Machine Learning Approach Based on Hybrid Features for Detection of Phishing URLs

Avinash Kumar Jha[1,#], Awishkar Ghimire[2], Aryan Mani Jha[3], Sushruti Mishra[2], Surendrabikram Thapa[2]

[1]Department of Civil Engineering, Delhi Technological University
[2]Department of Computer Science and Engineering, Delhi Technological University
[3]Department of Mechanical Engineering, Delhi Technological University
[#]avinash.jha.6696@gmail.com

ABSTRACT

Phishing attacks are one of the most widespread problems over the internet. A lot of internet users fall into the hands of attackers every day which accounts into millions of dollars of fraud around the globe every day. The availability of the internet among people who don't have the knowledge of cyber-attacks adds more to this problem. Thus, there is a need to employ intelligent algorithms to solve these serious problems. In this paper, we present different ways in which phishing URLs can be detected using machine learning algorithms. The URL based features as well as network-based features were used to feed to the machine learning classifiers. Similarly, other features that might add relevance to our problem are also discussed. The unbalanced dataset is made balanced using various oversampling and undersampling techniques and the performance for the various machine learning algorithms is evaluated for the dataset. The evaluation shows that the machine learning algorithms can show promising results in terms of precision, recall, f-score, and ROC AUC.

Keywords— Machine Learning, Phishing attacks, cybersecurity, ensemble learners, cybersecurity

1. INTRODUCTION

Machine learning algorithms have been widely used to solve various real-world problems [1]. From accelerating businesses to providing better healthcare, the use of machine learning algorithms is well explored [2]. The priorities of researchers in making the use of artificial intelligence (AI) for making lives easier has changed our lives in different ways. The never-ending exploration of the applications of AI in smart cities has led to numerous discoveries [3]. With the technologies evolving day by day, there have been some dark sides associated with it. The fraudulent activities are happening on a very large scale which needs to be stopped. It has been estimated that the loss due to cybercrimes is projected to reach $6 trillion by 2021, which is staggeringly huge [4]. One of the fraudulent activities that happen over the internet is phishing activities. There are a lot of ways in which attackers lure the victims into clicking the malicious links which may result in the victim losing their personal data and even money in some cases. Mostly, the attackers use emails, blog posts, chat sessions, etc. for posting the phishing URLs where it is highly likely that the victims click on them. [5] The links appear to be very tempting where attackers promise free money, high paying jobs, etc. Victims often fall into such fake promises and click the URLs. Phishing is even widespread with more users using the internet who don't even have basic knowledge about phishing and cyber-attacks. In this situation, there is a need for an automated system that helps in the identification of phishing URLs by using sophisticated learning techniques. Detecting malicious URLs is one of the most important steps towards preventing fraud over the internet. Thus, to make the internet a safer place, there is a great need for systems that help in identifying phishing URLs.

2. RELATED WORKS

A lot of work has been done in using machine learning techniques for the detection of suspicious URLs. James et al. [6] have researched the detection of phishing URLs using host-based features, lexical features, and page importance features. The URLs they used were collected from phistank.com, alexa.com, dmoz.com. The novelty in their approach lies in the feature extraction and the types of features they used. In their feature extraction technique firstly host-based analysis is performed and then relevant host-based features are extracted, and subsequently page/popularity-based feature analysis and lexical feature analysis are performed extracting the relevant features. Host-based features are those features that define the who hosts the websites and where/ how it is hosted. Page/popularity based features define how popular the website is among the users of the internet. Lexical features comprise those properties that arise out of the text of the URL itself and not the contents of the website. After feature extraction, they use a machine learning classifier. The classifiers they train on are the Naive

Bayes classifier, support vector machines, regression trees, and k-nearest neighbours. In their experiment, regression trees gave the highest accuracy. Similarly, Yan et al. [7] proposed an unsupervised learning algorithm that learns URL embeddings to get some features that can be used with machine learning algorithms. Kim et al. [8] also did a comprehensive research on exploring the similarity in the suspicious URLs which are due to the attacker's behavior. This enabled the authors to successfully classify the malicious and benign URLs with an accuracy of 70%. The past works give holistic information of what features can make URLs look more suspicious.

3. METHODOLOGY

The process goes through five major steps. The first and foremost step is to collect data for the study. The data collection step is followed by feature engineering where we take the features that are relevant and of our interest. After that, data preprocessing is done before feeding the data into machine learning algorithms. The performance of each classifier is evaluated as the final step. The overview of the methodology used in this paper is as shown in fig. 1.

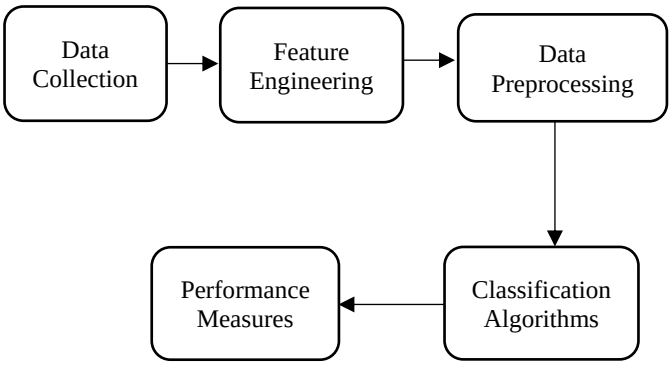

Fig. 1. Methodology of the experiment

A. Data Collection

The websites URL were collected from various sources like PhishTank etc. There are a total of 450,176 URLs out of which consists of 345,738 benign websites and 104,438 malicious websites. Statistically, 77% of the websites are benign and 23% of them are malicious. The URLs in the original form are unprocessed and need some preprocessing to extract the useful and relevant features that may serve as attributes for classifiers trained for the delineation of malicious URLs from the benign ones. The dataset used is publicly available.

B. Feature Engineering

The URLs that are related to phishing have certain anomalous features that distinguish themselves from the benign ones. Jeeva et al. used different features like domain length, number of slashes in the URLs, the number of special characters, etc. It is unanimously accepted by the researchers that certain length features and count of special tokens, digits, letters, directories, etc. play a vital role in the detection of anomalies in the URLs [9]. In our experiments, we use similar features like the length of the URLs, length of the hostname, length of the first directory, length of the path, and length of the top-level domain. Similarly, count features are also used like count of certain special characters (-, @, $, ?, /, %, . , =) and certain keywords like http, https, and www. Similarly, the count of digits, letters, and the number of directories were also taken as features that can influence the URLs trustability. On top of this, the binary features like the usage of IP (either "yes" or "no") and the use of shortening URL ("yes" or "no") have been used to make the features more robust. For the list shortening, if the shortening has been done using bit.ly, goo.gl, shorte.st, go2l.ink, x.co, ow.ly, t.co, tinyurl, tr.im, is.gd, cli.gs, yfrog.com, migre.me, ff.im, tiny.cc, doiop.com, short.ie, kl.am, wp.me, rubyurl.com, om.ly, to.ly, bit.do, t.co, lnkd.in, db.tt, qr.ae, adf.ly, bitly.com, cur.lv, tinyurl.com, url4.eu, twit.ac, su.pr, twurl.nl, snipurl.com, short.to, BudURL.com, ping.fm, post.ly, Just.as, bkite.com, snipr.com, fic.kr, loopt.us, ity.im, q.gs, is.gd, po.st, bc.vc, twitthis.com, u.to, j.mp, buzurl.com, cutt.us, u.bb, yourls.org, x.co, prettylinkpro.com, scrnch.me, filoops.info, vzturl.com, qr.net, 1url.com, tweez.me, v.gd, tr.im, and link.zip.net, the link has been labeled as shortened and the other links have been labeled as "not-shortened". In total, 21 features (5 lengths based + 14 counts based + 2 binary) were extracted and used in this research. In this experiment, the features are not just based on length but also based on counts of letters, numbers and binary values. Thus, the features are hybrid features and such features can help in building robust classifiers.

C. Data Preprocessing

The statistics of data used in this research show that more than 3/4th of the data belongs to one category and less than 1/4th of the data belongs to the other category. This situation cannot give correct performance measures in terms. The unbalanced data simply means that one class is very well represented in the dataset whereas the representation of the minority class is low. In order to tackle the unbalanced data, two major techniques are used in this research. First of all, the unbalanced data is undersampled so that both categories have nearly the same amount of data. After that, the performance measures are calculated for different classifiers. Similarly, the performance measures are also measured for original imbalanced data and oversampled data. For the undersampling of the data, the Near-miss technique is used whereas for oversampling of the data Synthetic Minority Oversampling Technique (SMOTE) is used.

Near-Miss: Near-miss [10] is an undersampling method that selects examples based on the distance of majority class examples to minority class examples. Actually, there are various variants of near-miss. In this research, the technique that selects majority class examples with minimum distance to each minority class example is used. This particular near-miss method used in this research is often known as Near-miss-3. After undersampling with Near-Miss-3, the data distribution becomes a bit balanced with 104,438 malicious websites and 70,574 benign websites. Statistically, there is 60% data belonging to the malicious category and 40% belonging to the benign category.

SMOTE: Synthetic Minority Oversampling Technique, popularly known as SMOTE, is an oversampling technique that generates synthetic samples from the minority class [11]. It is a technique based on nearest neighbors judged by Euclidean distance between data points in feature space. It is a widely used technique for handling imbalanced data and data augmentation especially when there is very little data in one category like fraud detection and malicious website detection. After oversampling using SMOTE, the data distribution becomes perfectly balanced with 345,738 observations in both malicious and benign websites category.

D. Classification Algorithms

The classification is done with ten different algorithms so that the best can be used for real-time inference. In this research, five machine learning algorithms and five ensemble learning algorithms have been used. All algorithms have been used with all three kinds of data i.e. undersampled, oversampled, and unbalanced data. The first five methods discussed below are machine learning methods and the remaining five are ensemble learners.

I. K-Nearest Neighbors (KNN)

K-Nearest Neighbors is a non-parametric clustering algorithm that helps in classification problems [12]. It works on a very simple idea of storing all the available observations and predicting the class of new instances by measuring its similarity with the stored observations. It is very useful especially with datasets with lower feature dimensions. When the dimension of data increases, it has a bit of difficulty in classification. Nevertheless, it is one of the widely used algorithms in machine learning and pattern recognition.

II. Support Vector Machine (SVM)

Support Vector Machines (SVMs) are famous machine learning classifiers developed by Vapnik et al. which are used both in classification and regression problems [13]. It has the ability to distinguish between the categories by drawing a hyperplane for categorization. When the algorithm is fed with the data, it creates an optimal hyperplane that categorizes given newer examples. In this experiment, SVM with Gaussian kernel is used. SVM is a bit slower with a large number of data because it requires the order of n2 computation while the model is trained, where n is the number of training data.

III. Decision Tree (DT)

A decision tree is a flowchart or tree-like representation which is based on the decisions and their possible consequences. It is a very basic algorithm and is capable of handling both categorical and numerical values [14]. This algorithm also works well with problems with multiple outputs. It is very simple and interpretable. Decision trees can moreover be visualized with the help of different libraries. In this experiment, the Gini index is used to measure the quality of the split. Unlike SVMs, the cost of using a decision tree is lower. It is logarithmic in the number of observations that are used for training the tree.

IV. Logistic Regression (LR)

Logistic Regression is a simple kind of algorithm that can be used to solve both regression and categorization problems. It is based on a logistic function for the prediction of the target values. The outcome of a logistic function is a probability value between 0 and 1 which gives the likelihood of a category being the correct one. In this experiment, the logistic regression has been used with C and penalty parameters as 1.0 and ‘l2’ respectively [15].

V. Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) is a technique to locate the ideal boundary design for a machine learning calculation. It iteratively makes little changes in accordance with a network setup of a machine-learning algorithm to diminish the mistake of the system [16]. After each training point, the configuration of the network is tuned to find the global minimum. Hingle loss and ‘l2’ penalty are used in this experiment.

VI. Random Forest (RF)

Random Forest is a kind of ensemble of learners that builds multiple decision trees at the time of training and the output is the modal class or the prediction of individual trees [15]. Though computationally expensive, it helps in better classification and also avoids overfitting to a larger extent. This amazing property of avoiding overfitting is due to the fact that the results of decision trees are averaged in a random forest algorithm.

VII. Gradient Boosting Classifier (GB)

This is another ensemble learner that is used to solve both classification and regression problems. It is a very powerful

boosting algorithm that uses decision trees. With its use increasingly in the Kaggle competitions for better performance, it has been widely used in classification tasks [17].

VIII. Adaptive Boosting (AdaBoost) (ADB)

Adaptive boosting, popularly known by its alias AdaBoost, is one of the most widely used classification algorithms that use decision trees [18]. It gives the flexibility of combining really weak classifiers into strong classifiers. The decision stumps which are the decision trees with a single split are used as weak learners which are combined into strong classifiers using adaptive boosting schemes.

IX. Extreme Gradient Boosting (XGBoost) (XGB)

This is one of the most loved machine learning algorithms. The algorithm is often implemented using XGBoost library which is an open-source software library. XGBoost is very fast for the implementation and hence is used in a wide range of applications [19].

X. Light Gradient Boosting Machine (LGBM)

Light Gradient Boosting Machine (LGBM) is yet another tree-based algorithm that is widely used in a variety of machine learning problems. Unlike other boosting algorithms that grow trees horizontally, LGBM is a quite different one that grows trees vertically [19].

E. Performance Measures

The classifiers are used and the validation is done using a 10 fold cross-validation technique. The accuracy, precision, recall, f-score, and AUC measures are calculated after a 10 fold cross-validation scheme. The formulas for the performance measures are given as:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1-Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

4. RESULTS AND DISCUSSION

First of all, the classifiers are used with the original dataset which is an unbalanced one. Then, the dataset is balanced once using the undersampling technique and again using the oversampling technique. The performance of the machine learning algorithms, as well as ensemble learners with original data, can be found in Table I. For the original dataset, AdaBoost shows the best performance with an accuracy of 99.07%. The precision and recall for the same are 0.98 and 0.99 respectively. Similarly, the f-score and ROC AUC for the AdaBoost algorithm are 0.99 and 0.9908 respectively. Also, further analysis was done with an undersampling technique called near-miss. The results given by the dataset which was undersampled using near-miss is shown in table II. With an undersampled dataset as well, the best performing algorithm was AdaBoost with an accuracy of 98.43%. The precision, recall and f-score are all equal to 0.98. The ROC AUC for the AdaBoost in undersampled data is 0.929. However, the gradient boosting classifier has a better performance in terms of ROC AUC with a value of 0.9837. Further experiments with oversampled data using SMOTE is shown in table III. Support Vector Machines (SVM) was best-performing algorithm with an accuracy of 99.50%. Similarly, the precision, recall and f-score were all equal to 0.999. Also, ROC AUC was very high when SVM was used with oversampled dataset. The accuracy of 99.07% with original dataset is good but the phishing detection is a kind of problem where we need very high accuracy. Even a slightest decline in performance can cause a huge monetary loss. Thus, using oversampled dataset is a great choice for this problem. Despite a high time for computation due to high amount of data in oversampled dataset, the better accuracy makes it very useful to use.

TABLE I. PERFORMANCE OF ALGORITHMS FOR ORIGINAL DATA

Classifier	Accuracy	Precision	Recall	F-score	ROC AUC
K-Nearest Neighbors (KNN)	0.9125	0.88	0.87	0.87	0.8669
Support Vector Machine (SVM)	0.9836	0.99	0.97	0.98	0.9667
Decision Tree (DT)	0.9873	0.98	0.99	0.98	0.9879
Logistic Regression (LR)	0.9937	0.99	0.99	0.99	0.9899
Stochastic Gradient Descent (SGD)	0.9941	0.98	0.99	0.98	0.9881
Random Forest (RF)	0.9906	0.98	0.99	0.99	0.9906
Gradient Boosting Classifier (GB)	0.9906	0.98	0.99	0.99	0.9404
Adaptive Boosting (AdaBoost) (ADB)	0.9907	0.98	0.99	0.99	0.9908
Extreme Gradient Boosting (XGBoost) (XGB)	0.9906	0.98	0.99	0.99	0.9906
Light Gradient Boosting Machine (LGBM)	0.9906	0.98	0.99	0.99	0.9905

TABLE II. PERFORMANCE OF ALGORITHMS FOR UNDERSAMPLED DATA

Classifier	Accuracy	Precision	Recall	F-score	ROC AUC
K-Nearest Neighbors (KNN)	0.7819	0.79	0.78	0.78	0.7789
Support Vector Machine (SVM)	0.9452	0.94	0.95	0.95	0.9471
Decision Tree (DT)	0.9733	0.98	0.97	0.97	0.9693
Logistic Regression (LR)	0.9828	0.98	0.98	0.98	0.9816
Stochastic Gradient Descent (SGD)	0.9782	0.97	0.97	0.97	0.9712
Random Forest (RF)	0.9786	0.98	0.98	0.98	0.9756
Gradient Boosting Classifier (GB)	0.985	0.99	0.98	0.98	0.9837
Adaptive Boosting (AdaBoost) (ADB)	0.9843	0.98	0.98	0.98	0.9824
Extreme Gradient Boosting (XGBoost) (XGB)	0.9816	0.98	0.98	0.98	0.9794
Light Gradient Boosting Machine (LGBM)	0.9813	0.98	0.98	0.98	0.9789

TABLE III. PERFORMANCE OF ALGORITHMS FOR OVERSAMPLED DATA

Classifier	Accuracy	Precision	Recall	F-score	ROC AUC
K-Nearest Neighbors (KNN)	0.9478	0.95	0.95	0.95	0.9478
Support Vector Machine (SVM)	0.9947	0.99	0.99	0.99	0.9947
Decision Tree (DT)	0.9919	0.99	0.99	0.99	0.9919
Logistic Regression (LR)	0.995	0.999	0.999	0.999	0.995
Stochastic Gradient Descent (SGD)	0.9945	0.99	0.99	0.99	0.9939
Random Forest (RF)	0.9938	0.99	0.99	0.99	0.9938
Gradient Boosting Classifier (GB)	0.9939	0.99	0.99	0.99	0.9939
Adaptive Boosting (AdaBoost) (ADB)	0.9935	0.99	0.99	0.99	0.9935
Extreme Gradient Boosting (XGBoost) (XGB)	0.9936	0.99	0.99	0.99	0.9936
Light Gradient Boosting Machine (LGBM)	0.9936	0.99	0.99	0.99	0.9936

TABLE IV. PERFORMANCE COMPARISON OF ALGORITHMS WITH BALANCED AND UNBALANCED DATA

Classifier	Original Data		Undersampled Data		Oversampled Data	
	F-score	ROC AUC	F-score	ROC AUC	F-score	ROC AUC
KNN	0.87	0.8669	0.78	0.7789	0.95	0.9478
SVM	0.98	0.9667	0.95	0.9471	0.99	0.9947
DT	0.98	0.9879	0.97	0.9693	0.99	0.9919
LR	0.99	0.9899	0.98	0.9816	0.999	0.995
SGD	0.98	0.9881	0.97	0.9712	0.99	0.9939
RF	0.99	0.9906	0.98	0.9756	0.99	0.9938
GB	0.99	0.9404	0.98	0.9837	0.99	0.9939
ADB	0.99	0.9908	0.98	0.9824	0.99	0.9935
XGB	0.99	0.9906	0.98	0.9794	0.99	0.9936
LGBM	0.99	0.9905	0.98	0.9789	0.99	0.9936

Table IV draws a comparison among the machine learning model's performances with balanced and unbalanced data. In most of the cases, the performance of the models with oversampled data is seen very well. Fig. 2 graphically represents and compares the performance of algorithms with balanced and unbalanced datasets.

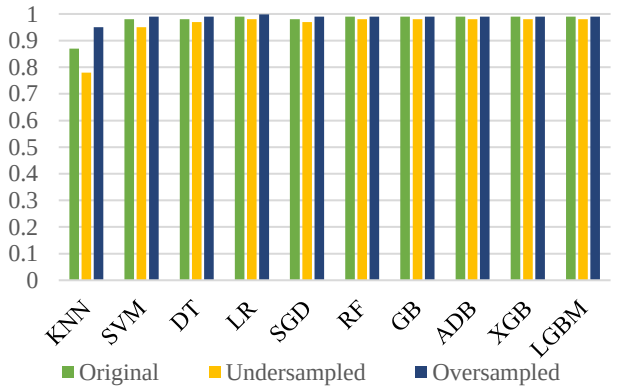

Fig. 2. F-score of the models with various data

It can thus be deciphered that with the given features and dataset, balanced data can be useful as compared to unbalanced data.

5. CONCLUSION AND FUTURE SCOPE

In this paper, we have discussed a framework for the detection of phishing URLs on the web using various features. The features are network-based and length based. The results are promising and can help us to detect the phishing URLs with great accuracy. Future works can involve the use of various other features which can be relevant to the detection of phishing URLs. More network-based features can be used in conjunction with length-based features to obtain higher accuracy. The algorithms used in the paper are used with default hyperparameters. In the future, the algorithms can be made more robust with hyperparameter optimization. This work uses machine learning algorithms only whereas deep learning algorithms can be further explored. Also, phishing has become more sophisticated with attackers using more advanced techniques for attacking. So, recent datasets that account for more sophisticated attacks should be used for carrying out the experiments.

REFERENCES

[1] A. Ghimire, S. Thapa, A. K. Jha, A. Kumar, A. Kumar, and S. Adhikari, "AI and IoT Solutions for Tackling COVID-19 Pandemic", in 2020 International Conference on Electronics, Communication and Aerospace Technology, 2020: IEEE.

[2] S. Thapa, S. Adhikari, U. Naseem, P. Singh, G. Bharathy, and M. Prasad, "Detecting Alzheimer's Disease by Exploiting Linguistic Information from Nepali Transcript," in International Conference on Neural Information Processing, 2020: Springer, pp. 176-184.

[3] C.-T. Lin et al., "IoT-based wireless polysomnography intelligent system for sleep monitoring," IEEE Access, vol. 6, pp. 405-414, 2017.

[4] A. Ghimire, S. Thapa, A. K. Jha, S. Adhikari, and A. Kumar, "Accelerating Business Growth with Big Data and Artificial Intelligence," in 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2020: IEEE, pp. 441-448.

[5] R. Verma and A. Das, "What's in a url: Fast feature extraction and malicious url detection," in Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics, 2017, pp. 55-63.

[6] J. James, L. Sandhya, and C. Thomas, "Detection of phishing URLs using machine learning techniques," in 2013 international conference on control communication and computing (ICCC), 2013: IEEE, pp. 304-309.

[7] X. Yan, Y. Xu, B. Cui, S. Zhang, T. Guo, and C. Li, "Learning URL Embedding for Malicious Website Detection," IEEE Transactions on Industrial Informatics, 2020.

[8] S. Kim, J. Kim, and B. B. Kang, "Malicious URL protection based on attackers' habitual behavioral analysis," Computers & Security, vol. 77, pp. 790-806, 2018.

[9] S. C. Jeeva and E. B. Rajsingh, "Intelligent phishing url detection using association rule mining," Human-centric Computing and Information Sciences, vol. 6, no. 1, pp. 1-19, 2016.

[10] I. Mani and I. Zhang, "kNN approach to unbalanced data distributions: a case study involving information extraction," in Proceedings of workshop on learning from imbalanced datasets, 2003, vol. 126.

[11] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, vol. 16, pp. 321-357, 2002.

[12] A. Saxena et al., "A review of clustering techniques and developments," Neurocomputing, vol. 267, pp. 664-681, 2017.

[13] C. Cortes and V. Vapnik, "Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273-297, 1995.

[14] S. Thapa, S. Adhikari, A. Ghimire, and A. Aditya, "Feature Selection Based Twin-Support Vector Machine for the diagnosis of Parkinson's Disease," in 2020 IEEE 8th R10 Humanitarian Technology Conference (R10-HTC), 2020.

[15] S. Thapa, P. Singh, D. K. Jain, N. Bharill, A. Gupta, and M. Prasad, "Data-Driven Approach based on Feature Selection Technique for Early Diagnosis of Alzheimer's Disease," in 2020 International Joint Conference on Neural Networks (IJCNN), 2020: IEEE, pp. 1-8.

[16] J. H. Friedman, "Stochastic gradient boosting," Computational statistics & data analysis, vol. 38, no. 4, pp. 367-378, 2002.

[17] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," Frontiers in neurorobotics, vol. 7, p. 21, 2013.

[18] R. E. Schapire, "Explaining adaboost," in *Empirical inference*: Springer, 2013, pp. 37-52.

[19] M. Massaoudi, S. S. Refaat, I. Chihi, M. Trabelsi, F. S. Oueslati, and H. Abu-Rub, "A novel stacked generalization ensemble-based hybrid LGBM-XGB-MLP model for Short-Term Load Forecasting," Energy, vol. 214, p. 118874, 2020.