applied sciences

MDPI

Article

ToU Pricing-Based Dynamic Electricity Theft Detection in Smart Grid Using Gradient Boosting Classifier

Rajiv Punmiya and Sangho Choe *

Department of Information, Communications and Electronics Engineering, The Catholic University of Korea, Seoul 14662, Korea; rajiv.punmiya@gmail.com
* Correspondence: schoe@catholic.ac.kr or sangho.choe@gmail.com

Featured Application: Smart Grid (SG); Smart City; Demand Side Management (DSM); Building Energy Management System; Home Energy Management System.

Abstract: In the near future, it is highly expected that smart grid (SG) utilities will replace existing fixed pricing with dynamic pricing, such as time-of-use real-time tariff (ToU). In ToU, the price of electricity varies throughout the whole day based on the respective utilities' decisions. We classify the whole day into two periods with very high and low probabilities of theft activities, termed as the "theft window" and "non-theft window", respectively. A "smart" malicious consumer can adjust his/her theft to mostly targeting the theft window, manipulate actual usage reporting to outsmart existing theft detectors, and achieve the goal of "paying reduced tariff". Simulation results show that existing schemes do not detect well such window-based theft activities conversely exploiting ToU strategies. In this paper, we begin by introducing the core concept of window-based theft cases, which is defined at the basis of ToU pricing as well as consumption usage. A modified extreme gradient boosting (XGBoost) based machine learning (ML) technique called dynamic electricity theft detector (DETD) has been presented to detect a new type of theft cases.

Keywords: AMI smart meter; theft detection; machine learning; XGBoost; time-of-use (ToU) pricing

check for updates

Citation: Punmiya, R.; Choe, S. ToU Pricing-Based Dynamic Electricity Theft Detection in Smart Grid Using Gradient Boosting Classifier. *Appl. Sci.* **2021**, *11*, 401. https://doi.org/10.3390/app11010401

Received: 4 November 2020
Accepted: 30 November 2020
Published: 4 January 2021

1. Introduction

One of the upcoming features of load analytics in the smart grid (SG) is to analyze overall load on the grid and determine dynamic pricing schemes so as to redistribute the consumption load in a balanced way. Using advanced metering infrastructure (AMI) smart meters, utilities are now capable of recording electricity usage on a much more frequent basis (e.g., every 15 min) and enabling all consumers, who previously had bulk usage meters, to be introduced to real-time pricing programs that better reflect differences in the electricity cost. Furthermore, real-time pricing such as time-of-use (ToU) is more favorable than existing fixed pricing from the perspective of efficient power scheduling, demand-side management (DSM), and grid safety, such that existing fixed pricing is expected to be replaced with real-time pricing in the near future [1]. ToU pricing can be either fixed (f-ToU), where the price for each of the broad blocks of hours is predetermined or dynamic (d-ToU), where the low, normal (or medium), and high price periods vary every day. Studies have shown that such a dynamic rate program could greatly affect the electricity usage pattern of consumers [2].

On the other hand, real-time pricing increases the need to detect and prevent malicious consumers that tamper with the smart meter readings to steal electricity, with the primary intention of paying a lower tariff while consuming the same or a higher amount of electricity in the SG. This kind of tampering with the smart meter represents a major problem for utility companies and is a part of non-technical loss (NTL). As we notice in the following literature review, most of the existing schemes consider only energy consumption and do not consider ToU pricing schemes or other external factors for NTL theft detection algorithms. This is the basis of the central idea of our paper to consider the ToU scheme for smarter theft detection. In this paper, we consider only non-technical loss (NTL) theft, which primarily refers to data tampering in the smart meter to minimize the tariff for the malicious user.

The literature review (including survey papers [3–10]) shows that techniques used for detecting theft or abnormal usage activities in the SG take note of the daily historical electricity consumption pattern of that consumer. Machine learning (ML)- or artificial intelligence (AI)-based approaches automate the detection of theft in the SG using various techniques such as decision trees [10], artificial neural networks (ANN) [11,12], support vector machine (SVM) [13], gradient boosting [14,15], clustering [16], deep convolution neural networks [17], and so on.

From the literature review, it is also noticeable that conventional theft detection is mainly based on energy consumption and not the pricing scheme. However, electricity consumption behavior is highly influenced by external factors such as time of use price [2], weather conditions [18], weekday/weekend, and so on. When these factors are considered, time periods with high and low theft probabilities can be estimated. As a result, in order to adapt to the layer of complexity introduced in the modern SG due ToU pricing, the theft cases need to be revised.

Most importantly, in order to study ToU-based theft cases, we need granular datasets. Currently, the dataset used in [17] is one of the first public electricity consumption datasets with realistic labelled theft cases, released by the State Grid Corporation of China (SGCC). However, the sampling rate is one per day, meaning, only the total daily energy usage of the customers has been provided, so the SGCC data are not eligible for theft detection. Fortunately, the ToU-based dataset of the Low Carbon London project solves this problem. However, the only downside is the lack of real theft cases. Therefore, like many other existing studies in the literature, we are limited to experimenting with synthetically generated theft cases in this paper. However, this issue can certainly be addressed in some form in our future work. Mathematically defined theft cases that mimic the attackers' practical intentions were first introduced by the authors in [10] to train their ML-based consumption pattern-based energy theft detector (CPBETD) algorithm. Several existing studies in the literature have followed the same style of theft dataset generation while using ML/AI-based techniques to detect theft. Therefore, currently, modelling synthetic theft cases is necessary if one were to test and evaluate his/her designed ML/AI techniques.

For theft detection, we propose a novel algorithm called the dynamic electricity theft detector (DETD) by adding window-based theft cases (discussed below), where the ToU-based dataset of the Low Carbon London project [19] is used as an example to illustrate our concept. We will use gradient boosting-based classifiers, specifically focusing on extreme gradient boosting (XGBoost) [20] as the base of our theft detector algorithm—DETD; XGBoost has been discussed in more detail in Section 3. The primary reason of choosing XGBoost is that its superiority has been proved in traditional NTL theft detection of SG [14,15] as well as in various data mining competitions of different domains [20]. However, we leave room for improvement in the DETD algorithm as a part of our future work by replacing gradient boosting techniques with other ML/AI techniques. In order to tackle the lack of ToU pricing scheme-based theft detection, this paper begins by discussing how to model realistically feasible synthetic theft cases in the SG, followed by proposing a novel algorithm to detect the theft.

To overcome the shortcomings of the existing schemes, the following are the novel characteristics of DETD, which constitute the contributions of the paper:

- DETD introduces a novel generation of new theft cases based on the ToU pricing model, whereas the majority of existing studies in the literature rely mainly on fixed pricing theft models. Hence, in the DETD algorithm, we use ToU as an example external factor, such that training dataset generation is modified to accommodate the latest ToU pricing-based SG metering.

- Another unique aspect of DETD is the data preprocessing technique by concatenating the usage vector with a window feature vector (explained in Section 4.2).
- Additionally, DETD handles imbalanced data by using the XGBoost [20] hyperparameter "scale_pos_weight" and thus, the overfitting issue in existing ML schemes that is usually caused due to using the Synthetic Minority Oversampling Technique (SMOTE) [21] is easily avoidable. Additionally, DETD reduces memory complexity due to lack of synthetic data generated by SMOTE.
- In the paper, we train and test the DETD algorithm using XGBoost as the base ML scheme for different consumers, report average performance results (detection rate (DR) and false positive rate (FPR)), and evaluate the effect of the testing and training dataset ratio on the performance of the theft detector (Section 4.3).
- In addition, windows based on other factors such as temperature, seasonal trends, and so on can be used to train and create parallel (or combined) theft detectors, which can contribute to a weighted average (weights being the relative numerical significance of a certain factor for theft) towards improved theft detection (whose report is seen in Section 4.5).
- Finally, DETD can harness the use of other alternative tree boosting algorithms (Section 4.6).

Broadly, this paper can be broken into two main blocks—first, the generation of smarter ToU-based synthetic theft cases and second, detection of the theft cases. The detailed organization of the paper is as follows. In Section 2, we examine the need for new theft cases by discussing the new smart grid pricing programs, explaining the dataset, and briefly introducing the motivation of window-based theft cases. We then explain in detail the generation of new synthetic theft cases on the basis of ToU pricing as a primary example along with a minor note regarding the required performance standards for a practical theft detector. Section 3 introduces the gradient boosting ML classifier and its hyperparameter tuning. In Section 4, we introduce the proposed DETD algorithm, report its performance, demonstrate additional theft cases detection, simulate theft windows using other factors, and discuss alternative tree boosting approaches. Section 5 concludes the paper.

2. ToU Pricing Dataset and Synthetic Theft Cases Generation

Electricity theft is the malicious behavior of stealing electrical power from power grids, which is done by tampering with a meter or hacking the pricing data being sent to the utilities. For replication of the act of electricity theft, we use a real consumption dataset. With the help of widely used mathematical models (discussed in Section 2.3), we develop a methodology to model synthetic theft cases that can be used by our ML-based classifier for tackling the smarter ToU pricing scheme theft.

In the following, we first discuss the new dynamic pricing models in the SG, introduce the dataset using those models, explain the concept of window-based theft cases as well as synthetic theft cases generation, and briefly address detection performance metrics.

2.1. New Types of Pricing Models for Smart Grid

Traditionally, consumers can only be billed for the electricity they used via static pricing models such as flat rates or tiered rates [1]. However, in the SG, utilities using smart meters are now capable of using new forms of time-based (or dynamic) pricing models such as [1]:

1. Fixed Time-of-Use pricing (f-ToU): this typically applies to usage over broad blocks of hours (e.g., for a weekday, on-peak = 6 h at afternoon; off-peak = 6 h at night; normal = rest of the hours), where the price for each period is predetermined and constant.
2. Real-Time Pricing (RTP): pricing rates generally apply to usage on an hourly basis.
3. Dynamic Time-of-Use pricing (d-ToU): a hybrid of fixed time-of-use and real-time pricing, where the different periods for pricing are defined in advance, but the price established for the on-peak period varies by utility and market conditions.

2.2. Dataset Metadata

The dataset [19] used in this paper contains electricity consumption (per half hour (unit: kWh)), unique household identifier, and date and time readings for a sample of 1100 consumers of London households that took part in the UK Power Networks-led Low Carbon London project between November 2011 and February 2014. Readings were taken at half hourly intervals, i.e., the sampling rate $f_s = 48$ (samples per day). The consumers were subjected to dynamic Time of Use (d-ToU) electricity prices throughout 2013. The tariff prices were given a day ahead via the Smart Meter IHD (In Home Display) or text message to mobile phones. Consumers were issued High (67.20 pence/kWh), Low (3.99 pence/kWh), or Normal (11.76 pence/kWh) price signals and the times of day these applied. Since each day has 48 readings or time indices, the default feature size is 48.

The consumption data along with d-ToU tariff data make the Low Cardon London dataset a perfect candidate to implement the proposed DETD algorithm. Figure 1 shows a sample energy consumption of a single day for a random consumer in the dataset with corresponding ToU tariff periods (0—low, 1—normal, and 2—high price periods, respectively).

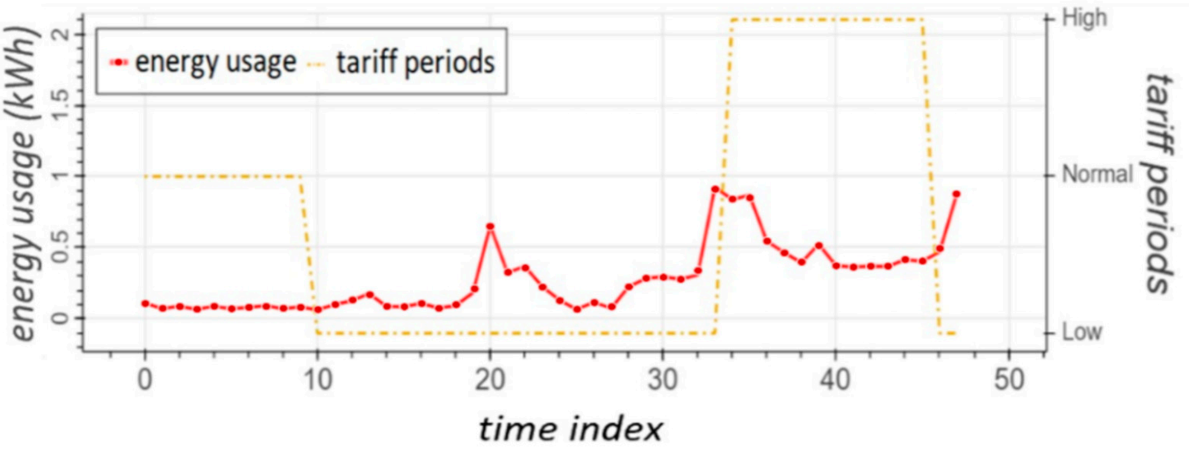


Figure 1. Example of a single day usage from the dataset (sampling interval of 30 min) of a consumer. Dotted line represents corresponding tariff period (0—low, 1—normal, and 2—high price periods, respectively).

2.3. Novel d-ToU and f-ToU Price-Based Theft Cases Generation Using Novel Window-Based Usage Analysis

As seen in the previous literature review, the dataset used in [17] is one of the first public electricity consumption datasets with labelled realistic theft cases, with a sampling rate of one per day: $f_s = 1$. However, in order to study ToU-based theft cases, we need datasets with a higher sampling rate (i.e., f_s is much larger than 1) and labelled theft data. Fortunately, the ToU-based dataset of the Low Carbon London project with $f_s = 48$ solves the first half of this problem. However, there still exists a lack of verified theft cases in this dataset. Therefore, like many other existing studies in the literature, we are limited to experimenting with synthetically generated theft cases in this paper. That is, until a realistic public ToU pricing scheme-based dataset is released by a reputable source that contains verified and labelled theft cases, synthetic theft case generation is necessary for the testing of the proposed ML-based theft detection algorithm (see Section 4) and not easily negligible; it is carefully analyzed in the next section.

In this paper, we assume that from the perspective of the attacker, any day can be broken down into the two "windows", i.e., the "theft period" window indicating time periods where theft could result in maximum profit to the attacker and the "non-theft period" specifying the least profit for the attacker. Hence, the synthetic theft cases proposed in this section are revisions of previous theft cases, taking these windows of variable time durations into consideration. The "windows" could be, of course, randomly chosen time slots or carefully planned time slots to maximize the profits of the attackers.

Based on our previous literature review, in many papers (such as [13,15]) where existing flat pricing is chosen, the probable theft cases have been mathematically modelled to mimic the intentions of the attacker using the entire consumption data of each day. However, in the case of ToU, with the tariff prices given a day ahead via the Smart Meter IHD text message to mobile phones, we assume that a malicious consumer may steal electricity (or manipulate meter readings) only during certain periods with the intentions to minimize his electricity bill and maximize his profits. From the dataset, we find that on average, the high and normal price periods together contribute to approximately 97% of the total tariff, which means the consumer will benefit the most if theft occurs during those periods. Hence, when dynamic pricing is adopted, the theft pattern could be different from static pricing and targeted to highly probable theft periods or simply the "theft period", denoted as x_{tp}. Such a theft strategy may significantly reduce the consumer's chance of being caught by traditional consumption only-based classifiers.

In this paper, for simplicity and the availability of real-world data, we extensively study the two window-based ToU pricing models, i.e., dynamic ToU (d-ToU) and fixed ToU (f-ToU) and apply them to update (or revise) the conventional theft cases. Although the f-ToU and d-ToU pricing systems are simulated in this paper, we can extend the proposed methods to other pricing systems with ease (for example, RTP, etc.).

In the case of d-ToU, the low, normal, and high (or on-peak) price periods are not at the same time every day, while high price and/or low (or off-peak) price periods are absent on some days. Therefore, these windows are variable in size every day. On the other hand, in the case of f-ToU, the on-peak, off-peak, and normal hours are at same time every day such that the theft periods x_{tp} are fixed for each day during the high and normal price periods. Although the prices may vary on weekends vs. weekdays as well as during different seasons, in our presented f-ToU, we assume a low-price period from 9:30 p.m. through to 8:30 a.m., a normal price period from 8:30 a.m. to noon and again from 6 p.m. to 9:30 p.m., and, finally, a peak price period from noon to 6:30 p.m. based on the reference [22].

Referencing the existing literature [10,12], the following would be typically defined theft cases for existing static pricing strategy,

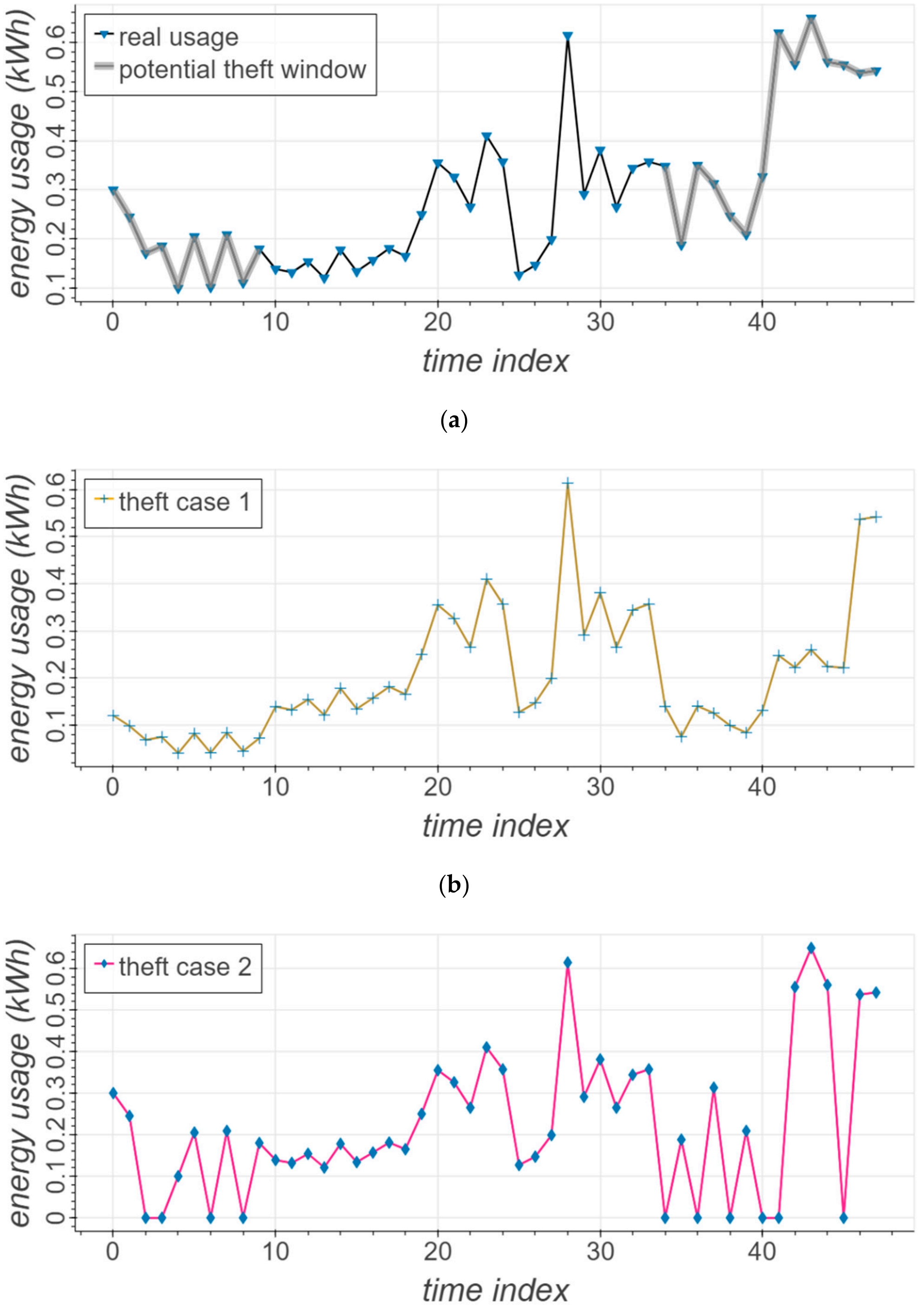
1. $t1(x_t) = \; x_t * random(0.1, 0.9)$;
2. $t2(x_t) = x_t * random[0, 1]$;
3. $t3(x_t) = \; x_t * \, r_t$ ($r_t = random(0.1, 1.0)$);
4. $t4(x_t) = mean(x) * random(0.1, 1.0)$;
5. $t5(x_t) = mean(x)$;
6. $t6(x_t) = \; x_{T-t}$, (where T is the sample size per day, that is, 48);

where x_t is the real usage of the consumer ($t \in [0,\; 47]$ is the time index).

In this paper, new theft cases based on ToU pricing models are derived (modified) from the six above listed theft cases as follows. After carefully observing the six conventional flat price-based theft cases [13,15], for window-based ToU pricing synthetic theft data generation, we eliminate the last three cases, which are trivial in our opinion, and revise the first two cases slightly—cases 1 and 3. As seen above, theft case 1 multiplies each day's usage values with a random number between 0.1 and 0.9, while case 3 simply multiplies usage values with a random value between 0.1 and 1.0 every sampling interval. However, for the window-based cases, the user might sometimes send "0" during the theft period. Therefore, we suggest that multipliers to 0.0 instead of 0.1 are used in both theft cases 1 and 3.

Furthermore, in the ToU pricing models, sending the reverse of corresponding real usage is not feasible during that window, since we do not know the accurate future usage beforehand. That is, especially the 6th theft case, where the consumer reports the reverse of the actual usage for that day, practically has very little significance (the 4th and 5th cases using mean values as well) and no association with window-based theft activities. Hence, we just choose the first three practical and reasonable theft cases from the above list of six cases, for which window-based electricity theft could be considerably active during the theft period—even simulation results regarding all six theft cases are included in Section 4.4 and still show comparable results.

Therefore, we generate the novel three theft cases (based on the above discussion), where theft occurs only during the theft periods, and train our classifier to detect them. Figure 2 shows a regular real usage and various possible malicious/theft usage generated by manipulating the real usage.

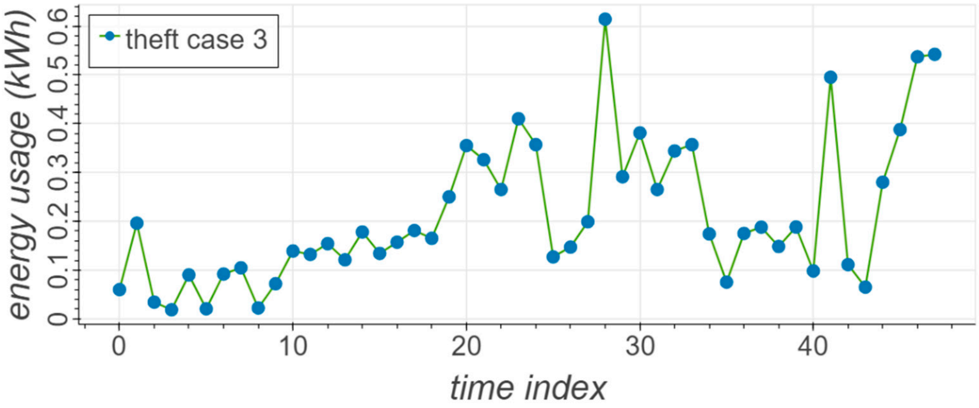


(**a**)

(**b**)

(**c**)

Figure 2. *Cont.*



(**d**)

Figure 2. (**a**) Real usage of an example day with the highlighted "theft period window" showing the most probable time period for theft to occur at. (**b**) Theft case 1 generated from the example day. (**c**) Theft case 2 generated from the example day. (**d**) Theft case 3 generated from the example day.

The following are the window-based theft cases for the ToU pricing scheme:

1. $t1(x_{tp}) = x_{tp} * random(0.0, 0.9)$;
2. $t2(x_{tp}) = x_{tp} * random[0, 1]$;
3. $t3(x_{tp}) = x_{tp} * r_t \ (r_t = random(0.1, 1.0))$;

where x_{tp} or the "theft period" indicates the normal and high price period. The remaining low-price consumption is assumed to stay unaffected by malicious behavior.

In this paper, we are using supervised ML, where artificially generated training theft cases are not limited to the above three cases. In other words, if a new type of suspicious attack is detected through other means by the utilities, additional training data based on that theft pattern can be generated and the presented algorithm can be used to retrain the theft detector to detect such additional cases (see Section 4.4).

2.4. Metrics Used for Theft Detector Performance

The dataset [19] used in this paper contains electricity consumption (per half hour (unit: kWh)) and a unique household identifier. We used detection rate (DR) and false positive rate (FPR) as the standard metric for reporting theft detection. From [13], if "I" stands for intrusion (theft) and "A" for alarm (detection), the following equation defines the probability of intrusion actually occurring, given that we detected some malicious activity:

$$\mathrm{P}(I|A) = \frac{\mathrm{P}(I) \times \mathrm{DR}}{\mathrm{P}(I) \times \mathrm{DR} + \mathrm{P}(\bar{I}) \times \mathrm{FPR}} \tag{1}$$

where P(I) is the probability of intrusion and $\mathrm{P}(\bar{I})$ the probability of no intrusion.

From (1), the lower the FPR, the better the probability is of detecting theft when it occurs. In the case of electricity theft, P(I) is usually a small value, even it varies in different areas. Thus, high DR with extra emphasis on low FPR is preferred for a real-world energy theft detector, meaning if we have two detectors with comparable DRs, we will prefer the one with lower FPR.

3. Extreme Gradient Boosting (XGBoost) Classifier Basics

The following subsections discuss XGBoost [20], a gradient boosting classifier that forms the base classifier of our earlier algorithm Gradient Boosting-based Theft Detector (GBTD) as well as the current proposed scheme "DETD" (see Section 4 for the details), and describe its various hyperparameters that could be manipulated while training.

3.1. Overview of XGBoost Objective Function

Typically, for an ML algorithm, the general objective function is the sum of loss function (L) and regularization term (Ω) over the parameters (θ) as follows:

$$Obj(\theta) = L(\theta) + \Omega(\theta). \tag{2}$$

The XGBoost's objective function (derived from Equation (2)) combines the sum of a specific loss function (l) evaluated over all n predictions (or samples) and the sum of a regularization term (ω) for all predictors (K decision trees) as follows:

$$Obj(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \omega(f_k), \tag{3}$$

where f_k is the kth decision tree function, y_i the actual label of the ith sample, and $\hat{y}_i$ the predicted label of the ith sample. The official XGBoost hyperparameter tuning guide [23] explains the decision tree structure and the objective function in more details.

The goal of the classifier algorithm is to decrease the objective function in (3) as much as possible. Loss function in (3) could be either log-loss function, squared loss function, or others. It controls the prediction error of the ML model, while the regularization term controls its complexity by adjusting the size of the tree structure, depth of the trees, and so on.

In short, XGBoost [20] involves creating and adding trees to the model sequentially. New trees are created to correct the residual errors in the predictions from the existing sequence of trees. The effect is that the model can quickly fit, then start to overfit the training dataset. In general, the phenomenon of overfitting needs to be avoided for biased performance in any ML model. This is usually achieved by tuning the hyperparameters of the ML model during the training and testing phases to achieve a balanced fitting performance. Additionally, the XGBoost classifier is equipped with a "feature_importance" module, which can help us to understand the classifier model deeply by providing a feature score (f-score) of every feature.

3.2. Hyperparameters

Table 1 is a brief explanation of the commonly used hyperparameters of the presented XGBoost-based DETD algorithm and lists their corresponding values to reproduce the results shown in this paper.

Table 1. Hyperparameter values of extreme gradient boosting (XGBoost) used in the dynamic electricity theft detector (DETD).

Hyperparameters	Values	Description
loss function	binary:logistic	Binary classification problem
learning_rate	0.1	Weighting factor for learning (optimizing the objective function) in gradient boosting
n_estimators	100	Number of trees to be generated
max_depth	1	Depth of the trees generated
scale_position_weight	0.33	Parameter to balance negative and positive classes
reg_lambda	1	L2-regularization term on weights of the leaf values
gamma	1	Decides if a node has to be split or not
column_sample_by_tree	1	Portion of columns to be randomly sampled for each successive tree
n_jobs	2	Number of cores to be used for parallel processing computation

The value of "loss function" is set to "binary: logistic", since we are dealing with a binary classification problem. The model can be made more robust by adjusting the "learning rate" parameter, which minimizes the size of each step during every iteration of the algorithm. We set the default value of "learning rate" as 0.1. The default value of "*n_estimators*" is set to 100, which specifies the number of trees to be generated. Details about other parameters can be found in [20] and Table 1. Additionally, in [24], the authors state that a lower FPR with reasonable DR is a good measure of a good intrusion/theft detection system. Another note is that the hypermeter values have been set using guidelines in [23,25] using random search.

4. Proposed DETD Algorithm and Simulation Results

In this section, we first reflect on the performance drawbacks of existing classifiers and then, introduce the DETD algorithm with step-by-step parameter tuning. Finally, we evaluate the proposed novel algorithm, whose performances are reported and compared to existing algorithms.

4.1. Short-Comings of Existing Schemes and Novel Features of DETD

Every day with variation of probable theft/non-theft durations requires the generation of new theft cases based on the concept of "theft window". Existing schemes based on our literature review have shown to have really high theft detection performance for traditional standard theft cases. However, for ToU pricing-based theft cases, we can see via simulation that existing schemes have deteriorated performance.

For example, in our average simulation results for 1000 consumers, Table 2 shows the existing SVM-based CPBETD algorithm [13] has a detection rate of 67% for d-ToU cases and 72% for f-ToU cases, while the XGBoost-based GBTD algorithm [15] has a detection rate of 87% for d-ToU cases and 89% for f-ToU cases. Additionally, the CPBETD algorithm exhibits 34% FPR in d-ToU cases and 29% in f-ToU cases, while the GBTD algorithm shows 11% FPR in d-ToU cases and 8% in f-ToU cases. As a result, existing schemes do not give satisfactory results for those ToU cases. Hence, in order to protect the SG from potentially such "smart" theft, this paper presents another XGBoost-based algorithm called "DETD", which will be explained in the next subsection in detail.

Table 2. Average theft detection performance of SVM-based CPBETD and XGBoost-based GBTD when using the dynamic ToU (d-ToU) and fixed ToU (f-ToU) window-based theft cases for an average of 1000 consumers (Existing Schemes).

CPBETD	**d-ToU**	**f-ToU**
DR [%]	67	72
FPR [%]	34	29
GBTD	**d-ToU**	**f-ToU**
DR [%]	87	89
FPR [%]	11	8

4.2. The Proposed Algorithm

Our proposed algorithm has the following steps which are repeated for every consumer in the SG. Since the performance and complexity of the theft detector vary based on the consumer usage pattern, we report the average performance of the algorithm. In order to replicate theft window-based theft cases, we use the d-ToU and f-ToU pricing data to generate our malicious samples. We assume that the "normal" and "high" price periods are the most likely theft period or "theft window period", while the "low" price period is the least likely theft period or the "non-theft window period".

To train the XGBoost module in the DETD algorithm, we label the generated theft patterns as "1" and corresponding real usage patterns as "0". During the training stage, the theft detector is trained to minimize error between actual labels and predicted labels (“0” or “1”). Figure 3 describes the algorithm in a block diagram.

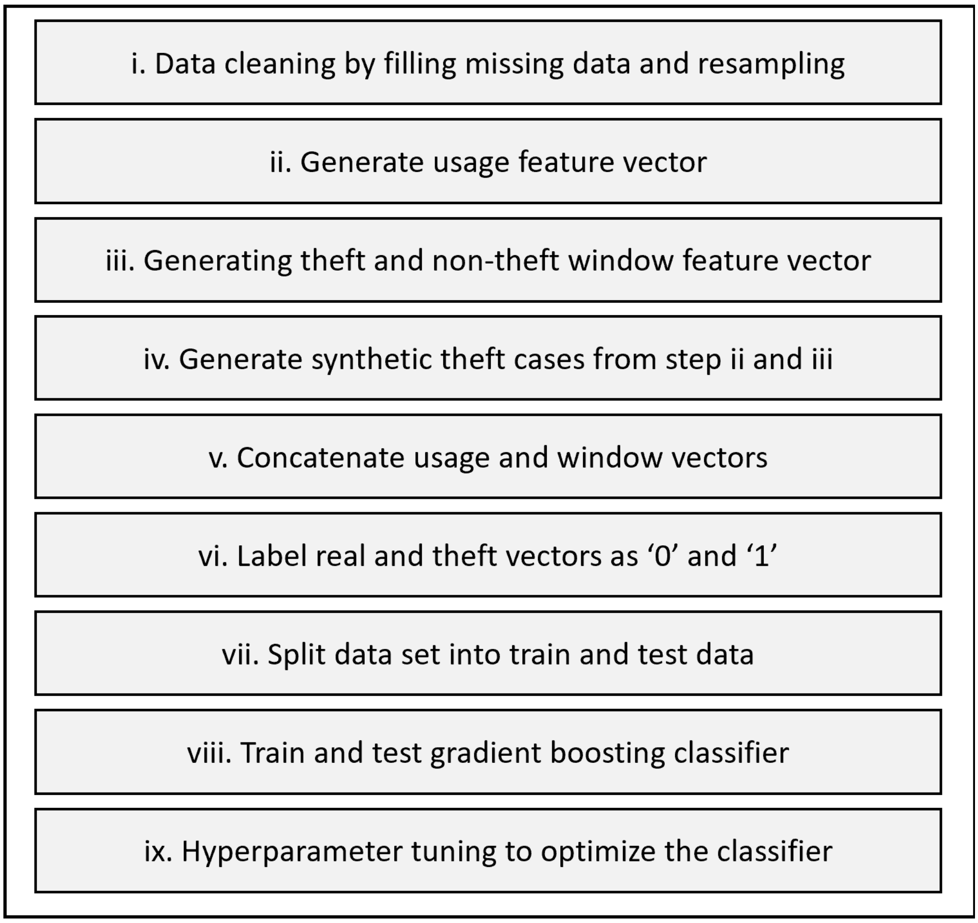

i. Data cleaning by filling missing data and resampling
ii. Generate usage feature vector
iii. Generating theft and non-theft window feature vector
iv. Generate synthetic theft cases from step ii and iii
v. Concatenate usage and window vectors
vi. Label real and theft vectors as ‘0’ and ‘1’
vii. Split data set into train and test data
viii. Train and test gradient boosting classifier
ix. Hyperparameter tuning to optimize the classifier

Figure 3. Schematic diagram of the DETD algorithm.

DETD Algorithm:

i. Dataset cleaning to fill in missing values or adjust sampling using standardized techniques.

ii. Store the usage values in a “usage feature vector” of size 48.

iii. Create the theft window period and non-theft window period feature vector using 0 or 1 as the respective values on the Y-axis and store them in a “window feature vector” of size 48.

iv. Generate synthetic theft cases which are based on the window feature vector corresponding to the real usage vectors.

v. Concatenate the usage feature vector generated in step ii with the window feature vector obtained in step iv to generate a feature vector of size 96.

vi. Label vectors with real-usage and theft-usage cases as “0” and “1”, respectively.

vii. Split the dataset into training and testing data in the ratio of 7:3.

viii. Train and test the binary classifier. Here, we use the XGBoost classifier with the novel training set while passing the hyperparameter “scale_pos_weight”, which is the “number of theft usage data/number of benign (real) usage data”, to scale the imbalance in the two classes.

ix. Use random search for hyperparameter tuning to optimize the XGBoost classifier.

We report the performance for d-ToU using the test set (for an average of 1000 consumers) in Table 3. Then, we repeat the same process (steps i to vii) for f-ToU and report it accordingly. Note that passing the “scale_pos_weight” value eliminates the need for creating synthetic minority class (benign) data using SMOTE [21], such that memory usage is reduced and the classifier is kept from overfitting.

Table 3. Average theft detection performance of DETD when using the d-ToU and f-ToU window-based theft cases for an average of 1000 consumers.

DETD	d-ToU	f-ToU
DR [%]	97.5	98
FPR [%]	4	3

Figure 4 shows the novelty of the DETD algorithm, showcasing the novel 96 feature data processing by concatenating the consumption data with ToU pricing data. Although emphasizing the ToU pricing-based window feature vector in the algorithm, we can substitute (or combine) it with other window vectors, if it is necessary.

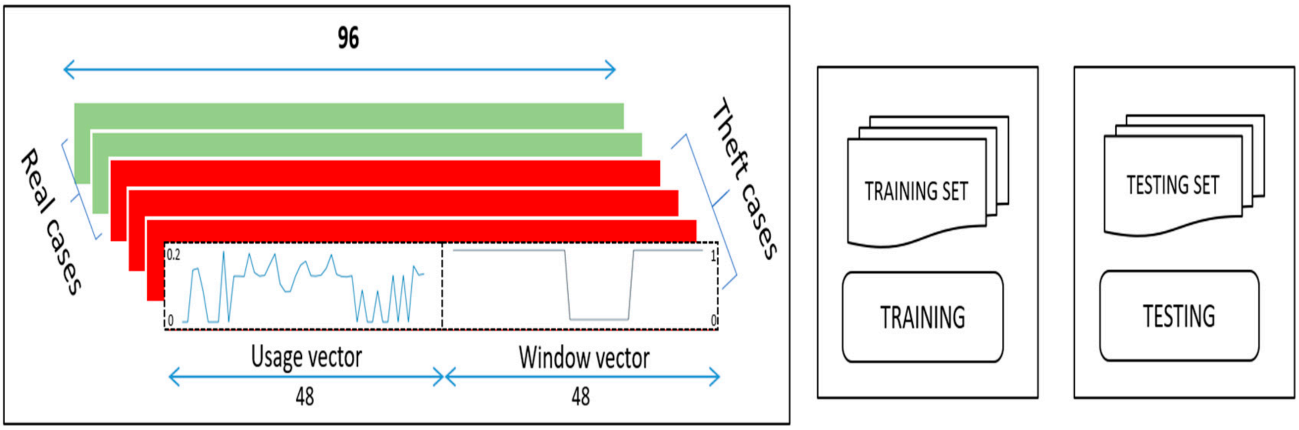


Figure 4. Novel concatenation data processing in the DETD algorithm.

Throughout the paper, all the simulations were performed using Python programming language on an Intel i7 processor, Nvidia RTX 2070 graphic card with a 16 GB RAM system.

4.3. Evaluation of the Proposed Method and Results

In Table 3, we report the average classification performance in terms of detection rate (DR) and false positive rate (FPR). From the simulation results of Table 3, we see that DETD has 97.5% DR and 4% FPR for d-ToU pricing, while having 98% DR and 3% FPR for f-ToU pricing. We confirm that the proposed scheme in Table 3 has an improved performance when compared to existing schemes in Table 2. We see that DETD detects higher theft cases than 87% and 89% of the theft cases detected by "vanilla" GBTD [12] and 67% and 72% of the theft cases detected by CPBETD for d-ToU and f-ToU cases, respectively.

The performance increase can be attributed to having trained with the new window-based theft cases, which are based on not only the energy consumption but also the tariff in contrast with earlier models. CPBETD and "vanilla" GBTD were primarily trained only on consumption, with theft occurring throughout the day. When faced with irregular and smart (i.e., window-based) theft cases, these existing algorithms failed to detect the majority of them because of lack of training.

Lack of historical data is also an important issue that can be addressed using DETD. To study the effect of testing and training set ratio on DETD's performance, we run the DETD algorithm multiple times, starting at 10 percent up to 95 percent of a year's data as training data, with a 5 percent increment at every iteration. Due to the intensive nature of this simulation, we present an average of 50 random consumers as a preliminary report in Figure 5.

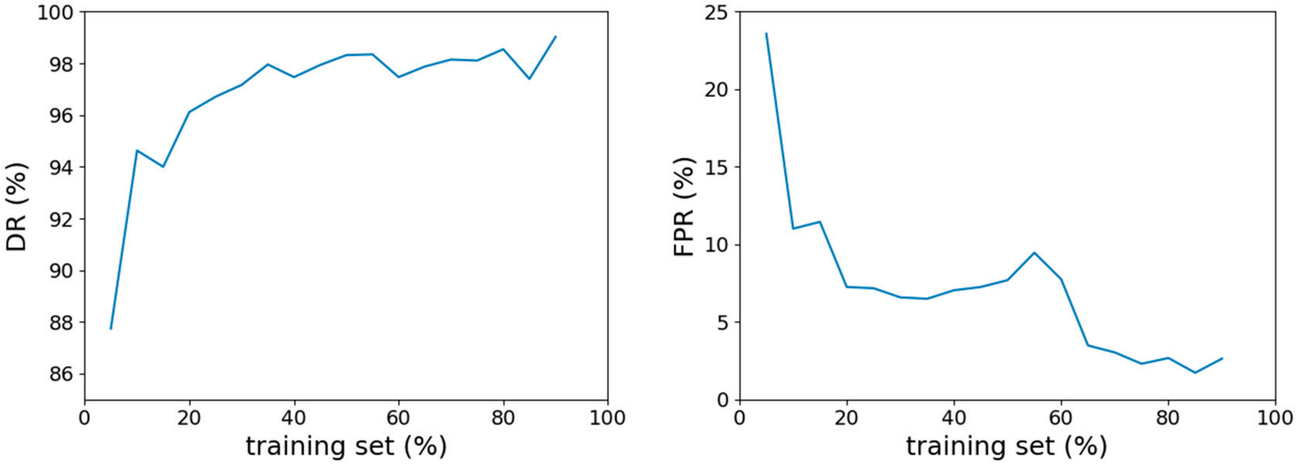


Figure 5. (**Left**) DR performances and (**Right**) FPR performances at different training set ratios for an average of 50 random users.

We observe that with as little as 5 percent of a year's data being used for training, the DR is around 87% and the FPR is around 24%. As expected, the DRs and FPRs gradually improve over the course of increasing training data, with DR being around 99% and FPR being around 3% when 90 percent of the data is used for training.

4.4. Additional Theft Case Detection

Power utility companies could face certain theft cases that are very much specific to a particular smart grid, which may be different (varied or expanded) from proposed mathematical cases. On such an occasion, the utilities may need to retrain (adapt) the presented theft detector with those specific theft cases.

As a simple example, we consider additional ToU theft cases, where the malicious consumer decides to report 0 kWh usage during the high price period.

The following defines such specific window-based theft cases, while the high-price period consumption is set to 0, i.e., $t\left(x_{high\_price}\right) = 0$:

1. $t1\left(x_{np}\right) = x_{np} * random(0.0, 0.9);$
2. $t2\left(x_{np}\right) = x_{np} * random[0, 1];$
3. $t3\left(x_{np}\right) = x_{np} * r_{np} \ (r_{np} = random(0.0, 0.9);$

where x_{np} indicates the normal price period. The remaining low-price consumption is assumed to stay unaffected by malicious behavior.

Via simulation, we generated a separate testing dataset using the above equations for such cases only and found that 98% of those cases were detected without extra need for retraining the entire classifier.

The discarded theft cases 4, 5, and 6 from Section 2.3 were also tested while using DETD without additional retraining and detected with an accuracy of up to 97% in the average sense. This shows that DETD is at par with current ML-based electricity theft detectors performance wise, even for conventional theft cases.

4.5. Theft Detection Using Other Factors Such as Weather Data

The authors of the paper [18] state that electricity supply and demand are becoming increasingly weather-dependent. For a simple demonstration, we assume a hypothetical situation during 3 months of summertime, where the total energy consumption in the grid is high during periods from 12 noon to 4 p.m. (assuming increased AC usage). During this window of high energy demand, the malicious user could report corrupted meter readings with high probability. Thus, as a theft detector trained on weather-dependent theft cases, DETD could be the next logical choice for the utility company. As shown in Table 4, a quick simulation of this simple experiment results in 97.3% DR with 8.1% FPR (training and testing on 3-month summer data). On a side note, this theft window resembles the f-ToU case discussed earlier in Section 2.3. Similarly, additional weather conditions can be combined to determine better theft windows and detectors could be trained accordingly using DETD.

Table 4. Average theft detection performance of DETD when using the weather-dependent theft cases for an average of 1000 consumers (trained and tested on 3-month data).

DETD	3-Month Summer Window
DR [%]	97.3
FPR [%]	8.1

4.6. Using CatBoost and LightGBM as Alternative Approaches

Additionally, instead of XGBoost, we use other alternative boosting algorithms such as LightGBM [26] and CatBoost [27] and report the corresponding results concisely in Table 5.

Table 5. Average theft detection performance of LightGBM- and CatBoost-based DETD when using the d-ToU window-based theft cases for an average of 1000 consumers.

LightGBM	d-ToU	f-ToU
DR [%]	97	97
FPR [%]	7	5
CatBoost	**d-ToU**	**f-ToU**
DR [%]	98	98
FPR [%]	3	2

In general, LightGBM performs the fastest with lower DR and higher FPR, while CatBoost performs slowest with higher DR and lower FPR amongst all the three gradient boosting variants. Additionally, LightGBM and CatBoost also have hyperparameter tuning capabilities to optimize performance based on the requirements.

5. Conclusions

For the upcoming SG utilities adopting the dynamic (d-ToU or f-ToU) pricing program, we have proposed a modified gradient boosting-based theft detector (DETD) that has a good detection capability for highly probable window-based theft cases. In the presented ML procedure, we have broken down the daily samples with two theft windows (“theft window” and “non-theft window”), generated such synthetic theft cases in advance, and trained and tested the DETD based on the usage data of the consumers and their corresponding ToU prices. The simulation results proved that our DETD algorithm that has additional training and provides fine tuning of hyperparameters, which improves the performance of theft detection over existing ML schemes.

While electricity theft is directly dependent on actual power consumption, it can be indirectly affected by other window factors that influence real usage such as the pricing scheme used in the SG, temperature data, seasonal trends, and so on. Taking that into consideration, in our proposed algorithm DETD, we have reported a ToU pricing scheme-based theft activity effect via simulation. As a classifier in our algorithm, alternatives to XGBoost, such as CatBoost and LightGBM, have been tested for their respective performances.

Further work may include creating a global theft detector for the entire grid that detects theft irrespective of the user (one detector for the whole grid instead of one for each consumer) is another important step towards futuristic and practical theft detection. Additionally, automated parameter tuning using Bayesian processes or a better replacement of gradient boosting techniques with other ML/AI techniques can further help in optimizing theft detection using DETD.

Author Contributions: Conceptualization, R.P. and S.C.; methodology, R.P.; software, R.P.; validation, R.P. and S.C.; formal analysis, R.P.; investigation, R.P. and S.C.; resources, R.P. and S.C.; data curation, R.P.; writing—original draft preparation, R.P.; writing—review and editing, S.C.; visualization, R.P.; supervision, S.C.; project administration, S.C.; funding acquisition, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the NRF grant funded by the Korea Government Ministry of Science and ICT (MSIT) under Grant 2017R1A2B4005840, and in part by the Catholic University of Korea, Research Fund, 2020.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. US Department of Energy Official Website. Available online: https://www.smartgrid.gov/recovery_act/time_based_rate_programs.html (accessed on 3 November 2020).
2. Wolak, F. Do residential consumers respond to hourly prices? Evidence from a dynamic pricing experiment. *Am. Econ. Rev.* **2011**, *101*, 83–87.
3. Ghori, K.M.; Imran, M.; Nawaz, A.; Abbasi, R.A.; Ullah, A.; Szathmary, L. Performance Analysis of Different Types of Machine Learning Classifiers for Non-Technical Loss Detection. *IEEE Access* **2020**, *8*, 16033–16048. [CrossRef]
4. Musleh, S.A.; Chen, G.; Dong, Y.Z. A Survey on the Detection Algorithms for False Data Injection Attacks in Smart Grids. *IEEE Trans. Smart Grid* **2019**, *11*, 2218–2234. [CrossRef]
5. Jaime, Y.; Tang, B. Detection of Electricity Theft in Consumer Consumption Using Outlier Detection Algorithms. In Proceedings of the 2018 1st International Conference on Data Intelligence and Security (ICDIS), South Padre Island, TX, USA, 8–10 April 2018; pp. 135–140.
6. Glauner, P.; Meira, J.A.; Valtchev, P.; State, R.; Bettinger, F. The Challenge of Non-Technical Loss Detection Using Artificial Intelligence: A Survey. *Int. J. Comput. Intell. Syst.* **2017**, *10*, 760–775. [CrossRef]
7. Messinis, G.M.; Hatziargyriou, N.D. Review of non-technical loss detection methods. *Electr. Power Syst. Res.* **2018**, *158*, 250–266. [CrossRef]
8. Ahmad, T.; Chen, H.; Wang, J.; Guo, Y. Review of various modeling techniques for the detection of electricity theft in smart grid environment. *Renew. Sustain. Energy Rev.* **2018**, *82*, 2916–2933. [CrossRef]
9. Viegas, J.L.; Esteves, P.R.; Melício, R.; Mendes, V.; Vieira, S.M. Solutions for detection of non-technical losses in the electricity grid: A review. *Renew. Sustain. Energy Rev.* **2017**, *80*, 1256–1268. [CrossRef]
10. Fragkioudaki, A.; Cruz-Romero, P.; Gómez-Expósito, A.; Biscarri, J.; De Tellechea, M.J.; Ángel, A. Detection of non-technical losses in smart distribution networks: A review. In *Trends in Practical Applications of Scalable Multi-Agent Systems, the PAAMS Collection*; Springer: Cham, Switzerland, 2016; pp. 43–54.
11. Adil, M.; Javaid, N.; Qasim, U.; Ullah, I.; Shafiq, M.; Choi, J.-G. LSTM and Bat-Based RUSBoost Approach for Electricity Theft Detection. *Appl. Sci.* **2020**, *10*, 4378. [CrossRef]
12. Buzau, M.-M.; Tejedor-Aguilera, J.; Cruz-Romero, P.; Gomez-Exposito, A. Hybrid Deep Neural Networks for Detection of Non-Technical Losses in Electricity Smart Meters. *IEEE Trans. Power Syst.* **2020**, *35*, 1254–1263. [CrossRef]
13. Jokar, P.; Arianpoo, Á.; Leung, M.C.V. Electricity theft detection in AMI using consumers' consumption patterns. *IEEE Trans. Smart Grid* **2016**, *7*, 216–226. [CrossRef]
14. Buzau, M.-M.; Tejedor-Aguilera, J.; Cruz-Romero, P.; Gomez-Exposito, A. Detection of Non-Technical Losses Using Smart Meter Data and Supervised Learning. *IEEE Trans. Smart Grid* **2019**, *10*, 2661–2670. [CrossRef]
15. Punmiya, R.; Choe, S. Energy Theft Detection Using Gradient Boosting Theft Detector with Feature Engineering-Based Preprocessing. *IEEE Trans. Smart Grid* **2019**, *10*, 2326–2329. [CrossRef]
16. Maamar, A.; Benahmed, K. A Hybrid Model for Anomalies Detection in AMI System Combining K-means Clustering and Deep Neural Network. *Comput. Mater. Contin.* **2019**, *60*, 15–39. [CrossRef]
17. Zheng, Z.; Yang, Y.; Niu, X.; Dai, H.-N.; Zhou, Y. Wide and Deep Convolutional Neural Networks for Electricity-Theft Detection to Secure Smart Grids. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1606–1615. [CrossRef]
18. Zhang, C.; Liao, H.; Mi, Z. Climate impacts: Temperature and electricity consumption. *Nat. Hazards* **2019**, *99*, 1259–1275. [CrossRef]
19. Smart Meter Energy Consumption Data in London Households. Available online: https://data.london.gov.uk/dataset/smartmeter-energy-use-data-in-london-households (accessed on 3 November 2020).
20. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016. [CrossRef]
21. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]

22. Fixed ToU Price Reference: Energy Sage. Available online: https://news.energysage.com/understanding-time-of-use-rates/ (accessed on 3 November 2020).
23. Hyperparameter Tuning in XGBoost. Available online: https://xgboost.readthedocs.io/en/latest/parameter.html (accessed on 3 November 2020).
24. Axelsson, S. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.* **2000**, *3*, 186–205. [CrossRef]
25. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
26. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.-I. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017.
27. Dorogush, A.; Ershov, V.; Gulin, A. CatBoost: Gradient boosting with categorical features support. *arXiv* **2018**, arXiv:1810.11363.