.958	**0.992**	**0.992**	0.958	0.981
	Telco3	0.839	0.969	0.985	0.972	0.989	0.934	**0.996**	0.979	0.992
	Insurance	0.505	0.939	0.972	0.941	0.972	0.978	**0.987**	0.944	0.980
LGB	Bank	0.593	0.851	0.774	0.855	**0.892**	0.821	0.878	0.842	0.877
	Mobile	0.698	0.888	0.949	0.897	0.952	0.704	0.892	0.921	**0.992**
	Telco1	0.572	0.847	0.830	0.847	**0.962**	0.652	0.810	0.850	0.946
	Telco2	0.848	0.918	0.918	0.924	0.956	0.986	0.987	0.975	**0.994**
	Telco3	0.874	0.975	0.990	0.977	0.991	0.931	**0.998**	0.982	0.992
	Insurance	0.551	0.938	0.973	0.939	0.973	0.532	0.665	0.946	**0.982**

Bold values represent CTGAN-ENN achieved the outperform on F1-score metric compared by another sampling method

Table 6 Experimental result on AUC

Alg	Dataset	NONE	ADA	AE	SM	SE	WG	WE	CT	CE
KNN	Bank	0.535	0.715	0.716	0.745	**0.961**	0.774	0.856	0.760	0.928
	Mobile	0.855	0.882	0.985	0.912	0.994	0.951	0.972	0.887	**0.999**
	Telco1	0.748	0.807	0.902	0.843	**0.989**	0.839	0.914	0.853	0.980
	Telco2	0.685	0.933	0.941	0.937	0.990	0.955	0.968	0.899	**0.994**
	Telco3	0.829	0.932	0.991	0.940	0.990	0.937	0.985	0.950	**0.992**
	Insurance	0.569	0.960	**0.998**	0.913	0.977	0.980	0.989	0.852	0.946
DT	Bank	0.683	0.798	0.794	0.796	0.809	0.887	**0.936**	0.774	0.825
	Mobile	0.776	0.853	0.906	0.847	0.931	0.926	0.955	0.909	**0.993**
	Telco1	0.655	0.782	0.822	0.794	0.929	0.909	0.920	0.794	**0.944**
	Telco2	0.844	0.864	0.862	0.879	0.874	0.987	**0.989**	0.892	0.940
	Telco3	0.897	0.955	0.973	0.956	0.971	0.918	**0.992**	0.965	0.982
	Insurance	0.703	0.898	0.941	0.901	0.941	0.977	**0.985**	0.925	0.976
NB	Bank	0.743	0.790	0.792	0.788	0.800	0.789	0.798	0.777	**0.865**
	Mobile	0.845	0.788	0.854	0.843	0.905	**0.927**	0.922	0.840	0.778
	Telco1	0.824	0.845	0.869	0.739	**0.959**	0.828	0.860	0.866	0.910
	Telco2	0.844	0.790	0.789	0.820	0.824	0.890	0.891	0.939	**0.981**
	Telco3	0.901	0.878	0.936	0.876	0.934	0.924	**0.965**	0.932	0.930
	Insurance	0.814	0.814	0.891	0.834	0.902	0.932	**0.939**	0.804	0.867
GBM	Bank	0.864	0.916	0.918	0.918	0.924	0.903	0.911	0.920	**0.960**
	Mobile	0.910	0.923	0.967	0.938	0.982	0.960	0.962	0.888	**1.000**
	Telco1	0.847	0.915	0.932	0.917	0.986	0.870	0.908	0.935	**0.991**
	Telco2	0.918	0.927	0.926	0.940	0.950	0.981	0.983	0.984	**0.999**
	Telco3	0.978	0.983	0.993	0.983	0.994	0.973	0.997	0.995	**0.999**
	Insurance	0.923	0.965	0.984	0.969	0.986	0.982	0.987	**0.988**	**0.988**
XGB	Bank	0.847	0.932	0.934	0.931	0.955	0.954	0.959	0.922	**0.964**
	Mobile	0.905	0.957	0.987	0.955	0.992	0.970	0.977	0.975	**1.000**
	Telco1	0.825	0.925	0.950	0.927	0.992	0.920	0.950	0.934	**0.995**
	Telco2	0.911	0.982	0.982	0.983	0.988	0.995	0.996	0.986	**0.999**
	Telco3	0.987	0.997	0.999	0.996	**1.000**	0.979	**1.000**	0.983	**1.000**
	Insurance	0.928	0.990	0.997	0.990	0.997	0.990	0.995	0.991	**0.999**
RF	Bank	0.852	0.930	0.932	0.931	0.942	0.984	**0.986**	0.916	0.956
	Mobile	0.899	0.955	0.988	0.954	0.993	0.979	0.994	0.973	**0.999**
	Telco1	0.826	0.922	0.950	0.926	0.990	0.962	0.976	0.927	**0.992**
	Telco2	0.913	0.984	0.985	0.985	0.989	0.997	0.997	0.984	**0.999**
	Telco3	0.985	0.993	0.999	0.993	0.999	0.981	**1.000**	0.997	0.999
	Insurance	0.926	0.987	0.996	0.987	0.996	0.997	**0.999**	0.990	**0.999**
LGB	Bank	0.861	0.928	0.932	0.929	0.949	0.931	0.963	0.927	**0.973**
	Mobile	0.909	0.950	0.982	0.951	0.990	0.945	0.972	0.976	**1.000**
	Telco1	0.838	0.926	0.949	0.927	0.991	0.878	0.936	0.936	**0.993**
	Telco2	0.909	0.976	0.977	0.977	0.987	0.995	0.995	0.992	**1.000**
	Telco3	0.989	0.997	0.999	0.996	0.999	0.981	**1.000**	0.998	**1.000**
	Insurance	0.934	0.988	0.996	0.988	0.996	0.934	0.954	0.991	**0.999**

Bold values represent CTGAN-ENN achieved the outperform on AUC metric compared by another sampling method

Table 7 Experimental result on G-Mean

Alg	Dataset	None	ADA	AE	SM	SE	WG	WE	CT	CE
KNN	Bank	0.289	0.659	0.783	0.680	0.833	0.495	0.625	0.689	**0.856**
	Mobile	0.737	0.780	0.833	0.783	0.866	0.887	0.830	0.813	**0.970**
	Telco1	0.622	0.742	0.842	0.764	**0.954**	0.713	0.812	0.777	0.922
	Telco2	0.553	0.807	0.809	0.820	0.921	0.906	0.916	0.847	**0.964**
	Telco3	0.637	0.841	0.942	0.852	0.948	0.893	0.951	0.879	**0.958**
	Insurance	0.569	0.909	**0.981**	0.913	0.977	0.927	0.948	0.852	0.952
DT	Bank	0.654	0.800	0.800	0.792	0.809	0.944	**0.948**	0.776	0.820
	Mobile	0.720	0.814	0.843	0.820	0.855	0.941	0.887	0.849	**0.962**
	Telco1	0.635	0.786	0.817	0.790	0.928	0.920	0.938	0.794	**0.944**
	Telco2	0.833	0.861	0.862	0.878	0.863	**0.992**	**0.992**	0.888	0.942
	Telco3	0.875	0.949	0.972	0.949	0.970	0.906	**0.996**	0.958	0.982
	Insurance	0.669	0.899	0.942	0.899	0.942	0.983	**0.991**	0.929	0.976
NB	Bank	0.263	0.720	0.561	0.720	0.705	0.439	0.515	0.707	**0.749**
	Mobile	0.656	0.564	0.652	0.641	0.732	**0.871**	0.802	0.633	0.682
	Telco1	0.749	0.771	0.793	0.738	**0.898**	0.749	0.778	0.797	0.843
	Telco2	0.710	0.702	0.705	0.714	0.681	0.855	0.854	0.877	**0.981**
	Telco3	0.760	0.750	0.829	0.745	0.820	0.834	**0.871**	0.799	0.755
	Insurance	0.648	0.676	0.757	0.682	0.762	0.878	**0.887**	0.670	0.761
GBM	Bank	0.665	0.838	0.799	0.841	0.843	0.675	0.707	0.833	**0.845**
	Mobile	0.765	0.818	0.852	0.843	0.891	0.882	0.817	0.888	**0.970**
	Telco1	0.681	0.830	0.843	0.831	0.942	0.685	0.778	0.852	**0.944**
	Telco2	0.857	0.846	0.841	0.866	0.871	0.962	0.966	0.962	**0.984**
	Telco3	0.871	0.933	0.960	0.936	0.963	0.924	0.971	0.961	**0.984**
	Insurance	0.617	0.897	0.938	0.902	0.941	0.914	0.932	0.942	**0.981**
XGB	Bank	0.685	0.858	0.850	0.856	**0.882**	0.796	0.822	0.837	0.864
	Mobile	0.755	0.859	0.869	0.861	0.913	0.884	0.830	0.880	**0.966**
	Telco1	0.681	0.844	0.865	0.844	0.957	0.782	0.860	0.847	**0.960**
	Telco2	0.872	0.931	0.933	0.938	0.942	0.994	**0.995**	0.973	0.988
	Telco3	0.912	0.973	0.985	0.971	0.985	0.941	**0.998**	0.976	0.991
	Insurance	0.673	0.947	0.977	0.947	0.997	0.942	0.961	0.947	**0.983**
RF	Bank	0.664	0.858	0.853	0.861	0.866	0.940	**0.946**	0.833	0.849
	Mobile	0.745	0.876	0.910	0.870	0.930	0.928	0.891	0.888	**0.968**
	Telco1	0.665	0.846	0.870	0.849	0.954	0.917	0.936	0.849	**0.955**
	Telco2	0.857	0.941	0.945	0.940	0.949	**0.993**	**0.993**	0.955	0.984
	Telco3	0.912	0.966	0.979	0.969	0.983	0.951	**0.997**	0.970	0.991
	Insurance	0.631	0.938	0.967	0.939	0.968	0.985	**0.991**	0.944	0.982
LGB	Bank	0.688	0.852	0.826	0.853	0.872	0.859	0.892	0.845	**0.902**
	Mobile	0.761	0.852	0.879	0.857	0.894	0.806	0.867	0.884	**0.962**
	Telco1	0.683	0.842	0.865	0.844	0.956	0.683	0.831	0.855	**0.957**
	Telco2	0.878	0.922	0.919	0.929	0.940	0.988	0.988	0.976	**0.994**
	Telco3	0.918	0.974	0.986	0.972	0.988	0.948	**0.999**	0.975	0.990
	Insurance	0.680	0.939	0.969	0.942	0.970	0.647	0.756	0.946	**0.984**

Bold values represent CTGAN-ENN achieved the outperform on G-Mean metric compared by another sampling method

Table 8 Result of Accuracy in Minority Class (%)

Alg	Dataset	None	ADA	AE	SM	SE	WG	WE	CT	CE
KNN	Bank	8.9	77.2	73.8	76.5	**93.8**	24.3	45.9	67.1	93.7
	Mobile	64.9	92.6	98.0	90.9	97.8	78.8	90.4	77.2	**98.9**
	Telco1	44.1	84.9	88.2	82.8	**96.6**	76.4	84.5	72.8	90.0
	Telco2	31.1	**99.8**	99.7	97.8	99.7	83.9	84.8	67.8	88.5
	Telco3	43.7	95.9	99.4	94.2	**99.2**	85.1	91.0	83.3	94.5
	Insurance	33.3	99.0	**99.6**	98.4	99.3	89.2	92.3	78.3	91.9
DT	Bank	50.2	80.8	72.8	81.0	84.8	74.3	82.0	82.1	**90.0**
	Mobile	66.4	89.3	93.8	89.9	93.9	91.4	89.3	91.2	**99.3**
	Telco1	50.0	78.9	77.6	80.2	93.7	89.9	96.5	78.8	**91.1**
	Telco2	73.6	88.8	88.4	89.6	92.4	98.6	98.3	92.3	**94.6**
	Telco3	80.3	95.8	98.0	95.8	97.4	85.4	**99.4**	95.9	97.8
	Insurance	48.9	90.4	95.5	90.7	95.2	98.0	**99.3**	92.4	97.2
NB	Bank	7.2	78.0	36.8	77.6	80.4	7.3	8.8	71.5	**92.1**
	Mobile	92.7	91.7	91.1	94.2	84.9	78.1	97.3	98.0	**98.5**
	Telco1	73.9	81.0	80.7	82.1	**89.6**	77.0	82.0	85.8	88.4
	Telco2	54.3	81.1	81.1	81.8	86.8	72.5	72.7	85.8	**91.3**
	Telco3	93.5	94.5	95.1	94.1	94.6	92.2	93.4	98.0	**98.0**
	Insurance	47.7	91.4	95.0	92.1	**95.4**	89.2	83.7	83.7	85.8
GBM	Bank	45.8	83.8	68.8	83.1	**87.7**	35.8	48.9	83.7	69.4
	Mobile	68.6	86.9	91.4	88.8	91.7	85.9	80.0	91.8	**98.4**
	Telco1	51.2	86.5	80.9	86.4	**95.5**	74.4	83.6	84.3	92.1
	Telco2	74.2	81.0	80.5	81.6	87.9	92.8	93.8	93.9	**96.2**
	Telco3	78.0	96.5	99.3	96.3	**98.8**	91.7	95.2	96.6	97.7
	Insurance	39.2	92.2	95.8	92.5	95.3	94.3	96.8	91.8	**97.1**
XGB	Bank	48.9	86.0	76.3	84.1	88.4	51.3	65.7	83.7	**93.8**
	Mobile	68.0	91.6	91.4	91.6	95.2	85.3	90.5	91.5	**98.5**
	Telco1	51.6	85.6	83.0	84.7	96.7	80.7	89.4	82.6	93.9
	Telco2	77.4	94.1	93.5	93.0	96.5	**98.6**	**98.6**	96.0	98.0
	Telco3	85.1	98.7	99.5	98.5	99.5	94.3	**99.6**	97.7	98.9
	Insurance	47.1	94.2	98.0	94.5	97.8	98.6	**99.4**	93.2	97.6
RF	Bank	45.6	86.1	74.7	86.3	90.0	69.9	80.4	83.7	**93.6**
	Mobile	63.7	92.5	96.2	90.6	94.6	94.7	97.4	90.5	**99.2**
	Telco1	48.9	85.2	83.6	85.7	**96.5**	92.8	96.4	83.3	93.1
	Telco2	74.5	92.7	92.7	91.2	95.9	98.7	**98.8**	96.3	97.8
	Telco3	79.9	98.0	99.7	94.3	99.7	97.3	**99.5**	97.7	99.0
	Insurance	41.1	96.4	98.2	96.3	98.0	98.4	**99.5**	92.2	97.3
LGB	Bank	49.5	84.8	74.3	85.2	89.9	42.3	57.1	83.9	**93.7**
	Mobile	68.2	91.0	94.6	90.8	94.2	79.7	89.4	91.5	**98.7**
	Telco1	52.1	85.6	82.2	85.3	**96.4**	76.9	86.0	83.1	94.1
	Telco2	77.7	90.8	90.7	90.4	94.9	97.4	97.9	96.2	**98.2**
	Telco3	86.1	98.7	99.5	98.6	99.6	95.5	**99.7**	97.8	98.9
	Insurance	47.9	94.3	97.5	94.4	97.3	**98.8**	99.5	93.0	97.5

Bold values represent CTGAN-ENN achieved the outperform on minority class prediction compared by another sampling method

Table 9 Cost-Sensitive Learning Result

Dataset	Metric	CS-DT	CE-DT	CS-RF	CE-RF	CS-LGB	CE-LGB	CS-XGB	CE-XGB
Bank	AUC	0.676	**0.825**	0.849	**0.956**	0.856	**0.973**	0.842	**0.964**
	F1-Score	0.680	**0.906**	0.733	**0.933**	0.688	**0.877**	0.744	**0.937**
	G-Mean	0.648	**0.820**	0.642	**0.849**	0.765	**0.902**	0.681	**0.864**
Mobile	AUC	0.778	**0.993**	0.900	**0.999**	0.908	**1.000**	0.904	**1.000**
	F1-Score	0.768	**0.992**	0.803	**0.994**	0.742	**0.992**	0.808	**0.991**
	G-Mean	0.784	**0.962**	0.794	**0.968**	0.824	**0.962**	0.794	**0.970**
Telco 1	AUC	0.651	**0.944**	0.824	**0.992**	0.856	**0.993**	0.821	**0.995**
	F1-Score	0.659	**0.928**	0.699	**0.945**	0.688	**0.946**	0.705	**0.949**
	G-Mean	0.631	**0.944**	0.642	**0.955**	0.765	**0.957**	0.672	**0.960**
Telco 2	AUC	0.831	**0.940**	0.917	**0.999**	0.920	**1.000**	0.915	**0.999**
	F1-Score	0.832	**0.934**	0.883	**0.981**	0.908	**0.994**	0.905	**0.986**
	G-Mean	0.823	**0.942**	0.825	**0.984**	0.897	**0.994**	0.869	**0.988**
Telco 3	AUC	0.890	**0.982**	0.974	**0.999**	0.986	**1.000**	0.984	**1.000**
	F1-Score	0.877	**0.982**	0.906	**0.992**	0.929	**0.992**	0.921	**0.992**
	G-Mean	0.879	**0.982**	0.892	**0.991**	0.944	**0.990**	0.943	**0.991**
Insurance	AUC	0.680	**0.976**	0.974	**0.999**	0.986	**0.999**	0.984	**0.999**
	F1-Score	0.686	**0.970**	0.906	**0.980**	0.929	**0.982**	0.921	**0.981**
	G-Mean	0.631	**0.976**	0.892	**0.982**	0.944	**0.984**	0.943	**0.983**

Bold values represent comparison between cost-sensitive learning (CS) and CTGAN-ENN (CE) in all across method, the result shows CE surprass CS in all method

Table 10 Class overlaps from features in datasets

Dataset	CTGAN	CTGAN-ENN
*Bank	6	**4**
*Mobile	50	**12**
*Telco1	13	**6**
*Telco2	14	**5**
*Telco 3	10	**5**
*Insurance	11	**3**

Bold values represent CTGAN-ENN has a smaller number of overlap features than CTGAN, the smaller overlap features impact on better performance of prediction method

*Measurement details: https://github.com/mahayasa/gan-hybrid-sampling-customer-churn/tree/main/measurement

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{3}$$

Table 8 shows the accuracy of minority class result by percentage, The comparison sheds light on the efficacy of CTGAN-ENN in addressing the challenges of imbalanced datasets. CTGAN-ENN outperformed on 19 of 42 scenario, and in the second-best rank is SE achieved best result on 10 of 42 scenario.

CTGAN-ENN and algorithm-level comparison

CTGAN-ENN achieved better results in all algorithm-level experiments, as shown in Table 9 we used Cost-Sensitive (CS) on Decision Tree (DT), Random Forest (RF), Light Gradient-Boosting Machine (LGB) and XGBoost (XGB). The cost that we used was 1 for class 0 or not churn, and 10 for class 1 or churn class, we given more cost

Table 11 Mean ranking score

Alg	Metric	NONE	ADA	AE	SM	SE	WG	WE	CT	CE
KNN	AUC	9	7.3	3.6	6.2	**2.2**	5	3.1	6.3	**2.2**
	F1-Score	9	6.2	3.2	4.8	1.8	6.6	4.8	6.6	**1.6**
	G-Mean	9	7.3	3.6	6.1	2.1	5.3	4.2	5.6	**1.5**
DT	AUC	9	7	5.5	6.4	3.8	3.6	**1.5**	5.6	2.2
	F1-Score	9	7	5.6	5.8	3.5	4.1	**1.6**	6.2	1.8
	G-Mean	9	7	5	6.5	4.2	3.2	**1.5**	5.6	2.1
NB	AUC	6.8	7	4.3	7.3	**3**	4.3	2.5	5.6	3.8
	F1-Score	8.8	6	4.3	4.8	**2.8**	5.3	4.3	4.2	4.2
	G-Mean	**7**	6.6	5.2	6.3	4.2	3.8	3.3	5	**3.1**
GBM	AUC	8.6	6.6	4.8	5.6	3.1	6.6	4.3	3.5	**1**
	F1-Score	9	6.6	4.5	5	2.5	6.8	5.5	3.6	**1.1**
	G-Mean	8.6	6.6	5.5	5.5	2.8	6.2	5.2	3.3	**1**
XGB	AUC	8.6	6.3	4	6.5	**2.3**	6	2.6	6	**1**
	F1-Score	9	5.8	4.2	5.1	2.7	7	4.5	4.5	**1.6**
	G-Mean	9	5.8	4.6	5.4	2.2	5.8	4.8	4.8	**1.8**
RF	AUC	8.8	7	4.1	6.5	3.1	4.1	1.6	6.3	**1.5**
	F1-Score	9	7	5.2	6.2	3.3	4.5	**1.6**	5.6	1.8
	G-Mean	9	7	5	6.5	3.1	3.3	**2**	5.8	2.8
LGB	AUC	8.6	6.3	5	5.6	2.8	6.5	4	4.5	**1.2**
	F1-Score	8.8	6	5	5	2.3	7.2	4.2	4.3	**1.6**
	G-Mean	8.6	6.3	5	5.6	2.8	6.5	4	4.5	**1.2**

Bold value indicates the best mean ranking score of all compared sampling methods

on minority class to adjust the robustness of algorithm. Based on our experiment CTGAN-ENN (CE) consistently gained better performance in AUC, F1-Score and G-Mean metric compared to cost-sensitive learning result.

Mean ranking score, class overlap degree, and time execution result

CTGAN-ENN achieved a lower Fisher's discriminant ratio is displayed in Table 10. The datasets degree of class overlap is considerably reduced by CTGAN-ENN.

The average rank within all experiment scenarios provided in Table 11, lower value of the mean ranking score indicates better results in the scenario. CE achieved the best average rank score in almost all scenarios, 14 of 21 scenarios. WE gained five best average ranks; WE outperformed in the DT classifier while CE outperformed in almost all ensemble machine learning models.

We calculate the algorithm time consumption after data preprocessing using CTGAN and CTGAN-ENN. CTGAN-ENN reduced the time consumption algorithm in all machine learning algorithms except in NB. Although the improvement is not much in several cases, CTGAN-ENN has less algorithm time execution compared with CTGAN. The time consumption of the algorithm is significantly reduced by 38.47% on average, indicating that CTGAN-ENN can work effectively in large data of customer churn. The detailed results of these comparisons are provided in Appendix 1.

Figure 5 represents the visualization of all datasets by using the T-SNE technique. We can spot the difference between the middle side of the figure and the right side

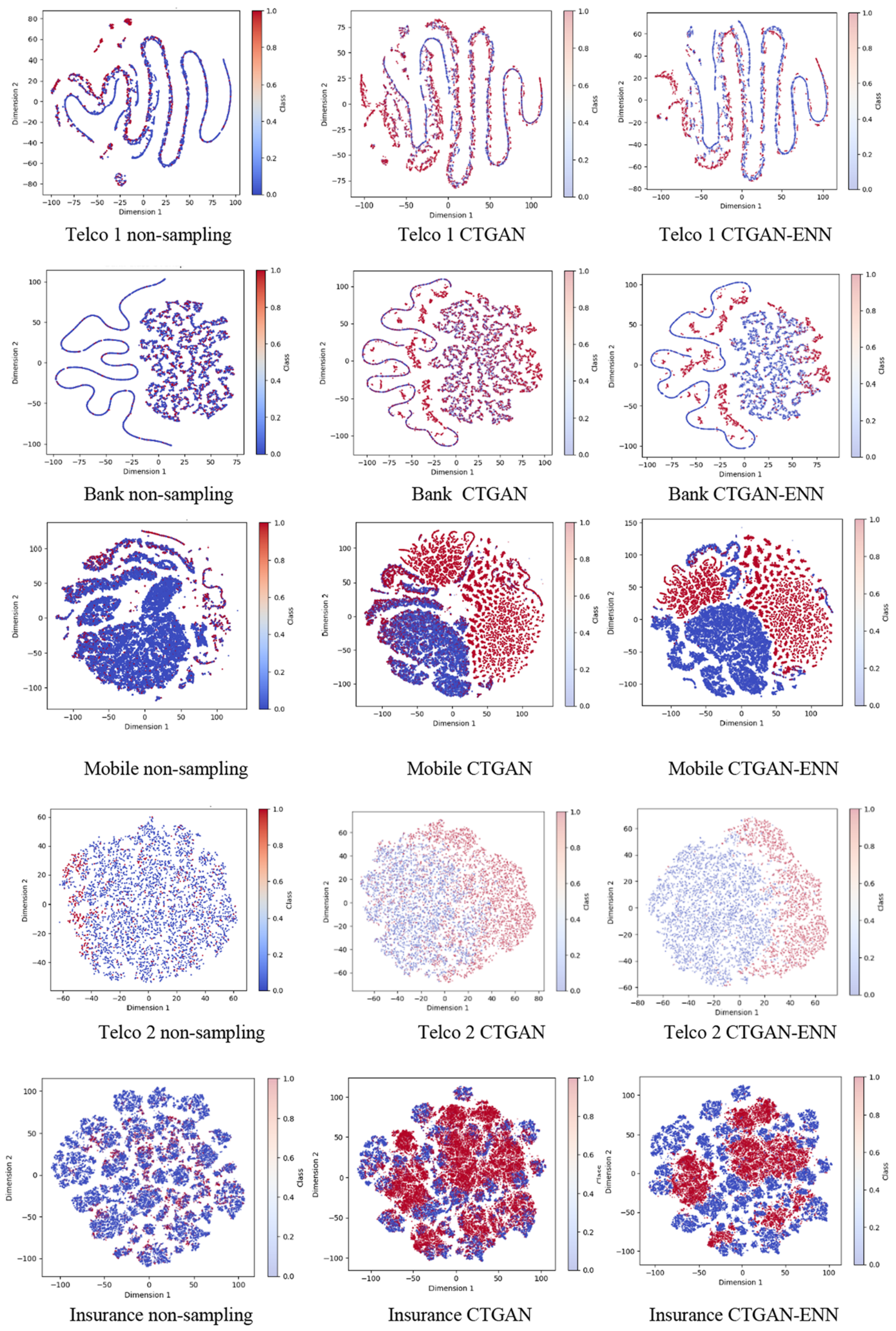

Fig. 5 The visualization of non-sampling (left), CTGAN (middle) and CTGAN-ENN (right) of all datasets

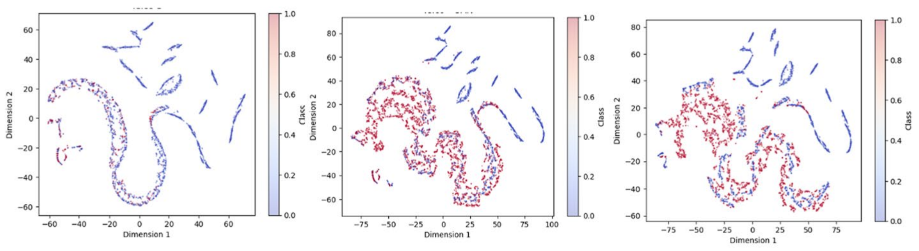
Telco 3 non-sampling Telco 3 CTGAN Telco 3 CTGAN-ENN

Fig. 5 continued

of the figure. Almost all stacked data points between classes were removed by ENN and produced a clear area between classes. The results of CTGAN-ENN in all datasets made machine learning algorithms learn more easily and achieved outperformed results compared to other sampling methods.

Finally, after all the experiment's measurements, our proposed research framework, CTGAN-ENN, outperforms compared to other over-sampling and hybrid sampling methods. From the experimental results, CTGAN-ENN is working very well on KNN and ensemble machine learning models GBM, XGB, and RF. If a company considers building a fast and optimal customer churn prediction model, KNN is the optimal model. Using our proposed research framework, a company can build the customer churn model using CTGAN-ENN and choose the machine learning algorithm according to their objective.

Compared with the latest work [2, 9], our work offers a new hybrid sampling method perspective using tabular GAN. The strong point of tabular GAN is the generative phase is built for tabular data while another GAN is originally built for image data.

Summary and analysis of results

The classical generative data algorithms like SMOTE and ADAYSN only work well in one dataset according to our experime nts. This happened because SMOTE and ADAYSN generate data using a cluster of original datasets. The variation of data might not be diverse for training in machine learning. In some datasets, SMOTE and ADAYSN work because the original data itself has a good distribution.

CTGAN-ENN in our work still has some limitations. The customer churn datasets usually have only two classes (churn and not churn). Meanwhile, that might be a multiclass problem for customer classifications in real-world datasets. The variation of GANs is widely developed nowadays. Besides the classification task, the GAN's technique can be used in other customer machine learning tasks. For example, predicting the demand for a product by days or predicting a customer transaction number by period. Our work does not yet cover all the possibilities for using GANs and their hybrid method.

Our proposed research framework with XGBoost algorithm achieved better results than the latest work on the Telco 1 and Insurance datasets [1]. Specifically, it achieved an F1-Score of 0.949 for Telco 1 and 0.981 for Insurance, while the latest work achieved 0.635 and 0.623 respectively. Algorithm-level approach by cost-sensitive learning used

DT, RF, LGB and XGB algorithms was compared with CTGAN-ENN, CTGAN-ENN consistently achieved better performance in all of experiments.

Compared with other latest work results [2], because the datasets are different, we compared with the same characteristics of datasets by imbalanced ratio for the latest work was IBM dataset with 2.76 imbalanced ratio and our works was Telco 1 with 2.7 imbalanced ratio. Our works showed better performance both in AUC and G-mean metrics. The latest work in AUC with GBM achieved 0.832 while our work achieved 0.991. In G-mean metric, the latest work achieved 0.743 with Random Forest algorithm while our work achieved 0.955.

Conclusion

Class imbalance and class overlap are common problems in customer churn prediction. A GAN-based hybrid sampling method was recently proposed to handle the issues. For a better result, we proposed a tabular GAN-based hybrid sampling method, CTGAN-ENN, which combines a tabular GAN-based oversampling and Edited Nearest Neighbor (ENN) under-sampling.

Our primary takeaway from the experiment's outcome is that CTGAN-ENN enhanced customer churn prediction performance. The KNN algorithm achieved the best results in algorithm time consumption. Using CTGAN-ENN as a preprocessing strategy, GBM and XGB can be the most accurate models for predicting customer turnover if time consumption is disregarded. In the other scenario, we discovered that the DT algorithm performed well with WGAN-GP+ENN. Compared to recent studies using identical datasets and comparable dataset properties, our suggested research framework produced better findings.

Our findings have practical implications for stakeholders who want to build customer churn predictions in their company data. In the real-world case customer data rapidly increases over time, and the results of this study can give insight into the big data fields. By choosing the right combination of CTGAN-ENN and machine learning algorithms, stakeholders can consider their resources to build a customer churn prediction model.

This study's limitation is that it only focuses on data-level solutions and binary classification problems; instead of binary classification tasks, we can investigate several modifications of our methodology for multi-class classification tasks in another customer classification problem. Our study results also have a problem with classical machine learning outcomes. This issue might be handled with algorithm-level solutions, our experiment in algorithm-level solutions only used simple cost-sensitive learning without further detail analysis. In future work, we can consider extending our framework using cost-sensitive learning in details analysis as algorithm-level solutions.

Appendix 1

CTGAN and CTGAN-ENN algorithm time execution (s)

Algorithm	Dataset	CTGAN	CTGAN-ENN
KNN	Bank	2.10	**1.54**
	Mobile	336.54	**276.99**
	Telco1	8.58	**4.89**
	Telco2	1.72	**1.14**
	Telco3	3.22	**1.84**
	Insurance	81.91	**44.45**
DT	Bank	1.78	**0.92**
	Mobile	79.83	**39.53**
	Telco1	1.38	**1.33**
	Telco2	2.07	**1.49**
	Telco3	0.79	**0.67**
	Insurance	7.40	**6.20**
NB	Bank	0.30	0.30
	Mobile	4.69	**3.52**
	Telco1	0.66	**0.58**
	Telco2	0.18	**0.16**
	Telco3	0.23	**0.22**
	Insurance	1.40	**0.90**
GBM	Bank	43.70	**27.34**
	Mobile	1820	**1673**
	Telco1	29.40	**23.80**
	Telco2	51.26	**39.15**
	Telco3	27.42	**23.18**
	Insurance	247.61	**177.9**
XGB	Bank	27.56	**11.95**
	Mobile	458.07	**353.40**
	Telco1	27.20	**23.00**
	Telco2	27.25	**16.98**
	Telco3	2.72	**2.19**
	Insurance	158.94	**123.6**
RF	Bank	41.12	**28.73**
	Mobile	605.53	**336.22**
	Telco1	23.90	**14.20**
	Telco2	37.72	**23.53**
	Telco3	12.18	**10.72**
	Insurance	141.45	**100.62**
LGM	Bank	6.79	**3.78**
	Mobile	58.51	**53.91**
	Telco1	**5.56**	7.48
	Telco2	**4.72**	6.12
	Telco3	11.03	**7.67**
	Insurance	17.52	**15.19**

Bold values represent the algorithm execution time of customer churn predcition, altought theimprovement of prediction not really significant in some cases comapred to CTGAN, CTGAN-ENN has a smallertime excetuion. That indicates the realibility of CTGAN-ENN to works on big scale data

Abbreviations

ADA	ADAYSN
ADAYSN	Adaptive Synthetic Sampling
AE	ADAYSN+Edited Nearest Neighbor
AUC	Area Under Curve
CE	CTGAN + Edited Nearest Neighbor
CS	Cost-sensitive
CT	CTGAN
DT	Decision tree
ENN	Edited Nearest Neighbor
KNN	K-nearest neighbor
LGB	Light Gradient-Boosting Machine
NB	Naïve Bayes
RF	Random forest
GAN	Generative Adversarial Network
GBM	Gradient boosting machine
SE	SMOTE+Edited Nearest Neighbor
SM	SMOTE
SMOTE	Synthetic Minority Over-sampling Technique
WG	WGAN-GP
WE	WGAN-GP+Edited Nearest Neighbor
XGB	XGBoost

Acknowledgements
This work was supported by Khon Kaen University ASEAN GMS grant and part of AIDA (Applied Intelligence and Data Analytics) lab in College of Computing, Khon Kaen University, Thailand.

Author contributions
I Nyoman Mahayasa Adiputra and Paweena Wanchai conceived of the presented idea. Both authors contributed to the design and implementation of the research, to the analysis of the results and to the writing of the manuscript. I Nyoman Mahayasa Adiputra carried out the experiments. Paweena Wanchai supervised the project, provided critical feedback, and helped shape the research and analysis. Both authors discussed the results and contributed to the final version of the manuscript.

Funding
Khon Kaen University.

Availability of data and materials
Code and datasets can be accessed on GitHub: https://github.com/mahayasa/gan-hybrid-sampling-customer-churn. No datasets were generated or analysed during the current study.

Declarations

Competing interests
The authors declare that this research works have no competing interests.

Received: 21 November 2023 Accepted: 16 August 2024
Published online: 02 September 2024

References

1. Wen X, Wang Y, Ji X, Traoré MK. Three-stage churn management framework based on DCN with asymmetric loss. Expert Syst Appl. 2022;207:117998. https://doi.org/10.1016/j.eswa.2022.117998.
2. Zhu B, Pan X, Vanden Broucke S, Xiao J. A GAN-based hybrid sampling method for imbalanced customer classification. Inf Sci. 2022;609:1397–411. https://doi.org/10.1016/j.ins.2022.07.145.
3. Das S, Mullick SS, Zelinka I. On supervised class-imbalanced learning: an updated perspective and some key challenges. IEEE Trans Artif Intell. 2022;3(6):973–93. https://doi.org/10.1109/TAI.2022.3160658.
4. Goodfellow IJ et al. Generative Adversarial Networks. 2014. http://arxiv.org/abs/1406.2661
5. Huyen C. Designing machine learning systems. Sebastopol: O'Reilly Media; 2022.
6. Geiler L, Affeldt S, Nadif M. An effective strategy for churn prediction and customer profiling. Data Knowl Eng. 2022. https://doi.org/10.1016/j.datak.2022.102100.
7. Wu S, Yau W-C, Ong T-S, Chong S-C. Integrated churn prediction and customer segmentation framework for telco business. IEEE Access. 2021;9:62118–36. https://doi.org/10.1109/ACCESS.2021.3073776.
8. Su C, Wei L, Xie X. Churn prediction in telecommunications industry based on conditional Wasserstein GAN, In: *2022 IEEE 29th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, 2022, pp. 186–191. https://doi.org/10.1109/HiPC56025.2022.00034.
9. Ding H, Sun Y, Wang Z, Huang N, Shen Z, Cui X. RGAN-EL: a GAN and ensemble learning-based hybrid approach for imbalanced data classification. Inf Process Manag. 2023;60(2):103235. https://doi.org/10.1016/j.ipm.2022.103235.

10. Sáez JA, Luengo J, Stefanowski J, Herrera F. SMOTE-IPF: addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. Inf Sci. 2015;291:184–203. https://doi.org/10.1016/j.ins.2014.08.051.
11. Vuttipittayamongkol P, Elyan E. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. Inf Sci. 2020;509:47–70. https://doi.org/10.1016/j.ins.2019.08.062.
12. Xu Z, Shen D, Nie T, Kou Y. A hybrid sampling algorithm combining M-SMOTE and ENN based on random forest for medical imbalanced data. J Biomed Inform. 2020;107:103465. https://doi.org/10.1016/j.jbi.2020.103465.
13. Elkan C. The Foundations of Cost-Sensitive Learning.
14. Guo G, Wang H, Bell D, Bi Y, Greer K. LNCS 2888—KNN model-based approach in classification. Berlin: Springer; 2003.
15. Altuve M, Alvarez AJ, Severeyn E. Multiclass classification of metabolic conditions using fasting plasma levels of glucose and insulin. Health Technol (Berl). 2021;11(4):953–62. https://doi.org/10.1007/s12553-021-00550-w.
16. Kumari S, Kumar D, Mittal M. An ensemble approach for classification and prediction of diabetes mellitus using soft voting classifier. Int J Cogn Comput Eng. 2021;2:40–6. https://doi.org/10.1016/j.ijcce.2021.01.001.
17. Chen T, Guestrin C. XGBoost: a scalable tree boosting system. 2016. https://doi.org/10.1145/2939672.2939785.
18. Biau G, Fr GB. Analysis of a random forests model. 2012.
19. Shrivastav LK, Jha SK. A gradient boosting machine learning approach in modeling the impact of temperature and humidity on the transmission rate of COVID-19 in India. Appl Intell. 2021;51(5):2727–39. https://doi.org/10.1007/s10489-020-01997-6.
20. Ke G et al. LightGBM: A highly efficient gradient boosting decision tree. https://github.com/Microsoft/LightGBM. Accessed 17 Mar 2023.
21. Xu L, Skoularidou M, Cuesta-Infante A, Veeramachaneni K. Modeling Tabular data using Conditional GAN. 2019. http://arxiv.org/abs/1907.00503. Accessed 8 May 2023.
22. Telco Customer Churn | Kaggle. https://www.kaggle.com/datasets/blastchar/telco-customer-churn. Accessed 07 Jun 2023.
23. Churn Modelling | Kaggle. https://www.kaggle.com/datasets/shrutimechlearn/churn-modelling. Accessed 07 Jun 2023.
24. mobile-churn-data.xlsx | Kaggle. https://www.kaggle.com/datasets/dimitaryanev/mobilechurndataxlsx. Accessed 07 Jun 2023
25. Customer Churn Prediction 2020 | Kaggle. https://www.kaggle.com/competitions/customer-churn-prediction-2020. Accessed 07 Jun 2023.
26. Customer Churn. https://www.kaggle.com/datasets/royjafari/customer-churn. Accessed 18 Mar 2024
27. Vinod Kumar. Insurance churn prediction : weekend hackathon. https://www.kaggle.com/datasets/k123vinod/insurance-churn-prediction-weekend-hackathon. Accessed 15 Mar 2023.
28. SMOTE—Version 0.10.1. https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html. Accessed 08 Jun 2023.
29. Lemaître G, Nogueira F, Aridas CK. Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. J Mach Learn Res. 2017;18(17):1–5.
30. ydata-synthetic Python package for synthetic data generation for tabular and time-series data. https://docs.synthetic.ydata.ai/1.3/. Accessed 04 Jul 2023.
31. ctgan · PyPI.https://pypi.org/project/ctgan/. Accessed 08 Jun 2023.
32. EditedNearestNeighbours—Version 0.10.1. https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.EditedNearestNeighbours.html. Accessed 08 Jun 2023.
33. SMOTEENN—Version 0.10.1. https://imbalanced-learn.org/stable/references/generated/imblearn.combine.SMOTEENN.html. Accessed 08 Jun 2023.
34. Pedregosa F, et al. Scikit-learn: machine learning in python. J Mach Learn Res. 2011;12(85):2825–30.
35. XGBoost Documentation—xgboost 2.0.3 documentation. https://xgboost.readthedocs.io/en/stable/. Accessed 19 Mar 2024.
36. sklearn.ensemble.RandomForestClassifier—scikit-learn 1.4.1 documentation. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed 12 Mar 2024.
37. lightgbm.LGBMClassifier—LightGBM 4.3.0.99 documentation. https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html. Accessed 19 Mar 2024.