IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 29 October 2022, accepted 14 November 2022, date of publication 17 November 2022, date of current version 23 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3222883

RESEARCH ARTICLE

Employing a Machine Learning Boosting Classifiers Based Stacking Ensemble Model for Detecting Non Technical Losses in Smart Grids

PAMIR[1], (Graduate Student Member, IEEE), NADEEM JAVAID[1], (Senior Member, IEEE), MARIAM AKBAR[1], (Senior Member, IEEE), ABDULAZIZ ALDEGHEISHEM[2], NABIL ALRAJEH[3], AND EMAD A. MOHAMMED[4]

[1]Department of Computer Science, COMSATS University Islamabad, Islamabad 44000, Pakistan
[2]Department of Urban Planning, College of Architecture and Planning, King Saud University, Riyadh 11574, Saudi Arabia
[3]Department of Biomedical Technology, College of Applied Medical Sciences, King Saud University, Riyadh 11633, Saudi Arabia
[4]Department of Engineering, Faculty of Science, Thompson Rivers University, Kamloops, BC V2C 0C8, Canada

Corresponding author: Nadeem Javaid (nadeemjavaidqau@gmail.com)

This work was supported by the King Saud University, Riyadh, Saudi Arabia, Researchers Supporting Project, under Grant RSP-2021/295.

ABSTRACT In the modern world, numerous opportunities help detect electricity theft happening in electricity grids due to the widespread shifting of people from old metering infrastructure to advanced metering infrastructure (AMI). It is done by studying the consumers' energy consumption (EC) readings provided by smart meters (SM). The literature introduces a variety of machine learning (ML) and deep learning (DL) strategies to use EC data for identifying power theft in smart grids (SGs). However, the existing schemes provide low performance in electricity theft detection (ETD) due to the usage of imbalanced data and using schemes individually. Moreover, the existing detectors are validated using a limited number of performance evaluation measures, which are unsuitable for conducting the model's comprehensive validation. To tackle the problems mentioned above, an ML boosting classifiers-based stacking ensemble model (MLBCSM) is proposed followed by an adaptive synthetic sampling technique (ADASYN) in the underlying work. Data preprocessing, data balancing and classification are the three major parts of the model introduced in this work. Besides, the EC data acquired from the consumers' SMs is used for detecting electricity theft. Moreover, the simulation results reveal that MLBCSM combines the benefits of adaptive boosting (AdaBoost), extreme gradient boosting (XGBoost), histogram boosting (HistBoost), categorical boosting (CatBoost), and light gradient boosting (LGBoost). Additionally, the model's validation is ensured via different metrics. It is deduced via extensive simulations that the proposed model's outcomes are superior to those of the individual models in terms of ETD.

INDEX TERMS AdaBoost, CatBoost, electricity theft detection, healthcare, HistBoost, LGBoost, stacking ensemble model, state grid corporation of China dataset, smart cities, XGBboost.

I. INTRODUCTION

Electricity is one of the significant resources in human life, which is provided to consumers by electric utilities. In return, the electric utilities obtain benefits in the form of money. However, electricity losses occur while dispatching the energy from the generation side to the consumption side. The losses heavily disturb the economic benefits of both utility and electricity consumers. Typically, the division of electric losses is done into two groups: technical and non-technical [1]. The physical state of the power system's devices becomes the reason for technical losses (TLs). These losses can be reduced to some extent (but not fully) by changing the hardware components of the power system. Electricity theft, non-payment of energy consumption (EC) bills, meter installation flaws, accounting errors, etc., become

The associate editor coordinating the review of this manuscript and approving it for publication was Alba Amato.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

the reasons for non-technical losses (NTLs). Among these, electricity theft is responsible for a remarkable loss [2]. Illegal energy usage in multiple ways is referred to as stealing electricity, also known as electricity theft [3]. The principal reason for energy theft is tapping, which is responsible for almost 80% of the total NTLs [1]. The rate of electricity theft in developing countries is more than 30% [4]. Due to energy theft, China incurs a loss as high as 20 billion Chinese Yen per year [4]. The US loses around 6 billion US dollars per year to electricity theft [5]. Conventionally, technicians were hired to study the monthly meter readings over several months to identify the abnormal energy usage. Afterwards, they visited each consumer in-person to look over the connection and status of each energy meter [6]. However, this method of detecting energy theft needs experts' knowledge.

Moreover, the decisions made by the respective domain experts are scarce in contrast to the maximization of the EC readings on a day-to-day basis [5]. The advanced metering infrastructure (AMI) is a significant element of the smart grid (SG) comprising the smart meters (SMs) that record and monitor the EC readings. With the emergence of AMI and SG, a massive amount of EC data are available. Therefore, a new hope is raised when employing machine learning (ML) and deep learning (DL) algorithms for detecting abnormal EC patterns from massive data [5]. These techniques reduce the working load of technicians and obtain better NTLs' detection accuracy values. Besides, ML and DL techniques are used in other fields such as transport, healthcare, agriculture, etc. Recently, many ML and DL-based models have been proposed to tackle the problem of NTLs [7], [8], [9], [10], [11]. These approaches employ the EC readings' history of consumers to analyze the data to detect NTLs in SGs. However, some of these techniques have low detection accuracy. In addition, some of them give high false positive rate (FPR) values. These bad performances are caused due to various reasons. The ML classifiers are individually employed in the abovementioned techniques, and no stacking ensemble model is developed from multiple heterogeneous techniques to achieve improved performance in electricity theft detection (ETD). Furthermore, the imbalanced class problem is not handled, resulting in biased results in terms of majority class samples. Very few performance metrics are considered in some articles for performance validation of their models, which are not enough to perform comprehensive and accurate performance validation of the proposed approaches.

To address the abovementioned limitations, we developed an ML boosting classifiers-based stacking ensemble model (MLBCSM) followed by an adaptive synthetic (ADASYN) sampling technique for detecting NTLs in SGs. In addition, the proposed MLBCSM model is comprehensively validated using eight popular performance measures, namely, accuracy, precision, receiver operating characteristic-area under the curve (ROC-AUC), precision recall-AUC (PR-AUC), F1 score, FPR and false negative rate (FNR).

The following points highlight the vital contributions made in the underlying research article.

- An MLBCSM stacking ensemble model for detecting NTLs in SGs is proposed, similar to the model proposed in [12] for financial market forecasting. It comprises five boosting classifiers: four are considered as base learners, and one is selected as a meta-learner.
- We tackle the data imbalance issue through an ADASYN approach. The approach is employed to oversample the minority class samples to achieve balanced data and avoid biased NTLs' detection results.
- Extensive simulations are conducted on a substantial realistic EC dataset by considering eight performance evaluation measures for comprehensive validations of our proposed model. Simulation results depict that our MLBCSM followed by ADASYN provides magnificently better NTLs' detection performance than baseline models.

Following is the breakdown of the remaining sections of this research paper. Section II presents the related work while the problem statement is presented in Section III. The proposed model's discussion is offered in Section IV. The simulations' findings are provided in Section V while the concluding text is provided in Section VI.

II. RELATED WORK

The authors in [1] suggest ensemble classifiers for ETD in SGs. They employ the EC data gathered from the smart meters (SM) of the commission for energy regulation. The techniques they use for ETD include adaptive boosting (AdaBoost), light gradient boosting (LGBoost), XGBoost, categorical boosting (CatBoost), etc. In addition, EC data is normalized via the min-max normalization technique. However, all the ensemble classifiers are used individually to find FPR and TPR or detection rate (DR) for each ensemble model. Most of the current electricity theft detection (ETD) research is based on ML and DL classifiers because of the development of advanced metering infrastructure (AMI) and SGs. The authors in [4] propose an adaptive time series recurrent neural network (TSRNN) to identify theft consumers in time series EC data. To tackle the data imbalance issue, synthetic minority oversampling technique (SMOTE) is employed. Whereas, the grid search method is leveraged for hyperparameters' optimization of the TSRNN. The analysis is performed using 820 days' EC data from 01 January 2017 to 31 March 2019. Moreover, accuracy, false alarm rate, and true positive rate (TPR) metrics are considered to validate the TSRNN. However, these metrics are limited, and lack in providing the model's comprehensive and fair evaluation. The authors in [5] propose an extreme gradient boosting (XGBoost) classifier to detect electricity theft using EC data from the Irish smart energy trails dataset. The preprocessing of the EC data in terms of dealing with the missing or not a number (NaN) values, outliers, and unscaled data is also done. Furthermore, a data-oriented approach is proposed in [7], in which a gradient-boosting classifier is employed as a data-oriented model for ETD. One new synthetic theft attack is also added to the previously designed six theft attacks. The authors intended to improve FPR and accuracy values using this combined strategy. However, FPR calculation is ignored and 88% accuracy is achieved, which is not as improved as normally needed.

To identify energy theft in SGs, a DL model, a modified convolution neural network (CNN), is developed [8]. Additionally, EC privacy is protected via a Paillier technique. The data of energy consumed by the users acquired from the state grid corporation of China (SGCC) is used to analyze electricity theft. According to the simulation findings, the modified CNN achieves 92.67% accuracy on the ETD task. However, CNN is a standalone classifier that is used for ETD. Moreover, the data imbalance problem is ignored. Furthermore, in [9], a specialized theft detector, named as a deep neural network with low FPR (LFPR-DNN), is proposed to achieve minimum FPR. The data imbalance issue is tackled with the help of the focal loss; the extension of cross_entropy function. In addition, to optimize the FPR value, the particle swarm optimization (PSO) technique is leveraged. LFPR-DNN model's hyperparameters are tuned via grid search method. Moreover, recall or TPR, FPR, AUC, and bayesian detection rate (BDR) performance measures are used for the LFPR-DNN model's validation. However, the performance measures selected for the proposed model's validation prove insignificant to performing proper and extensive validation. In addition, a comparatively poor recall value is obtained, which needs to be improved. Moving further, a stacked sparse denoising autoencoder (SSDAE) is used to identify electricity theft in [10]. To enhance the robustness and feature extraction abilities of the SSDAE, noise and sparsity parameters are added to the autoencoder. In addition, the PSO technique is used to optimize of both sparsity and noise hyperparameters. Moreover, only DR and FPR measures are selected to validate SSDAE, which are not enough to conduct fair and comprehensive performance validation of SSDAE. A CatBoost based theft detector is used in [13] to classify honest and dishonest consumers. In addition, k nearest neighbors (KNN) imputation and min-max scaler methods are used to fill in the missing data and normalize the unscaled data in EC dataset, respectively. Furthermore, SGCC dataset is used for theft analysis. The proposed model obtains 92% TPR, 93% accuracy, and 95% precision values in ETD.

A gradient-boosting theft detection model is proposed in [14], based on three modern gradient-boosting models: LGBoost, XGBoost, and CatBoost. The Irish EC data is used two Irish organizations release that: (1) electric Ireland, which is an Irish gas and electricity utility company, and (2) electric Ireland and sustainable energy authority of Ireland (SEAI), which is an Irish governmental agency. In addition, since the dataset contains all the honest consumers' EC data, revised synthetic six theft attacks are proposed and employed to generate the theft class data synthetically. Afterwards, SMOTE is used to balance the data. However, the three boosting classifiers are employed individually. Furthermore, in [15], a pattern-based context-aware ETD (PCETD) strategy is introduced. KNN and dynamic time warping are used to check the consumers' anomalousness and examine the relationship between two different EC patterns, respectively. In addition, seven novel theft attacks are introduced in this work to address the imbalanced data problem. A stacked autoencoder (SAE) and under-sampling and resampling-based RF (UaRe-RF) ETD approach is put forward in [16]. The extraction of essential features is done via SAE. At the same time, data balancing and ETD are performed via UaRe-RF. Furthermore, for identifying energy theft in the transformers installed at the distribution side, the authors in [17] put forward an efficient approach that used EC curve comparison for the designated task. It is achieved by a static state estimator of EC data obtained from SM. In addition, self-organizing maps and multilayer perceptron artificial neural network are leveraged to identify energy theft users. The Irish EC dataset released by electric Ireland and SEAI is employed. The results depict the model's inferior energy theft detection capability.

III. PROBLEM STATEMENT

Electricity theft is a hazardous activity for electric grids, such as economic destruction, energy scarcity, and grid's instability. Besides, the main goal of an electricity thief is to underpay the amount of the consumed energy. Therefore, efficient detection of electricity theft in the SGs is crucial as it saves the utilities from major loss and helps avoid all the abovementioned issues. Conventionally, individual classifiers were deployed for detecting energy theft in SGs. However, low classification results were obtained [1]. In addition, the proportion of the theft consumers in most cases is tiny. The existing EC datasets are often imbalanced. As a result, the class with more samples is given more consideration in the training stage. The prediction results are negatively affected and biased results are obtained [18]. Furthermore, in most cases, very few validation metrics are considered for the proposed models' evaluation. Moreover, using fewer validation metrics does not yield accurate, appropriate, fair, and detailed validation of a model. It proves to be inefficient in terms of detecting electricity theft in the SGs.

IV. PROPOSED SYSTEM MODEL

In the underlying work, an ETD approach is put forward to identify electricity consumption abnormalities. Data preprocessing, balancing, and classification make up the proposed ETD approach. A comprehensive view, in Fig. 1, presents the proposed system model. All of these components are comprehensively discussed in the subsequent subsections.

A. DATA PREPROCESSING

The SGCC data [19] is used for performing ETD in SGs. The SGCC dataset's metadata is being provided in in Table 1. 38757 honest (non-theft) consumers' EC data and

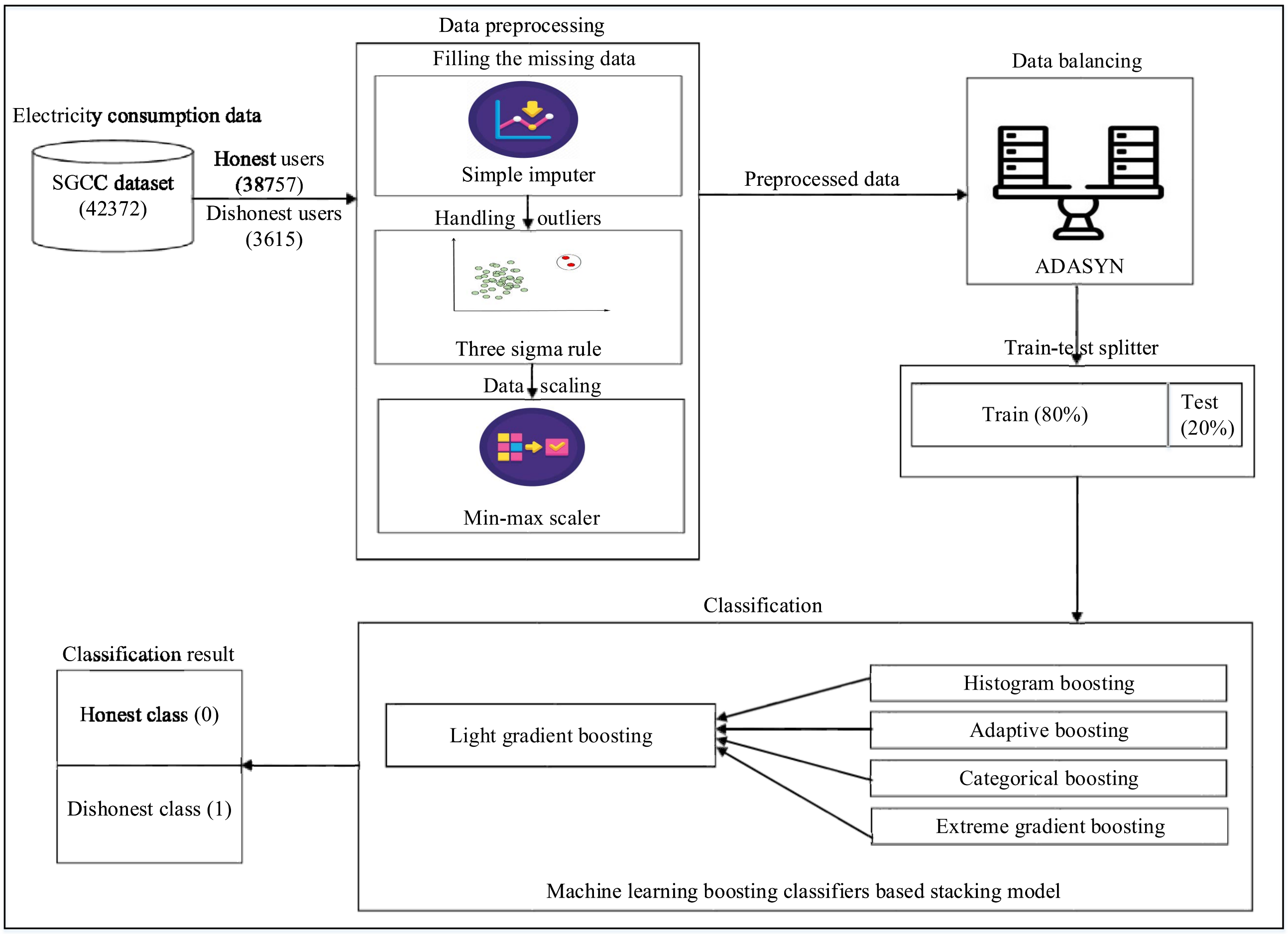


FIGURE 1. Proposed system model.

3615 dishonest (theft) consumers' EC data for the duration of two years and ten months, which is equivalent to 1034 days is present in the dataset. The original dataset contains a large amount of NaN data, outliers, and unscaled data. It is not suitable to train a model using a dataset having missing values, outliers and unscaled data. So, we need to preprocess the dataset first.

EC data contains missing data due to various reasons, such as signal communication errors and hardware device malfunction [20]. To compute the missing data and impute it into the dataset, a simple imputer method (SIM) is used in this article. SIM is implemented using SimpleImputer class with $strategy = mean$ through the scikit-learn library in Python [21]. In addition, as previously mentioned, the dataset also contains some outlier values, which affect the model's predictive performance. Therefore, outliers are reset via three sigma rule of thumb [22], implemented using Equation 1 [22].

$$f(x_{i,a}) = \begin{cases} X_{avg} + 3.X_\sigma, & x_{i,a} > X_{avg} + 3.X_\sigma \\ x_{i,a}, & Otherwise. \end{cases} \quad (1)$$

where $x_{i,a}$ indicates the EC related to the i-th electricity consumer at a-th time slot (i.e., day). In our scenario, $i = 42373$ and $a = 1034$. X is a dataframe created in Python that contains several $x_{i,a}$ EC values. X_σ represents the standard deviation of the dataframe X. Moreover, X_{avg} represents the average value of X. After dealing with the outliers and missing values, it is time to deal with data diversity. Thus, min-max scaler is employed for normalizing the data [23] via Equation 2 [23].

$$f(x_{i,a}) = \frac{x_{i,a} - X_{min}}{X_{max} - X_{min}}, \quad (2)$$

where the minimum and maximum values of X are denoted by X_{min} and X_{max}.

B. DATA BALANCING

In this subsection, the data balancing component is exploited to tackle the severe class imbalanced issue existing in the SGCC dataset. Generally, the ETD datasets are severely imbalanced. It means there are many honest users' samples in them while theft users' samples are limited. This work employed ADASYN [24] to deal with data imbalanced

TABLE 1. SGCC dataset's metadata.

Description	Values
Time duration	01-01-2014 to 31-10-2016
Honest records	38757
Dishonest records	3615
Overall records	42372
Honest data percentage	91.47%
Dishonest data percentage	8.53%

problems. In real-life, examples of everyday consumers can be numerously found. On the other hand, theft of consumers' data can be rarely found. It is the reason why the class imbalance problem exists in the ETD datasets. In such a scenario, if we train our ML or DL classifiers with the original ETD dataset, in the prediction process, classifiers generate results biased towards the majority (normal) class and generate very bad results on the minority (theft) class examples, which are very significant to be correctly classified. For example, suppose you are classifying the credit card fraud users and there are only five fraud transactions out of one million transactions. It means the dataset is severely imbalanced. If you train your model on this imbalanced dataset, in testing phase, your model classifies all the examples as negative (non-fraud transactions). As a result, the model achieves an accuracy of 99.9995%. In this way, the model arguably learns to classify and predict the non-fraud transactions without caring what the input is, which is fully purposeless [25]. To fight this problem, data balancing is needed. The data must be balanced with the similar or same frequency of fraud and non-fraud samples.

To perform data balancing, traditionally, oversampling and undersampling techniques are employed. In undersampling techniques, most examples are deleted until they are equivalent to the minority samples. However, it is data inefficient. The deleted data may contain important information regarding the majority (non-fraud) class. To deal with this data inefficiency issue, oversampling is employed. In oversampling, the examples of the minority (theft) class are duplicated (copied) N-times until its sample count becomes similar or equivalent to the majority class sample count. However, the model overfits the minority class due to multiple sample copies in such cases. Therefore, to combat both the abovementioned issues created due to oversampling and undersampling techniques, ADASYN [25] is introduced. It is a data balancing method that creates synthetic examples by not copying the same minority class' data samples, but, it generates more synthetic data for harder-to-learn samples. The steps to be taken for creating synthetic data using ADASYN are given below [25].

1) Compute the ratio between minority and majority data samples using Equation 3.

$$R = \frac{S_{min}}{S_{maj}}, \tag{3}$$

where the numbers of majority and minority class samples are denoted by S_{maj} and S_{min}.

2) Compute the total number of the synthetic minority class data samples to be generated. This is done using Equation 4.

$$TM = (S_{maj} - S_{min})\beta, \tag{4}$$

where β represents the desired post-ADASYN implementation ratio of minority:majority data samples. $\beta = 1$ denotes the post-ADASYN perfectly balanced dataset. TM indicates the total number of minority class samples to be generated.

3) Find the K nearest neighbours (KNNs) for each minority sample and compute r_i. r_i is calculated using Equation 5.

$$r_i = \frac{N_{maj}}{K} \tag{5}$$

where N_{maj} is the number of majority samples in KNNs of a specific minority sample. K shows the number of nearest neighbours to be chosen for a specific minority example. r_i shows the dominance and superiority of the majority class in KNNs of a specific minority sample. The maximum value of r_i comprises the majority of samples that are hard to learn.

4) Normalize and scale r_i values using Equation 6

$$.\hat{r}_i = \frac{r_i}{\sum r_i}, \tag{6}$$

5) Compute the number of synthetic samples to be generated per each neighborhood using Equation 7.

$$TM_i = TM\hat{r}_i, \tag{7}$$

if the value of r_i is maximum, it means that the neighbourhood of a specific minority sample is largely composed of the majority class samples. So for such a situation, more artificial minority samples will be generated. In this way, we can say ADASYN has an adaptive nature; more minority samples were created for difficult to learn minority samples.

6) Generate TM_i samples for every single minority neighbourhood. First select the minority sample x_i. Afterwards, select another minority sample in the neighbourhood of x_i randomly. Then, create the new sample using Equation 8

$$NX_i = x_i + \lambda(rx_i - x_i), \tag{8}$$

where λ represents a random value between 0 and 1. NX_i shows the newly generated minority sample. rx_i and x_i are the minority samples in the same neighborhood area.

Using the above steps, the dataset is now balanced and the balanced data is then passed to the proposed MLBCSM for accurate detection of electricity theft with low FPR value.

C. CLASSIFICATION

After completing the data preparation and balancing steps, data classification is performed to detect electricity theft using the proposed MLBCSM. Electricity theft and non-theft consumers are classified using the proposed MLBCSM. We selected the stacking ensemble strategy in this article for the reason that the stacking ensemble strategy outperforms the techniques employed in the literature. Moreover, stacking ensembles recently won many data science competitions specifically Kaggle and Netflix for classification problems [26]. Hence, stacking ensembles are considered the best of all classifiers. Therefore, we chose the stacking ensemble strategy to obtain maximum ETD performance accuracy. The stacking ensemble strategy is an efficient strategy that comprises multiple standalone classifiers at two levels (level-0 and level-1), where level-0 and level-1 classifiers are also called base-learners and meta-learner, respectively. Our proposed MLBCSM consists of multiple ML boosting classifiers as base and meta-learners. AdaBoost, XGBoost, HistBoost, and CatBoost are selected as level-0 learners while LGBoost is chosen as a level-1 learner for our proposed MLBCSM. The pseudo-code of the proposed MLBCSM for theft and non-theft consumers' classification is given in Algorithm 1. More in-depth details about these base and meta-learners are provided in the subsequent subsections.

Algorithm 1 MLBCSM Algorithm

```
Input: ADASYN based balanced dataset
Output: Predict consumers' labels
1:  Create HistBoost (first level-0 learner):
2:  HistGB = HistGradientBoostingClassifier()
3:  Create AdaBoost (second level-0 learner):
4:  ADB = AdaBoostClassifier()
5:  Create CatBoost (third level-0 learner):
6:  CB = CatBoostClassifier()
7:  Create XGBoost (fourth level-0 learner):
8:  XG = XGBClassifier()
9:  Create a dictionary namely Base_estimators that contains
    all level-0 learners:
10: Base_estimators = [('HB',HB), ('ADB', ADB), ('CB',
    CB), ('XG', XG)]
11: Create LGBoost (level-1 learner):
12: LGB = LGBMClassifier()
13: Create a stacking classifier and pass the Base_estimators
    dictionary and level-1 learner:
14: Stacking_model = StackingClassifier( estimators =
    Base_estimators, final_estimator = LGB)
15: Train the stacking model:
16: Stacking_model.fit(X_train, y_train)
17: Predict labels by the stacking model:
18: SM_test=Stacking_model.predict(X_test)
```

1) LEVEL-0 LEARNERS

Level-0 learners, i.e., CatBoost, AdaBoost, HistBoost, and XGBoost, are trained using the training dataset. The predictions along with the original labels are then passed to the level-1 classifier for training [27].

A detailed description of each of the four base-learners is provided below.

- **Adaptive boosting**

  AdaBoost [28] is a popular model in data science. It was developed for the first time by Freund and Shapire in 1996 [29]. It is built based on the concept of boosting type ensemble strategy where the main idea of boosting is that multiple weak learners can be combined to create a robust algorithm using voting strategy. The learner that slightly surpasses a tossing coin in terms of prediction result is regarded as a weak learner. Such a learner achieves 55% or any other value close to 50% accurate results. In other words, a classifier with the loss of less than but close to 50% is called a weak learner. In this scenario, the in-sample loss rate is the count of wrongly classified samples, i.e., $(y_i \neq G(x_i))$, divided by the total data samples' size (N), as given in Equation 9 [28].

$$\overline{error} = \frac{1}{N}\sum_{i=1}^{N} I(y_i \neq G(x_i)) \tag{9}$$

  In boosting, multiple weak learners are trained sequentially using a consecutive modified version of data points. It means that in the first boosting cycle, a weak learner $(G_1(x))$ is trained and prediction results are generated, in which we can observe that some of the examples are misclassified. In the second boosting cycle, some weight (W_i) is assigned to each of the examples; however, the previously misclassified records are weighted more than correctly classified records to force the second weak learner to learn and correctly classify them. Now, the second weak learner correctly classifies the previously misclassified observations. However, it may misclassify the previously correctly classified observations. After iterating this process for M times, weak learners are combined using a robust meta-learner $(G(x))$. The final meta-learner now assigns a prediction label to each record using a weighted majority voting mechanism provided in Equation 10 [28].

$$G(x) = sign\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right), \tag{10}$$

  where α is the weight of the weak learners in the final majority voting mechanism. In AdaBoost, multiple weak learners (i.e., stumps in AdaBoost) are trained sequentially. These weak learners create a meta-learner that obtains prediction results employing a weighted majority voting strategy. In every boosting round, more weights are assigned to the previously wrongly predicted samples. This process is wrapped up using Algorithm 2 [28].

- **Extreme gradient boosting**

  XGBoost [30] was introduced by Tianqi Chen and Carlos Guestrin at the Washington University. In gradient

Algorithm 2 AdaBoost Algorithm

```
1: Initialize the same weights to all records $W_i$=1/N, i=1, 2, 3, ..., N.
2: for m = 1 to M do
3:     Train a weak learner $G_m(x)$ on training data using $W_i$
4:     Calculate
           $$error_m = \frac{\sum_{i=1}^{N} W_i I(y \neq G_m(x_i))}{\sum_{i=1}^{N} W_i}$$
5:
6:     Calculate $\alpha_m = log((1 - error_m)/error_m)$
7:     Put $W_i = W_i.exp[\alpha_m.I(y = G_m(x_i))], i = 1, 2, 3, \ldots, N$.
8: end for
9: Final output is obtained using Equation 10
```

Algorithm 3 XGBoost Algorithm

```
Input Features: $X$; labels: $Y$; loss function: $l(y, f(x))$; total number of trees: M
1: for i = 1 to M do
2:     Initialize m-th decision tree $f_m(x_i)$
3:     Calculate $g_i = \partial_{\hat{y}_i^{(m-1)}} loss\left(y_i, \hat{y}_i^{(m-1)}\right)$
4:     Calculate $h_i = \partial^2_{\hat{y}_i^{(m-1)}} loss\left(y_i, \hat{y}_i^{(m-1)}\right)$
5:     Employ the statistics to grow a new decision tree greedily $f_m(x_t)$: $obj^m = -\frac{1}{2}\sum_{j=1}^{M}\frac{G_j^2}{H_j+\lambda} + \gamma M$
6:     Add the best tree $f_m(x_t)$ into current model based on: $\hat{y}_i^{(m)} = \hat{y}_i^{(m-1)} + \epsilon f_m(x_i)$
7: end for
8: Until all M weak learners are executed
9: Create a strong classification model based on weak trees.
10: Result in a prediction value, i.e., 0 or 1
```

boosting, the weak learners are trained using a gradient descent optimizer and a differentiable loss function. Therefore, it is called gradient boosting algorithm. XGBoost is a computationally faster and extremely effective-type of gradient boosting algorithm [31]. As XGBoost is fast to execute and obtains better predictive results, we chose it as one of the base-learners in our proposed MLBCSM. In XGBoost, we train a learner using the gradient of the loss from the previous learner. Moreover, in XGBoost, the gradient boosting technique is modified to make it able to work with any of the differentiable loss functions [32]. XGBoost integrates the decision trees with a gradient boosting mechanism. At each training round of a tree (weak learner), the residual of the previous tree is used in the next tree to minimize the loss function [33]. XGBoost also avoids overfitting problems and minimizes computational complexity. The final classification result, in the end, is acquired by combining all the weak learners, i.e., decision trees. The final output is predicted using Equation 11 [33].

$$G(X_i) = \sum_{m=1}^{M} g_j(X_i) \tag{11}$$

where $g_j(X_i)$ represents the output generation function of each weak learner. X_i denotes the data given to a weak learner. The algorithmic description of the XGBoost is given in Algorithm 3 [34]. Compared with the LGBoost and CatBoost, XGBoost cannot deal with the categorical data and only handles numerical data like a random forest bagging classifier. However, if someone wants to process categorical data using XGBoost, some encoding methods, such as one-hot encoding, label encoding, etc., must be applied first.

In Algorithm 3, $f_m(x_i)$ is the best weak learner in m-th iteration. $\hat{y}_i^{(m-1)}$ is the current classification model. Whereas, $\hat{y}_i^{(m)}$ is the new classification model. ϵ is the shrinkage parameter used to avoid overfitting. obj^m is the objective of XGBoost algorithm. Besides, g_i is the loss function and h_i is the second derivative of the loss function used to define a loss calculation function that is twice differentiable.

- **Histogram boosting**
  The gradient-boosting decision tree algorithm requires more training when dealing with a big dataset, and sometimes the prediction accuracy is compromised. HistBoost is an effective model for dealing with a huge dataset [12]. HistBoost minimizes the training time without degrading the accuracy. Consequently, it can be stated that HistBoost is an algorithm that rapidly trains weak learners in a gradient-boost framework. The HistBoost's splitting procedure is different from other gradient boosting methods. Instead of determining the splitting points on feature values, the HistBoost buckets the continuous values of features into discrete bins, using which multiple feature histograms are created. As HistBoost is both training time and memory-consumption efficient algorithm, we select it in our proposed MLBCSM as one of the base (level-0) classifiers. More details about how histogram-based algorithm works can be found in Algorithm 4 [35]. Furthermore, the gradient-boosting decision trees ensemble training process is expedited in the proposed work. Big training datasets containing tens of thousands of rows or even more lead to a deadly slow creation of decision trees as splitting points on each value, for each dimension, must be taken into account while creating the trees [36]. Moreover, the training process of weak learners, usually the decision trees, appended into the ensemble model can be expedited due to binning (discretizing) the continuous input features to only a few hundred unique values. Thus, the gradient boosting ensemble models, which implement this (binning) method and adjust (tailor) the training model over the input features that follow the transform made by binning, is known

as histogram-based gradient boosting ensemble models. Furthermore, suitable data structures like histograms can be employed to represent data discretization. In this way, the decision trees' creation algorithm can be further adjusted for histograms' effective and efficient employment in creating every decision tree. Hence, from the above discussion, it is concluded that a gradient-boosting technique supporting histogram data structures is referred to as the HistBoost technique.

Algorithm 4 HistBoost Algorithm

```
Input Training set: $X$; Maximum depth: $d$; Feature dimension: $m$; Nodes set: $\{0\}$; Row set: $\{0, 1, 2, \ldots\}$
1:  for i = 1 to d do
2:      for Node in NodeSet do
3:          $UsedRows = RowSet[Node]$
4:          for k=1 to m do
5:              $H = newHistogram()$
6:              Create histogram
7:              for j in UsedRows do
8:                  $bin = X.f[k][j].bin$
9:                  $H[bin].y = H[bin].y + X.y[j]$
10:                 $H[bin].n = H[bin].n + 1$
11:             end for
12:             Find the optimal split on H
13:         end for
14:     end for
15:     Update the Node set and Row set based on the optimal splitting points
16: end for
```

- **Categorical boosting**
  CatBoost was introduced by Yandex (a technology company in Russia) in 2018 [37]. It is a better technique than other gradient-boosting models due to its ability of directly tackle categorical features (without applying any encoding scheme) and faster training [12]. Besides, categorical features, it can also handle textual and numerical features as well. However, it has a better handling method for categorical data [38]. As CatBoost directly supports the categorical features (without using any encoding method), therefore, it is called CatBoost [39]. Generally, gradient boosting based models perform better in huge and small datasets. The algorithmic details of the CatBoost are provided in Algorithm 5 [40]. XGBoost and LGBoost are the widely employed ensembles; however, CatBoost is the most modern ensemble model. CatBoost is the successor of MatrixNet model that is broadly employed within the Yandex company for forecasting, recommendations, and ranking tasks. The CatBoost uses Algorithm 5 to update base predictors and compute model values for the gradient calculation.
  Based on Algorithm 5, for each sample X_k, we train an independent model M_k that is not updated by a gradient

Algorithm 5 CatBoost Algorithm

```
Input Features set: $X_k$; Labels: $Y_k$, where $k = 1, 2, \ldots, n$ ordered based on $\sigma$; Number of trees: $I$
$M_i = 0$ for $i = 1, \ldots, n$
1:  for iteration = 1 to I do
2:      for i = 1 to n do
3:          for j = 1 to i−1 do
4:              $g_j = \frac{d}{da} loss(y_j, a)$, $a = M_i(x_j)$
5:          end for
6:          M=$LearnOneTree((X_j, g_j)$ for $j = 1$ to $i - 1)$
7:          M=$M_i + M$
8:      end for
9:  end for
10: return $M_1, M_2, \ldots, M_n; M_1(X_1), M_2(X_2), \ldots, M_n(X_n)$
```

estimation for this sample. Using M_k, we compute (estimate) gradient on X_k and leverage this computed gradient to get the output tree. $loss(y_j, a)$ is the loss optimization function in which y is the label (target) value and a represents the formula (predicted label) value.

2) LEVEL-1 LEARNER

In order to make the final classification decision based on the results generated by the level-0 learners, the level-1 learner is selected. LGBoost is chosen as level-1 classifier for the proposed MLBCSM.

- **Light gradient boosting**
  LGBoost is one of the widely used classifiers. It was introduced by Guolin Ke in 2017 [41]. Basically, LGBoost enhances the basic gradient boosting technique by appending the ability to focus on samples with comparatively huge gradients and feature selection, which lead to faster training as well as enhanced prediction results of the classifier [42]. Exclusive feature bundling (EFB) is an automatic feature selection technique, used for bundling the sparse (i.e., mostly zero and rarely nonzero) mutually exclusive features. Gradient-based one-sided sampling (GOSS) focuses on the training samples that comparatively have higher gradients and exclude the notable portion of the samples with low gradients to estimate (compute) the information gain. As the data samples having large gradient values play a significant role in information gain's computation, GOSS can yield accurate computation of the information gain with a smaller dataset. Since GOSS focuses on examples with larger gradients, it results in faster learning and minimizes the computational speed of the algorithm. Together, the two modifications above expedite the training time of the model upto 20 times. Due to these significant qualities, we selected LGBoost as a meta-learner for our proposed MLBCSM. It can be concluded that LGBoost consists of a gradient-boosting algorithm combining EFB and GOSS. The algorithm of LGBoost is shown in Algorithm 6 [43].

```
Algorithm 6 LGBoost Algorithm
Input Training set: D=$(x^1, y^1)$, $(x^1, y^1)$, ..., $(x^N, y^N)$; Loss function: $L(y, \theta(x))$; Maximum Iterations: M
High gradient data sampling ratio: a; low gradient data sampling ratio: b
1:  Combine mutually exclusive features using EFB method
2:  Put $\theta_0(x) = argmin_c \sum_{i=1}^{N} L(y_i, c)$
3:  for m=1 to M do
4:      Compute absolute gradient values:
        $$r_i = \left| \frac{\partial L(y, \theta(x))}{\partial \theta(x_i)} \right|_{\theta(x)=\theta_{m-1}(x)}, i = 1, 2, \ldots, N$$
5:      Resample the data using GOSS:
        $$TopN = a \times len(D); RandN = b \times len(D);$$
    $Sorted = GetSortedIndices(Abs(r));$
6:  $A = Sorted[i : TopN];$
7:  $B=RandPick(Sorted[TopN : len(D)], RandN);$
8:  $D' = A + B;$
9:      Compute information gains using:
10:
        $$V_j(d) = \frac{1}{n}\left( \frac{\left(\sum_{x_i \in A_l} r_i + \frac{1-a}{b} \sum_{x_i \in B_l} r_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_i \in A_r} r_i + \frac{1-a}{b} \sum_{x_i \in B_r} r_i\right)^2}{n_l^j(d)} \right)$$
11:     Build a new decision tree $\theta_m(x)'$ on $D'$
12:     Update $\theta_m(x)$=$\theta_{m-1}(x) + \theta_m(x)$
13: end for
14: return $\theta'(x)$=$\theta_M(x)$
```

In Algorithm 6, step 5 shows that GOSS first sorts out the data samples based on absolute value of their gradients and choose top $a \times len(D)$ or 100% of the samples. D represents the training data. $frac1 - ab$ is the constant value that is multiplied with the summation of absolute gradient values when computing the information gain in step 6 of the algorithm.

V. SIMULATION RESULTS AND DISCUSSION

The proposed model's simulations' settings, the performance evaluation measures, and the simulation results of the proposed and baseline (individual) classifiers with respect to eight various performance measures are discussed in this section. More details are available in the subsequent subsections.

A. SIMULATIONS' SETTINGS

The proposed model for ETD employs the EC readings obtained by the SGCC [19] for analyzing electricity theft. Moreover, the simulations are performed using the DELL Intel core i5-2450M system with a 500 GB hard drive and a total 8 GB RAM in two slots. Python programming language with scikit-learn, lightboost, xgboost, and catboost ML libraries is used to implement the proposed MLBCSM. Google Colab is employed to execute Python's code on cloud servers owned by Google. By default, the SGCC dataset contains 42372 data instances and 1034 columns (features). From 42372 instances only 3615 instances belong to the theft (abnormal) class and the rest, i.e., 38757 instances belong to the non-theft (normal) class. The classes' distribution ratio is 8.53% and 91.47%. It is clear from the distribution ratio that the dataset is severely imbalanced. To balance the dataset, ADASYN is used. It oversamples the minority class (theft class) instances to raise the total number of instances from 42372 to 77050. As a result, the classes' distribution ratio becomes almost equal. Hence, the dataset is balanced.

B. PERFORMANCE EVALUATION MEASURES

This subsection provides a detailed discussion of the selected performance. In supervised ML algorithms, the data with the proper labels and features are passed to the algorithm for training purposes. Afterwards, the trained classifier is tested to evaluate its ability to predict and generalize unlabeled data. The said models' are assessed via accuracy, F1 score, FPR, FNR, ROC-AUC, precision, recall, and PR-AUC metrics [44], [45] are considered the most appropriate and reliable metrics that can be employed for a fair and comprehensive evaluation. However, in [7], [9], and [10], very few inappropriate performance evaluation metrics are employed, which are not enough for fair and comprehensive evaluations of their models. Therefore, to conduct a fair and extensive evaluation of our proposed MLBCSM, accuracy, ROC-AUC, F1 score, FPR, FNR, precision, recall, and PR-AUC are considered. The calculation of all the selected metrics is based on the confusion matrix [26], which consists of four unique values defined below.

- False positives (FP): if an honest (non-theft) electricity consumer is classified as dishonest (theft) by the model.
- False negatives (FN): if the classifier predicts a theft consumer as non theft.
- True positives (TP): if a theft consumer is predicted as theft by the model.
- True negatives (TN): if an honest consumer is classified as honest by the classifier.

The performance measures are elaborated on below.

1) ACCURACY

It is one of the most often used performance measures. It can be regarded as the proportion of all categorized samples that were correctly classified [46]. It is suitable to employ when there is an equal frequency of samples from all classes. Accuracy is calculated via Equation 12 [26].

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{12}$$

TABLE 2. Comparison of the proposed MLBCSM with standalone models.

Classifier	Accuracy	ROC-AUC	Precision	Recall or DR	F1 score	PR-AUC	FNR	FPR
HistBoost	0.85678	0.85682	0.82631	0.88028	0.85244	0.88977	0.11972	0.16405
AdaBoost	0.71966	0.71972	0.67375	0.74247	0.70644	0.79169	0.25753	0.29932
CatBoost	0.90870	0.90873	0.88140	0.93253	0.90624	0.92764	0.06747	0.11271
XGBoost	0.77774	0.77782	0.71601	0.81740	0.76335	0.83324	0.18260	0.25324
LGBoost	0.86139	0.86143	0.82968	0.88620	0.85701	0.89301	0.11380	0.16050
Proposed MLBCSM	0.92395	0.92396	0.91458	0.93222	0.92332	0.94129	0.06778	0.08405

2) ROC-AUC

The ROC curve, where the y-axis displays the TPR and the x-axis displays the FPR, illustrates how well a binary classifier performs on the positive class. Recall or sensitivity are other names for TPR [45]. The TPR and FPR are calculated as follows [26], [45].

$$TPR\ or\ Recall = \frac{TP}{TP + FN} \tag{13}$$

$$FPR = \frac{FP}{FP + TN} \tag{14}$$

A model is considered to have no discriminative ability between theft and non-theft classes if it forms a diagonal line between TPR of 0 and FPR of 0, i.e., ($coordinate(0, 0)$ or classify all negative (honest)) to a TPR of 1 and FPR of 1, i.e., ($coordinate(1, 1)$ or classify all positive (theft)). Thus, ROC-AUC shows a classifier's discriminative power between TPR and FPR.

3) PR-AUC

Precision calculates the number of correctly predicted positive results by a classifier. It is calculated by following formula [26], [45].

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

Precision's output is between 0 and 1 where 1 shows perfect precision and 0 shows no precision. Furthermore, out of all positive (theft) predictions, recall measures the proportion of true (correct) positive predictions. The computation formula for the recall is given in Equation 13. Its output also ranges between 0 and 1. 1 means perfect recall and 0 means no recall. Both recall and precision focus only on the theft samples [45]. Consequently, in the precision-recall (PR) plot, recall is shown on the abscissa and precision is presented on the ordinate. A no-skill classifier, having equal values of precision and the count of positive (minority) samples, is illustrated by a horizontal line. Contrarily, the curve given by the perfect (skillful) classifier inclines towards the (1,1) coordinate, and is denoted by PR-AUC. The PR curve value is 0.5 in the balanced case of data. The most appropriate measure for binary classification techniques based on imbalanced data is the PR curve since it pays attention to the positive class [45].

4) F1 SCORE

For binary classification models' evaluation, F1 score is a widely used metric [47]. It is computed using precision and recall scores. Its value ranges between 0 and 1. The said metric is computed using Equation 16 [26].

$$F1\ score = 2 * \frac{R * P}{R + P} \tag{16}$$

where R denotes Recall and P denotes Precision.

5) FNR

It is another important performance metric that is rarely employed in evaluating the models designed for ETD in SGs. But, it is very significant to consider it for evaluating ETD models. A high FNR leads to a vast and considerable problem as compared to a high FPR. It is risky and threatening to wrongly classify a theft consumer as an honest consumer [48]. The high FNR value leads to high electricity loss, financial loss, energy supply quality loss, and grid safety loss to the electric utility. The mathematical formula to calculate FNR is provided in Equation 17 [49].

$$FNR = \frac{FN}{FN + TP} \tag{17}$$

C. PROPOSED MLBCSM's PERFORMANCE RESULTS

Table 2 and Fig. 2 show the performance comparison of the proposed MLBCSM and another baseline (individual) models. All the individual classifiers are chosen and implemented with their default parameters. Moreover, individual or standalone classifiers with default parameters are then combined using a stacking ensemble mechanism to develop our proposed MLBCSM for detecting electricity theft in SGs. The dataset split is performed using *train_test_split* class in scikit-learn library in Python. The split ratio for testing and training data is 20% and 80%.

Table 2 and Fig. 2 present that our proposed MLBCSM outperforms the standalone models, CatBoost, HistBoost, XGBoost, AdaBoost, and LGBoost, for different performance measures, given in Table 2 and Fig. 2. However, in terms of recall and FNR, CatBoost generates slightly better results than our proposed model. Since CatBoost has an overfitting detection mechanism by default, its results are slightly better in terms of recall and FNR than all the models used in the paper. It is concluded that our proposed model achieves poor performance in terms of FNR and recall. However, better performance is achieved in accuracy, F1 score, ROC-AUC, precision, PR-AUC, and FPR. Moreover,

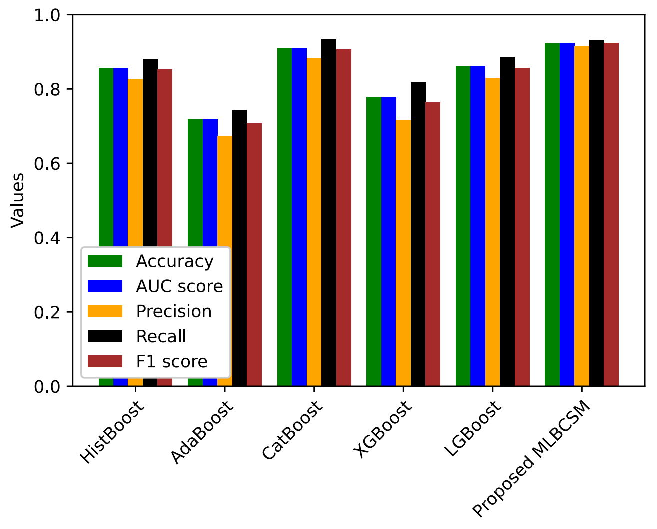


FIGURE 2. Comparison of the proposed MLBCSM with other benchmarks.

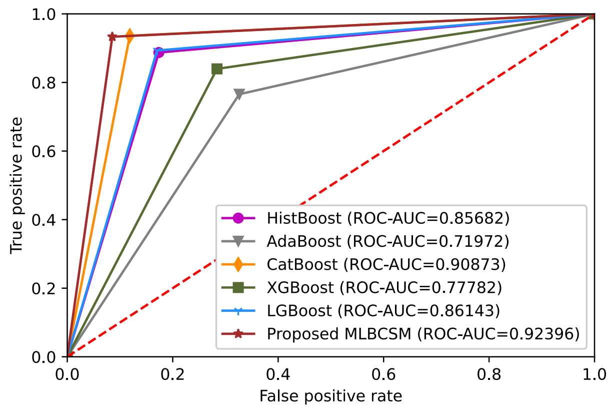


FIGURE 3. ROC-AUC results' comparison.

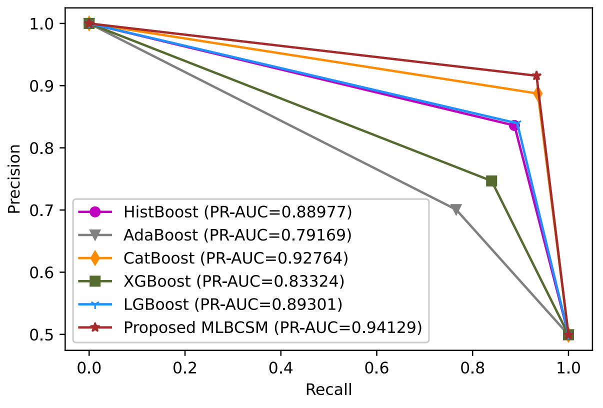


FIGURE 4. PR-AUC results' comparison.

two key characteristics enable CatBoost to provide better results than our proposed and other baselines with respect to FNR and recall. The first one is building balanced trees, which helps CatBoost control overfitting and leads to improved performance. The second one is that CatBoost employs the concept of ordered booting (a permutation-based approach to fit a technique on a subset of data while computing the residuals on another subset), which protects CatBoost from overfitting and target leakage problems. Thus, resulting in improved theft detection in terms of all the above mentioned performance metrics. Furthermore, the reason for the improved results generated by our proposed model is that our proposed MLBCSM exploits and take benefits from the multiple good-performing individual classifiers (i.e., CatBoost, AdaBoost, HistBoost, LGBoost, and XGBoost).

After our proposed model and CatBoost, LGBoost provides promising results, as seen in Table 2 and Fig. 2. This classifier generates good results because it has some unique properties over the other boosting models, i.e., abilities to keep full attention on data samples with higher gradients and feature selection, which are achieved using GOSS and EFB methods, respectively and these unique properties make LGBoost able to produce better predictive results with faster training. On the other hand, AdaBoost is the worst-performing classifier. Its sensitivity to distorted (noisy) data and it can only perform better on a quality dataset (dataset free of outliers and noisy data). With noisy data, AdaBoost is prone to overfitting and provides poor classification results. SGCC dataset may contain some noisy data due to which AdaBoost gets prone to the overfitting issue and generates the worst results among other models implemented in this article.

Fig. 3 shows the ROC curves of the proposed MLBCSM and baselines. The proposed scheme achieves a 0.92396 ROC-AUC value, which is better than all the baselines, HistBoost, AdaBoost, CatBoost, LGBoost, and XGBoost. It simply means that our proposed model very effectively differentiates, and separates the normal and abnormal classes, as it can exploit multiple individual classifiers and perform better than single learners. Furthermore, the PR-AUC is presented in Fig. 4. In the case of ETD, both precision and recall are crucial for utility companies. The maximum PR-AUC score shows the efficiency of the model. The proposed MLBCSM yields a PR-AUC value of 0.94129, which is the maximum as compared to all the baselines under consideration. This proves that our proposed model is advantageous to electric utilities in pinpointing the energy fraudsters and saving maximum energy losses.

FNR is also thought as an important metric. In FNR, the abnormal energy consumers are predicted as normal by the classifier, which is very dangerous and negatively affects the power utilities in terms of financial loss, energy loss, energy supply quality, and power system safety. Thus, FNR is needed to be minimized. Therefore, we considered and calculated this metric, as shown in Fig 5. The proposed model yields the FNR value of 0.06778, which is the minimum value among all the baseline classifiers. On the other hand, AdaBoost achieves the FNR value of 0.25753, which is the highest value among all employed classifiers.

FPR is an important performance measure in which the non-theft energy consumers are considered as theft, which maximizes the misclassification rate of the classifier. There

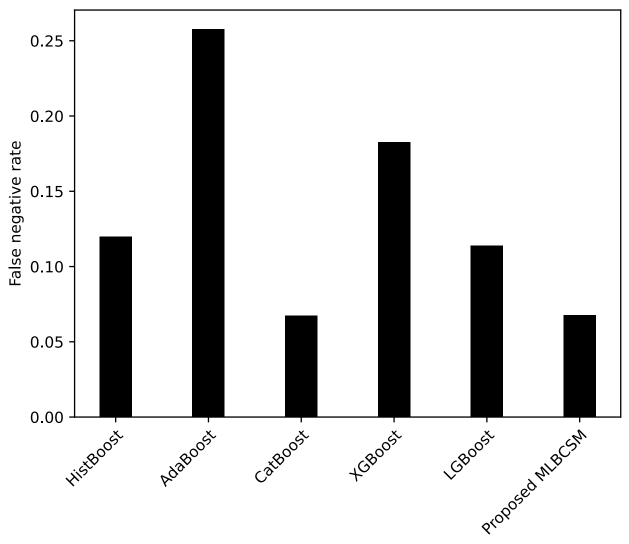


FIGURE 5. FNR results' comparison.

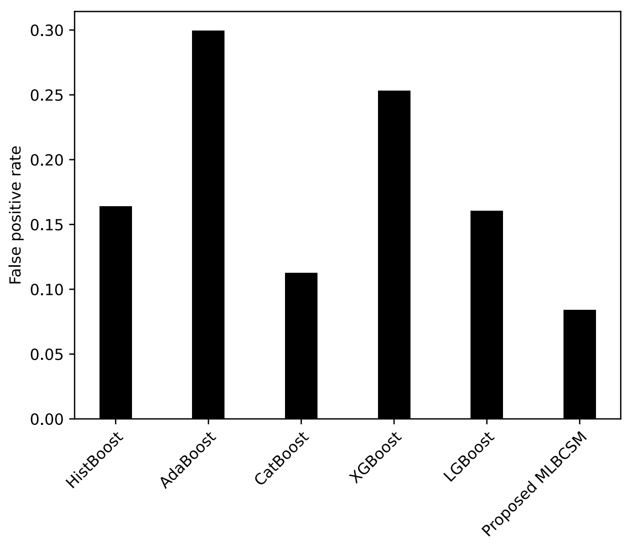


FIGURE 6. FPR results' comparison.

exists a direct relation between FPR score and the on-site inspection cost. One of the key objectives of ETD is to reduce this cost. Therefore, FPR is considered a significant measure for ETD in SGs. We computed FPR in this work and is shown in Fig. 6. Our proposed MLBCSM obtains the minimum FPR value of 0.08405. Whereas, AdaBoost has the maximum FPR of 0.29932. It is worth mentioning that in stacking ensemble models, the performance of the meta-learner is dependent upon the level-0 (base) learners and therefore, we select powerful boosting classifiers for level-0, which provide accurate predictions to the meta-learner that have a positive impact on classification results of the overall stacking ensemble model. This is why our proposed stacking model obtains better results than the baselines. Moreover, AdaBoost's worst performance in terms of FPR is that it needs a good-quality dataset to perform well. Otherwise, it faces an overfitting problem due to noisy data.

Finally, our proposed MLBCSM's superiority in terms of having the highest accuracy, ROC-AUC, F1 score, PR-AUC, precision, and the lowest FPR values as compared to the baselines is proved through extensive simulations.

VI. CONCLUSION

In the work performed in this research, an MLBCSM is introduced as a binary classifier for electricity consumers' classification. The publicly available SGCC data is used in this study. Moreover, ADASYN is leveraged as a data balance for oversampling the minority samples that are difficult to learn. For performing classification, the data balanced through ADASYN is forwarded to MLBCSM. The results generated by the proposed MLBCSM are compared with five standalone classifiers. The proposed model is extensively validated using eight performance evaluation measures and compared with the standalone models. The proposed model achieves 92.395% accuracy, 92.396% ROC-AUC, 91.458% precision, 92.332% F1 score, 94.129% PR-AUC, 8.405% FPR, 6.778% FNR, and 93.222% recall scores. We conclude from the simulations that our proposed model obtained enhanced performance compared to its baselines in terms of ETD. Thus, our proposed MLBCSM followed by ADASYN proved well-suited for correctly classifying all the actual positive theft cases.

REFERENCES

[1] S. K. Gunturi and D. Sarkar, "Ensemble machine learning models for the detection of energy theft," *Electric Power Syst. Res.*, vol. 192, Mar. 2021, Art. no. 106904.

[2] Pamir, N. Javaid, S. Javaid, M. Asif, M. U. Javed, A. S. Yahaya, and S. Aslam, "Synthetic theft attacks and long short term memory-based preprocessing for electricity theft detection using gated recurrent unit," *Energies*, vol. 15, no. 8, p. 2778, Apr. 2022.

[3] Pamir, N. Javaid, A. Almogren, M. Adil, M. U. Javed, and M. Zuair, "RFE based feature selection and KNNOR based data balancing for electricity theft detection using BiLSTM-LogitBoost stacking ensemble model," *IEEE Access*, vol. 10, pp. 112948–112963, 2022.

[4] G. Lin, H. Feng, X. Feng, H. Wen, Y. Li, S. Hong, and Z. Ni, "Electricity theft detection in power consumption data based on adaptive tuning recurrent neural network," *Frontiers Energy Res.*, vol. 9, Nov. 2021, Art. no. 773805.

[5] Z. Yan and H. Wen, "Electricity theft detection base on extreme gradient boosting in AMI," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–9, 2021.

[6] T. Hu, Q. Guo, X. Shen, H. Sun, R. Wu, and H. Xi, "Utilizing unlabeled data to detect electricity fraud in AMI: A semisupervised deep learning approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3287–3299, Nov. 2019.

[7] S. O. Tehrani, A. Shahrestani, and M. H. Yaghmaee, "Online electricity theft detection framework for large-scale smart grid data," *Electric Power Syst. Res.*, vol. 208, Jul. 2022, Art. no. 107895.

[8] D. Yao, M. Wen, X. Liang, Z. Fu, K. Zhang, and B. Yang, "Energy theft detection with energy privacy preservation in the smart grid," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7659–7669, Oct. 2019.

[9] D. Gu, Y. Gao, K. Chen, J. Shi, Y. Li, and Y. Cao, "Electricity theft detection in AMI with low false positive rate based on deep learning and evolutionary algorithm," *IEEE Trans. Power Syst.*, vol. 37, no. 6, pp. 4568–4578, Nov. 2022.

[10] Y. Huang and Q. Xu, "Electricity theft detection based on stacked sparse denoising autoencoder," *Int. J. Electr. Power Energy Syst.*, vol. 125, Feb. 2021, Art. no. 106448.

[11] N. Javaid, U. Qasim, A. S. Yahaya, E. H. Alkhammash, and M. Hadjouni, "Non-technical losses detection using autoencoder and bidirectional gated recurrent unit to secure smart grids," *IEEE Access* vol. 10, pp. 56863–56875, 2022.

[12] D. K. Padhi, N. Padhy, A. K. Bhoi, J. Shafi, and M. F. Ijaz, "A fusion framework for forecasting financial market direction using enhanced ensemble models and technical indicators," *Mathematics*, vol. 9, no. 21, p. 2646, Oct. 2021.

[13] S. Hussain, M. W. Mustafa, T. A. Jumani, S. K. Baloch, H. Alotaibi, I. Khan, and A. Khan, "A novel feature engineered-CatBoost-based supervised machine learning framework for electricity theft detection," *Energy Rep.*, vol. 7, pp. 4425–4436, Nov. 2021.

[14] R. Punmiya and S. Choe, "Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing," *IEEE Trans. Smart Grid*, vol. 10, no. 2, pp. 2326–2329, Mar. 2019.

[15] R. K. Ahir and B. Chakraborty, "Pattern-based and context-aware electricity theft detection in smart grid," *Sustain. Energy, Grids Netw.*, vol. 32, Dec. 2022, Art. no. 100833.

[16] G. Lin, X. Feng, W. Guo, X. Cui, S. Liu, W. Jin, Z. Lin, and Y. Ding, "Electricity theft detection based on stacked autoencoder and the undersampling and resampling based random forest algorithm," *IEEE Access*, vol. 9, pp. 124044–124058, 2021.

[17] M. A. de Souza, J. L. R. Pereira, G. D. O. Alves, B. C. de Oliveira, I. D. Melo, and P. A. N. Garcia, "Detection and identification of energy theft in advanced metering infrastructures," *Electric Power Syst. Res.*, vol. 182, May 2020, Art. no. 106258.

[18] M.-M. Buzau, J. Tejedor-Aguilera, P. Cruz-Romero, and A. Gomez-Exposito, "Hybrid deep neural networks for detection of non-technical losses in electricity smart meters," *IEEE Trans. Power Syst.*, vol. 35, no. 2, pp. 1254–1263, Mar. 2020.

[19] *State Grid Corporation of China*. Accessed: Aug. 18, 2022. [Online]. Available: http://www.sgcc.com.cn/html/sgcc_main/col2017011822/column_2017011822_1.shtml

[20] S. Jung, J. Moon, S. Park, S. Rho, S. W. Baik, and E. Hwang, "Bagging ensemble of multilayer perceptrons for missing electricity consumption data imputation," *Sensors*, vol. 20, no. 6, p. 1772, Mar. 2020.

[21] scikit. *Sklearn.impute.SimpleImputer*. Accessed: Aug. 20, 2022. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html

[22] F. Shehzad, N. Javaid, A. Almogren, A. Ahmed, S. M. Gulfam, and A. Radwan, "A robust hybrid deep learning model for detection of non-technical losses to secure smart grids," *IEEE Access*, vol. 9, pp. 128663–128678, 2021.

[23] W. Liao, Z. Yang, K. Liu, B. Zhang, X. Chen, and R. Song, "Electricity theft detection using Euclidean and graph convolutional neural networks," *IEEE Trans. Power Syst.*, early access, Aug. 8, 2022, doi: 10.1109/TPWRS.2022.3196403.

[24] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jun. 2008, pp. 1322–1328.

[25] R. Nian. (Dec. 13, 2019). An Introduction to ADASYN (With Code!). Medium. Accessed: Aug. 21, 2022. [Online]. Available: https://medium.com/@ruinian/an-introduction-to-adasyn-with-code-1383a5ece7aa

[26] I. U. Khan, N. Javeid, C. J. Taylor, K. A. A. Gamage, and X. Ma, "A stacked machine and deep learning-based approach for analysing electricity theft in smart grids," *IEEE Trans. Smart Grid*, vol. 13, no. 2, pp. 1633–1644, Mar. 2022.

[27] *Stacking Classifier Approach for a Multi-Classification Problem.* Accessed: Aug. 28, 2022. [Online]. Available: https://towardsdatascience.com/stacking-classifier-approach-for-a-multi-classification-problem-56f3d5e120c8

[28] *AdaBoost From Scratch—Towardsdatascience.com*. Accessed: Aug. 12, 2022. [Online]. Available: https://towardsdatascience.com/adaboost-from-scratch-37a936da3d50

[29] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, vol. 96, 1996, pp. 148–156.

[30] *XGBoost Algorithm: Long May She Reign!—Towards Data Science.* Accessed: Aug. 12, 2022. [Online]. Available: https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

[31] J. Brownlee. Extreme Gradient Boosting (XGBoost) Ensemble in Python. MachineLearningMastery.com. Accessed: Aug. 12, 2022. [Online]. Available: https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/

[32] P. Mandot. How Exactly XGBoost Works? Medium. Accessed: Aug. 13, 2022. [Online]. Available: https://medium.com/@pushkarmandot/how-exactly-xgboost-works-a320d9b8aeef

[33] S. B. Jabeur, N. Stef, and P. Carmona, "Bankruptcy prediction using the XGBoost algorithm and variable importance feature engineering," *Comput. Econ.*, pp. 1–27, 2022, doi: 10.1007/s10614-021-10227-1.

[34] H. Zhang, D. Qiu, R. Wu, Y. Deng, D. Ji, and T. Li, "Novel framework for image attribute annotation with gene selection XGBoost algorithm and relative attribute model," *Appl. Soft Comput.*, vol. 80, pp. 57–79, Jul. 2019.

[35] J. Hu and X. Ao. (Jun. 8, 2022). Competition Killer: Tree Model and Ensemble Learning. NLP, Chowdera. Accessed: Aug. 14, 2022. [Online]. Available: https://chowdera.com/2022/159/202206080618407739.html

[36] J. Brownlee. (Apr. 26, 2021). Histogram-Based Gradient Boosting Ensembles in Python. MachineLearningMastery.com. Accessed: Aug. 17, 2022. [Online]. Available: https://machinelearningmastery.com/histogram-based-gradient-boosting-ensembles/

[37] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.

[38] S. Adebayo. (Jan. 4, 2021). How CatBoost Algorithm Works in Machine Learning. Dataaspirant. Accessed: Aug. 15, 2022. [Online]. Available: https://dataaspirant.com/catboost-algorithm/

[39] J. Brownlee. (Apr. 26, 2021). Gradient Boosting With Scikit-Learn, XGBoost, LIGHTGBM, and CatBoost. MachineLearningMastery.com. Accessed: Aug. 17, 2022. [Online]. Available: https://machinelearningmastery.com/gradient-boosting-with-scikit-learn-xgboost-lightgbm-and-catboost/

[40] A. Malakhov, F. Goncharov, and E. Gryazina, "Testing machine learning approaches for wind plants power output," in *Proc. Int. Youth Conf. Radio Electron., Electr. Power Eng. (REEPE)*, Mar. 2019, pp. 1–6.

[41] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–9.

[42] J. Brownlee. (Apr. 26, 2021). How to Develop a Light Gradient Boosted Machine (LightGBM) Ensemble. MachineLearningMastery.com. Accessed: Aug. 15, 2022. [Online]. Available: https://machinelearningmastery.com/light-gradient-boosted-machine-lightgbm-ensemble/

[43] A. A. Taha and S. J. Malebary, "An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine," *IEEE Access*, vol. 8, pp. 25579–25587, 2020.

[44] G. M. Messinis and N. D. Hatziargyriou, "Review of non-technical loss detection methods," *Electric Power Syst. Res.*, vol. 158, pp. 250–266, May 2018.

[45] J. Brownlee. (Sep. 15, 2020). ROC Curves and Precision-Recall Curves for Imbalanced Classification. MachineLearningMastery.com. Accessed: Aug. 24, 2022. [Online]. Available: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-imbalanced-classification/

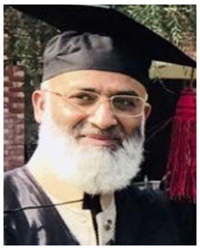
[46] J. Brownlee. (Aug. 30, 2020). Metrics to Evaluate Machine Learning Algorithms in Python. MachineLearningMastery.com. Accessed: Aug. 24, 2022. [Online]. Available: https://machinelearningmastery.com/metrics-evaluate-machine-learning-algorithms-python/

[47] M. Zhang, J. Li, Y. Li, and R. Xu, "Deep learning for short-term voltage stability assessment of power systems," *IEEE Access*, vol. 9, pp. 29711–29718, 2021.

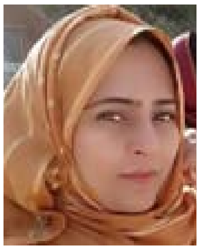
[48] J. Mijalkovic and A. Spognardi, "Reducing the false negative rate in deep learning based network intrusion detection systems," *Algorithms*, vol. 15, no. 8, p. 258, Jul. 2022.

[49] S. Hussain, M. W. Mustafa, K. H. A. Al-Shqeerat, B. A. S. Al-rimy, and F. Saeed, "Electric theft detection in advanced metering infrastructure using Jaya optimized combined Kernel-Tree boosting classifier—A novel sequentially executed supervised machine learning approach," *IET Gener., Transmiss. Distrib.*, vol. 16, no. 6, pp. 1257–1275, Mar. 2022.

PAMIR (Graduate Student Member, IEEE) received the B.S. degree in software engineering from the National University of Modern Languages (NUML), Islamabad, Pakistan, in 2016, and the M.S. degree in software engineering from the Communication Over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad (CUI), Islamabad, under the supervision of Dr. Nadeem Javaid, in 2018, where he is currently pursuing the Ph.D. degree in computer science. He has authored two journals and 12 conference proceedings in international journals and conferences. His research interests include data science, smart grids, and optimal power flow.



NADEEM JAVAID (Senior Member, IEEE) received the bachelor's degree in computer science from Gomal University, Dera Ismail Khan, Pakistan, in 1995, the master's degree in electronics from Quaid-i-Azam University, Islamabad, Pakistan, in 1999, and the Ph.D. degree from the University of Paris-Est, France, in 2010. He is currently a Professor and the Founding Director of the Communications Over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, Islamabad Campus. He has supervised 158 master's and 30 Ph.D. theses. He has authored over 900 articles in technical journals and international conferences. His research interests include energy optimization in smart/micrograids and in wireless sensor networks using data analytics and blockchain. He was a recipient of the Best University Teacher Award (BUTA'16) from the Higher Education Commission (HEC), Pakistan, in 2016, and the Research Productivity Award (RPA'17) from the Pakistan Council for Science and Technology (PCST), in 2017. He is an Associate Editor of IEEE ACCESS and an Editor of *Sustainable Cities and Society*.



MARIAM AKBAR (Senior Member, IEEE) received the M.Sc. degree from the Department of Physics and the M.Phil. degree from the Department of Electronics, Quaid-i-Azam University, Islamabad, Pakistan, in 2001 and 2004, respectively, under the supervision of Dr. Nadeem Javaid, and the Ph.D. degree from COMSATS University Islamabad, Pakistan, in 2016, with the thesis titled "On Network Lifetime Maximization In Wireless Sensor Networks with Sink Mobility." She is currently working as an Assistant Professor with the Department of Computer Science, COMSATS University Islamabad. Her research interests include underwater wireless sensor networks and energy management and blockchain.

ABDULAZIZ ALDEGHEISHEM received the Ph.D. degree in urban planning and spatial information from the University of Illinois at Urbana–Champaign, USA. He worked as the Head of the Department of Urban Planning, in 2012. He worked as an Adviser in a number of government agencies and supervised many projects and specialized studies. He is currently the Dean of the College of Architecture and Planning, King Saud University, and a Professor with the Department of Urban Planning. He also works as an Adviser with the Vision Realization Office (VRO) at the University, and the Supervisor of the Traffic Safety Technologies Chair. His research interests include spatial information in urban planning and management, also he focuses on areas related to city planning, spatial management, and smart city technologies.

NABIL ALRAJEH received the Ph.D. degree in biomedical informatics engineering from Vanderbilt University, USA. He worked as a Senior Advisor with the Ministry of Higher Education, where his role was implementing development programs, including educational affairs, strategic planning, and research and innovation. He is currently a Professor of health informatics with King Saud University. He is a board member of several private universities in Saudi Arabia.

EMAD A. MOHAMMED is currently working as an Assistant Teaching Professor and the Department Chair with Thompson Rivers University, Canada. His research interests include big data, artificial intelligence, machine learning, distributed software systems, and healthcare analytics.