2023 5th International Conference on Energy, Power and Environment: Towards Flexible Green Energy Technologies (ICEPE) | 979-8-3503-1312-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICEPE57949.2023.10201607

Multi-layer Model Classifier for Cyberattack detection in Smart Electric Grid

Sourabh Singh[a], Student Member, IEEE and Ch. Venkaiah[b], Senior Member, IEEE
Department of Electrical Engineering National Institute of Technology Warangal, Hanamkonda, Telangana, India
[a]*sourabhsingh@ieee.org*, [b]*ch.venkaiah@ieee.org*

***Abstract*—In the Smart Grid, communication lines and physical open access points are always prone to cyber-attacks, and electric theft is the most common one. To detect electricity theft, researchers have developed several advanced machine learning models. However, existing work has not explored the problem of data imbalance properly, which is one of the significant challenges in electricity consumption data. This paper aims to compare various data balancing techniques and present an integrated theft detection model.**

This paper presents a multi-layer model for detecting fraudulent consumers in the smart grid. The detection process starts with data preparation steps, which include data interpolation, outlier handling, and data standardization. The next crucial step is handling data imbalance. Various techniques are tested, and AdaSys performs better than others. The model is being trained on a balanced dataset and validated on a real imbalanced dataset for realistic results. For higher performance, a two-layer model is chosen for electricity theft detection. The first layer consists of three heterogeneous machine learning models, and an Artificial Neural Network (ANN) model is used for the second layer. The first layer's probabilistic prediction serves as input to the second layer, which makes the final prediction. Experimental results confirm that multilayer model classifiers perform better than individual classifiers for detecting cyber-attacks on real consumption datasets.

***Index Terms*—Electricity fraud detection, Big data, Preprocessing, Imbalance Handling, Multilayer Model, smart grid.**

I. Introduction

Smart grids have revolutionized the power sector with their ability to integrate and manage distributed energy resources, improve system reliability, and reduce greenhouse gas emissions. However, the smart grid also faces several challenges, including losses in the energy supply chain. These losses are classified into two types: technical losses and non-technical losses (NTLs).

Technical losses refer to the losses that occur during the transmission and distribution of electricity due to the inherent physical properties of the power grid. These losses are inevitable and result from factors such as resistance, inductance, and capacitance of the transmission lines, transformers, and other power system components. NTLs, on the other hand, are the losses that occur due to theft, metering errors, meter bypassing and tampering, synchronously switching power circuits, and tapping on secondary voltages [1]. Compared to physical attacks, cyber attacks are more covert and more flexible, and hence they are more popular in smart grids [2]. These losses not only result in revenue loss for utility companies but also affect the reliability and stability of the power grid [3]. In recent years, the focus has shifted towards reducing NTLs as they have become a significant problem for utility companies worldwide. The International Energy Agency (IEA) estimates that NTLs account for approximately 10% of the total electricity distributed globally, resulting in annual revenue losses of around $100 billion [4].

The identification of NTL through supervised methods encounters three crucial obstacles, which are effectively dealing with missing and outlier data points in the data pre-processing phase, handling the class imbalance issue in the data, and appropriately selecting a classifier that fits the problem.

To create a robust fraudulent consumer detection model, the first step is to perform thorough data preprocessing. Effective data preprocessing is crucial for enhancing the performance of the final model. According to a survey of 34 machine learning models for electric theft detection, only 50% of the research articles investigated the issue of data preprocessing [3]. This paper address above problem using an efficient data preprocessing sequence.

Handling data imbalance is crucial in achieving unbiased electricity theft detection. Data imbalance is a common issue in electricity theft detection, where the number of legitimate electricity consumers is much larger than the number of electricity thieves. The resulting imbalanced data set can negatively affect the accuracy and performance of the electricity theft detection model, as most algorithms are designed to work best with balanced data sets. Therefore, handling data imbalance is essential to improve the accuracy and performance of electricity theft detection models. Several approaches have been proposed in the literature [5]- [7] to handle data imbalance in electricity theft detection. These approaches include data sampling, cost-sensitive learning, and ensemble techniques. Data sampling techniques involve balancing the data set by either undersampling the majority class or oversampling the minority class. Cost-sensitive learning involves assigning different costs to misclassification errors for different classes. Ensemble methods involve combining multiple models to achieve better performance. In this paper, various data sampling and ensemble technique are compared and a better data imbalance handling sequence is preposed for realistic results. The next challenge is to select appropriate machine learning and deep learning algorithm for our multilayer model. In this paper a combination of heterogeneous algorithm which

979-8-3503-1312-3/23/$31.00 ©2023 IEEE

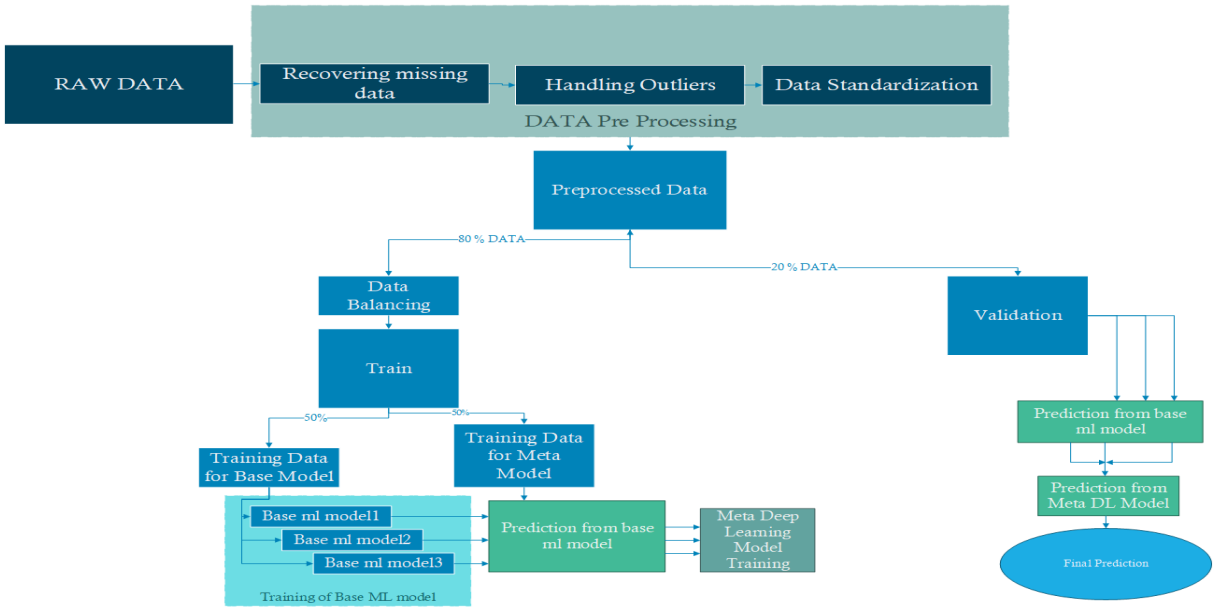

Fig. 1. Proposed framework for electricity fraudulent consumer detection Classifier

works differently form each other and learn different feature from the data set are selected for first layer and an Artificial neural network (ANN) is used in second layer.

The final and crucial step in classifier model development is testing, as it demonstrates the model's reliability. Therefore, it should be carried out appropriately. In this paper, the testing of classifier model is performed on a real raw dataset, in contrast to the approach taken in [8], where synthetic data generated by an imbalance handler were included in the test dataset. The inclusion of synthetic data may lead to falsely inflated results, and the classifier model can easily detect this data.

The paper presents several significant contributions, which are as follows:(i) Formulating a multilayer classifier model that outperforms individual ML classifier models in detecting electricity fraudulent consumers. (ii) Conducting an extensive exploration of data imbalance handling techniques and comparing their impact on classifier model performance. (iii) Implementing an improved data processing sequence to achieve more realistic and accurate results.

The rest of the paper discusses various topics related to the classifier model. Section II explains the overall flow of the classifier model. Section III demonstrates the data preprocessing techniques that are used in the model. In Section IV, methods for handling data imbalance are presented. Section V implements the first and second layers of the multilayer classifier model. The results are then evaluated and compared in Section VI. Finally, Section VII provides the conclusion and summarizes the key findings of the study.

II. Model Overview

To improve the effectiveness of the electric theft detection classifier model, it is crucial to accurately identify the minority class (fraudulent consumers). This requires implementing an appropriate data processing sequence and selecting the most effective data balancing technique available.

The process of detecting electric fraudulent consumers starts with data preparation, which involves handling missing values, outliers, and normalizing the data. After data preparation, the data is split into training and validation sets, with 80% of the data used for training and 20% for validation, as shown in Figure 1. To ensure that the model is trained on a balanced dataset, a data balancing technique is applied to the training data. The balanced training dataset is then split into two parts for training the first and second layers of the classifier model. Three heterogeneous machine learning models, namely SVM, KNN, and RF, are used for the first layer, while an ANN model is selected for the second layer due to its simplicity and effectiveness. Finally, the performance of the multilayer model is evaluated using the validation dataset.

III. Data Preparation

A. Recovering Missing Data

It is important to handle missing values present in the time series dataset. Otherwise it will badly impact the training of our model. A better way of tackling the missing value is to replace them with the mean of their neighbour value. Fill the missing value using the interpolation method in [9], as follows,

$$f(x_i) = \begin{cases} \frac{x_{i-1}+x_{i+1}}{2} & x_i \in \text{NaN}, x_{i-1}, x_{i+1} \notin \text{NaN} \\ x_i & x_i \notin \text{NaN}, \end{cases} \quad (1)$$

where x_i represent missing (null) value. The non-numeric Character NaN is the representation of null value. For the consumer having less than 7 missing values, equation (1) is used for filling that value, otherwise if consumer data has a missing value between 7 and 200 then the NaN will be

replaced by zero. The consumer that has more than 200 NaN values will be removed from the dataset.

B. Handling outliers

The presence of an outlier in the training dataset may confuse model and leads to longer computation time. This results in deterioration and performance of the model. the problem of outliers will be solved by the "two sigma rule of thumb" which is similar to the outlier handler as reported in [9]

$$f(x_i) = \begin{cases} \text{avg}(\mathbf{x}) + 2 \cdot \text{std}(\mathbf{x}) & \text{if } x_i > \text{avg}(\mathbf{x}) + 2 \cdot \text{std}(\mathbf{x}), \\ x_i & \text{otherwise,} \end{cases} \tag{2}$$

where x indicates electricity consumption consumer.

C. Data Normalization

Data normalization is used to scale the data so that it has a common scale, which helps the machine learning model converge faster and perform better. It is also used to handle outliers and reduce the impact of large values on the model. Additionally, normalization helps to prevent certain types of models from becoming sensitive to the scale of the input data. Overall, it improves the accuracy and stability of the model.

Data normalization can be done using Min-Max method which can be repersented by equation (3).

$$f(x_i) = \frac{X_i - min(X)}{max(X) - min(X)}. \tag{3}$$

IV. Imbalance Handling

Data imbalance is a common problem in electricity consumption data, and it can cause the classifier to become skewed toward the majority class. To address this issue, several techniques, such as oversampling, undersampling, and a combination of both [10], can be used to balance the dataset and enhance the performance of the classifier models. In this paper, we compare various sampling techniques to determine the optimal approach for handling data imbalance.

A. SMOTE

Synthetic Minority Oversampling Technique (SMOTE) [11] addresses this issue by artificially creating samples for the minority class. The algorithm selects a minority class sample at random and then selects one of its k-nearest neighbors. It then generates a new synthetic sample at a random point between the selected sample and its neighbor. This process is repeated until the minority class is balanced with the majority class.

SMOTE helps in improving the performance of Classifier models on minority classes by increasing the diversity of the training data. However, it can also create unrealistic synthetic samples that do not accurately represent the minority class. Additionally, it does not take into account the distribution of the majority class, which can lead to oversampling of specific regions of feature space.

While SMOTE is a useful technique for handling imbalanced datasets, it should be used with caution and in conjunction with other techniques. It's important to evaluate the results of the model on a validation set to ensure that it is not overfitting and to evaluate the quality of the synthetic samples generated by the algorithm.

B. Tomek link undersampling

Tomek link [12] undersampling is a technique based on removing samples from the majority class that is closest to samples from the minority class. The algorithm works by identifying pairs of samples, one from the majority class and one from the minority class, that are closest to each other. These pairs are called Tomek links.

The algorithm then removes the majority class sample from the pair, leaving only the minority class sample. This process is repeated until all Tomek links are removed.

C. Edited Nearest Neighbors

Edited Nearest Neighbours (ENN) [12] is a technique based on removing samples from the majority class that is misclassified by the nearest neighbor classifier.

The algorithm works by identifying samples from the majority class that is misclassified by their k nearest neighbors. These samples are then removed from the dataset. This process is repeated until all misclassified samples are removed.

D. KMeansSMOTE

KMeansSMOTE [11] is a technique based on creating synthetic samples for the minority class using K-means clustering.

The algorithm works by first applying K-means clustering to the minority class samples, creating K clusters. Next, synthetic samples are created for each cluster by interpolating between the cluster centroid and a randomly chosen minority class sample from that cluster. This process is repeated until the desired number of synthetic samples is created. This method aims to create synthetic samples that are similar to the real minority class samples, and that is representative of the minority class distribution. By increasing the number of minority class samples, the algorithm aims to improve the balance of the dataset.

E. NearMiss

The NearMiss(NM) [12] algorithm works by selecting a subset of the majority class, such that the selected samples are the closest in distance to the minority class samples. This helps to balance the class distribution and prevents the model from being biased toward the majority class. The approach is based on the nearest neighbor method. The algorithm goes through each sample in the minority class and selects samples from the majority class that is closest in distance. The selected samples are then removed from the dataset, resulting in a more balanced distribution of classes. This technique is particularly useful for datasets where the minority class is rare and the majority class is very large.

F. ADAsys

ADASYS (Adaptive Synthetic Sampling) [11] is a technique based on creating synthetic samples for the minority class using adaptive synthetic sampling.

The algorithm works by first building a classifier using the majority class samples. Next, it applies this classifier to the minority class samples and classifies them as either safe or border samples. Safe samples are samples that are correctly classified by the classifier, while border samples are samples that are misclassified. Synthetic samples are then created by interpolating between two randomly chosen border samples. This process is repeated until the desired number of synthetic samples is created. This method aims to create synthetic samples that are similar to the border samples, and that is representative of the minority class distribution. By increasing the number of minority class samples, the algorithm aims to improve the balance of the dataset. Additionally, it is also important to note that ADASYS attempts to take into account the characteristics of the majority class in creating the synthetic samples, which can be beneficial for the overall performance of the model.

V. CLASSIFIER MODEL

After performing data preparation and addressing the imbalance, the data was cleaned, formatted, and transformed to train the classifier. Among various state-of-the-art methods, multilayer machine learning and deep learning models are considered one of the best classifiers, having recently won many Netflix and Kaggle competitions for classification tasks [13]. This approach involves learning high-level classifiers (second layer) on top of the base classifier (first layer) to achieve high classification accuracy. In this study, three different machine learning models were stacked on top of the deep learning model as a classifier.

A. First Layer of Machine learning Model

Support Vector Machines (SVMs) [14] are a type of supervised learning algorithm that can be used for classification or regression tasks. In this paper, SVM is used for regression. The algorithm finds the best boundary (or "hyperplane") that separates the different classes in the data. Points closest to the boundary are called support vectors and have the greatest impact on the position of the boundary. The goal is to choose a boundary that maximizes the margin, which is the distance between the boundary and the closest data points from each class.

Random Forest (RF) [14] is a machine learning algorithm that builds multiple decision trees, known as forests, and then averages their predictions. It uses a technique called bagging to create multiple subsets of the data and then trains a decision tree on each subset. This results in a diverse set of trees that reduces overfitting. During the prediction phase, each tree in the forest makes a prediction, and the final output is the average of all predictions. This allows the model to capture complex relationships in the data, making it a powerful technique for classification and regression tasks.

K-Nearest Neighbors (KNN) [14] regression is a type of machine learning algorithm that is used for predicting continuous values. It works by finding the k-nearest neighbors of a given point and using their values to make a prediction. The algorithm first finds the k-nearest data points to the given point and then calculates the average of the values of those points. This average value is used as the prediction for the given point. KNN regression is considered a non-parametric method as it does not make any assumptions about the underlying distribution of the data. This makes it a useful tool for dealing with complex and non-linear relationships in data.

B. Second Layer of Artificial Neural Network

A higher-level model in a stacked machine learning or deep learning model is a model that is trained to make predictions using the outputs of one or more lower-level models. The lower-level models, also known as base models, are trained on the input data first, and their outputs are then used as input to the higher-level model. The higher-level model is trained to learn the relationship between the outputs of the base models and the target variable. This allows the higher-level model to make predictions using the combined knowledge of all the base models. The idea behind using a higher-level model in a stacked model that can potentially improve the performance of the final prediction by taking into account the outputs of multiple base models. As a result, it can make more accurate predictions than a single base model by combining the predictions from different models.

ANN is selected for this layer because of its ability to model complex interactions between features. Even with a small number of features, ANNs can learn complex relationships and patterns that may not be captured by traditional linear models. The ANN used in this paper has 5 hidden layers with RELU activation function and an Output layer with sigmoid activation function as shown in Fig. 2.

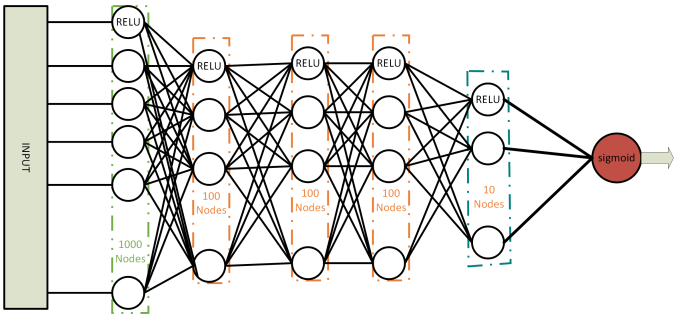

Fig. 2. Second Layer Moder (ANN)

VI. EXPERIMENTS AND RESULTS

The experiments in this study were conducted using Google Colab [15], which is an open-source platform provided by Google. In order to identify fraudulent consumers in the smart grid, we developed the proposed model using Python 3.7 [18], as described in Section II. A two-layer model was used for this purpose. In the first layer, three machine learning models were

used to process the input data. The predictions of this base model were then used as input for the second layer, which consisted of a deep learning model for the final prediction. The realistic consumption data of smart grid consumers was obtained from SGCC [16]. The dataset used in this study included the consumption data of 42,372 users for 1,035 days (2014-2016). Of these users, 38,757 were honest consumers, while 3,615 were dishonest or fraudulent consumers.

A. Performance of different imbalanced handling method

The dataset analysis in Figure 3 shows that honest consumers outnumber dishonest ones. However, training a model with imbalanced data may result in a skewed model towards the majority class, leading to inaccurate results favoring honest consumers. Therefore, addressing the issue of data imbalance is crucial for effectively training an unbiased model that can successfully detect fraudulent behavior among the minority class. This will ensure that the model achieves its objective and provides reliable results.

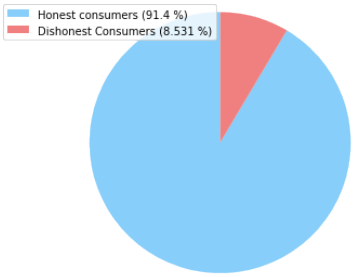

Fig. 3. Proportion of Honest and Dishonest consumer in dataset

Various combinations [17] of oversampling and undersampling techniques have been utilized to address the issue of imbalanced classes in the given dataset. These combinations have been tested on several machine learning algorithms, including Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), Logistic Regression (LR), and K-Nearest Neighbors (KNN). Figure 4 displays a comparison of the results obtained by selecting the optimal performance of each sampling algorithm. The bar chart highlights the superiority of ADASyn among the sampling techniques, indicating its better performance in mitigating the class imbalance problem.

B. Selection of Epoch for training of Model

The number of epochs is a crucial factor that can significantly affect the performance of a model. If the number of epochs is too small, the model may not have sufficient time to learn the underlying consumption patterns in the data, leading to underfitting. Conversely, if the number of epochs is too high, the model may memorize the training data, resulting in overfitting.

The proposed model was trained for 15 epochs, as depicted in Figure 5. It was observed that the training accuracy of the model stabilized at its highest value, and the loss reached its minimum level. Based on this observation, the number of

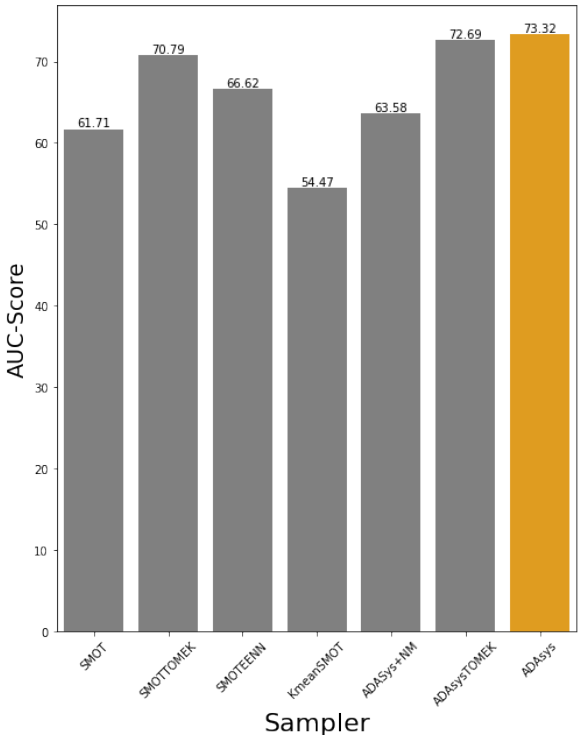

Fig. 4. Comparison of sampling method

epochs for the final model was chosen as 15, to strike a balance between underfitting and overfitting.

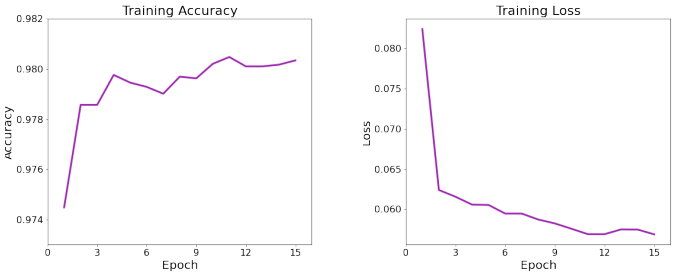

Fig. 5. Training Performance of Second layer model (ANN)

C. Selection of Threshold for meta model

The threshold is a critical parameter that determines whether a given input belongs to one class or another, based on the output of the ANN. If the output of the network exceeds the threshold, the input is classified as belonging to the Theft consumer class, while if it is below the threshold, it is classified as belonging to the Honest consumer class. Setting the threshold too high results in a high number of false negatives, whereas setting it too low results in a high number of false positives.

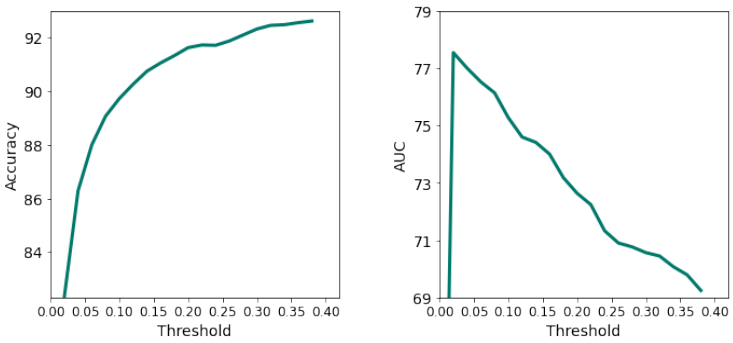

Fig. 6. Improving Classifier Model Predictions through Threshold Selection

After analyzing Figure 6, the optimal threshold value was determined to be 0.1. At this point, both accuracy and AUC-score were found to be at their maximum values. This approach ensures that the threshold value is selected to provide the highest possible accuracy and AUC-score, balancing the trade-off between false negatives and false positives.

D. Multilayer model comparison with individual model

This section presents a comparison between the Multilayer model proposed in this study and an existing benchmark model for theft detection. In the Multilayer model, three heterogeneous ML models (SVM, RF, and KNN) in the base layer and ANN in the meta layer are employed. Accuracy and AUC-score as the evaluation metrics for comparison are utilised. Results indicate that the Multilayer model outperformed the existing benchmark method in both accuracy and AUC-score. These findings demonstrate that the effectiveness of the proposed Multilayer model for theft detection serves as a reliable tool for fraud detection in the energy sector.

TABLE I
COMPARISON OF MULTILAYER MODEL WITH INDIVIDUAL MODEL

Classifer	**Performance Matrix**	
	Accuracy	***AUC Score***
Decision Tree	85.43	60.7
Random Forest	74.72	64.54
ANN	88.79	72.09
Support Vector Machine	91	73.37
Multilayer Model	91.03	74.69

VII. CONCLUSIONS

This paper presents a comprehensive comparison between the Multilayer model and individual machine learning models for detecting electricity theft using a real dataset from the State Grid Corporation of China (SGCC), one of China's largest power utilities. To enhance the model's efficiency, a data preparation sequence was implemented, including interpolation for missing values, two sigma rules for outlier handling, and data standardization to prepare the data for model training. The paper also addresses the issue of data imbalance, which is a significant problem in electricity fraud detection, and tests various imbalance handling techniques to find that ADAsys performs the best. Unlike many other papers that use synthetic data in their testing, only real data is used in this study to achieve more realistic results. The proposed Multilayer model consists of two layers, with the first layer generating data for the second layer, which makes the final prediction. The model achieved an accuracy of 91.03 and an AUC-score of 74.69 on real data, outperforming individual models.

In future, to increase the robustness of the Multilayer model, it can be tested on synthetically generated attacks in the smart grid. This will enable the model to handle unseen data and improve its ability to detect electricity theft.

REFERENCES

[1] P. P. Biswas, H. Cai, B. Zhou, B. Chen, D. Mashima and V. W. Zheng, "Electricity Theft Pinpointing Through Correlation Analysis of Master and Individual Meter Readings," IEEE Transactions on Smart Grid, vol. 11, no. 4, pp. 3031-3042, July 2020, doi: 10.1109/TSG.2019.2961136.

[2] X. Xia, Y. Xiao, W. Liang and J. Cui, "Detection Methods in Smart Meters for Electricity Thefts: A Survey," Proc. IEEE, vol. 110, no. 2, pp. 273-319, Feb. 2022, doi: 10.1109/JPROC.2021.3139754.

[3] A. M. Tureczek and P. S. Nielsen, "Structured literature review of electricity consumption classification using smart meter data," Energies, vol. 10, no. 5, pp. 584, 2017.

[4] International Energy Agency (IEA), "The Future of Cooling in a Warming World," 2018.

[5] Z. Yan and H. Wen, "Performance Analysis of Electricity Theft Detection for the Smart Grid: An Overview," IEEE Transactions on Instrumentation and Measurement, vol. 71, pp. 1-28, 2022, doi: 10.1109/TIM.2021.3127649.

[6] B. Pes, "Handling Class Imbalance in High-Dimensional Biomedical Datasets," 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Napoli, Italy, 2019, pp. 150-155, doi: 10.1109/WETICE.2019.00040.

[7] Z. Wang, C. Cao and Y. Zhu, "Entropy and Confidence-Based Undersampling Boosting Random Forests for Imbalanced Problems," IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 12, pp. 5178-5191, Dec. 2020, doi: 10.1109/TNNLS.2020.2964585.

[8] 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 Transactions on Smart Grid, vol. 13, no. 2, pp. 1633-1644, March 2022, doi: 10.1109/TSG.2021.3134018.

[9] Z. Zheng, Y. Yang, X. Niu, H. -N. Dai and Y. Zhou, "Wide and Deep Convolutional Neural Networks for Electricity-Theft Detection to Secure Smart Grids," IEEE Transactions on Industrial Informatics, vol. 14, no. 4, pp. 1606-1615, April 2018, doi: 10.1109/TII.2017.2785963.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique", J. Artif. Intell. Res., vol. 16, pp. 321-357, Jan. 2002.

[11] Over Sampling Technique [Online], Available: https://imbalanced-learn.org/stable/over$_sampling.html$

[12] Under Sampling Technique [Online], Available: https://imbalanced-learn.org/stable/under$_sampling.html$

[13] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?", Mach. Learn., vol. 54, no. 3, pp. 255-273, 2004. https://doi.org/10.1023/B:MACH.0000015881.36452.6e

[14] "Supervised ML model". [Online], Available: https://scikit-learn.org/stable/supervised$_learning.htmlsupervised-learning.$

[15] E. Bisong, Building Machine Learning and Deep Learning Models on Google Cloud Platform, Springer, 2019.

[16] "SGCC Dataset", [online] Available: https://github.com/henryRDlab/ElectricityTheftDetection.

[17] J. Brownlee, How to Combine Oversampling and Undersampling, Jan. 2020, [online] Available: https://machinelearningmastery.com/combine-oversampling-and-undersampling-for-imbalanced-classification/.

[18] G. Van Rossum and F. L. Drake, Python 3 Reference Manual. Scotts Valley, CA: CreateSpace, 2009, ISBN: 1441412697.