# Machine Learning Algorithm for Detection of False Data Injection Attack in Power System

Ajit Kumar
*School of Comp. Science & Engineering*
*Soongsil University*
Seoul, South Korea
ajitkumar.pu@gmail.com

Neetesh Saxena
*School of Comp. Science & Informatics*
*Cardiff University*
Cardiff, UK
nsaxena@ieee.org

Bong Jun Choi
*School of Comp. Science & Engineering*
*Soongsil University*
Seoul, South Korea
davidchoi@soongsil.ac.kr

***Abstract*—Electric grids are becoming smart due to the integration of Information and Communication Technology (ICT) with the traditional grid. However, it can also attract various kinds of Cyber-attacks to the grid infrastructure. The False Data Injection Attack (FDIA) is one of the lethal and most occurring attacks possible in both the physical and cyber part of the smart grid. This paper proposed an approach by applying machine learning algorithms to detect FDIAs in the power system. Several feature selection techniques are explored to investigate the most suitable features to achieve high accuracy. Various machine learning algorithms are tested to follow the most suitable method for building a detection system against such attacks. Also, the dataset has a skewed distribution between the two classes, and hence data imbalance issue is addressed during the experiments. Moreover, because the response time is critical in a smart grid, each experiment is also evaluated in terms of time complexity.**

***Index Terms*—Data Injection Attack, Smart Grid, Machine Learning, Power System**

## I. Introduction

Today, the electric power grid is becoming more intelligent, called a smart grid, by adopting advanced Information and Communication Technology (ICT) [1]. This intelligence brings efficiency, accuracy, ease-of-use, and connectivity in the generation, distribution, and consumption of electricity. However, at the same time, it also introduces new vulnerability issues that result in various kinds of cyber-attacks [2], [3]. Any potential attack or disturbance in the functioning of the smart grid is critical because electricity has become a vital part of our daily lives. Also, most industrial devices (e.g., manufacturing plants) and social infrastructure (e.g., hospitals) run on electric power. The recent attack on the Ukraine grid infrastructure is an alarming example which calls for robust security requirements to protect against human errors and economic losses [4].

The False Data Injection Attack (FDIA) compromises the integrity of the data (value of sensors, meters, etc.) in which either attacker or environment manipulates the real values with falsified values. It is a critical and most occurring attack from the set of known attacks to the smart grid. Depending on the type of successful FDIAs, it can cause damage to physical infrastructure or result in economical loss [5] and, importantly, in many cases, it can be life threatening [6]. So, considering the scale of damage detecting and preventing FDIA in the power system is of high importance. Although there were some existing theoretical (modeling) and ICT-based solutions proposed and adopted, there still exist intrinsic limitations like the real-time generation of huge events, complex physical and cyber interface, post-analysis, etc. Therefore, we aim to use machine learning (ML) for FDIA detection, which will provide a pro-active detection and help perform FDIA detection in real-time.

In particular, we investigate and evaluate the capabilities of various ML algorithms to detect FDIA to fulfill the requirements of the smart grid. The objective has been achieved by conducting various experiments on the publicly available dataset on the power system [7]. The experiments in the proposed work focus on exploring the impact of feature selection on performance, finding ranking of features, and addressing the dataset's specific problem, such as missing value, corrupted value, and imbalanced dataset for training and testing various ML algorithms. The contributions of this work can be summarized as follows:

1) We develop a method to pre-process the existing power system dataset [7] and convert it into a binary class problem by filtering and merging FDIA events and non-FDIA events[1].
2) With different feature selection techniques (i.e., filter, wrapper, and embedded), we rank the features and compared (top and bottom 10 features) the outcome for selection techniques.
3) We also addressed imbalance class and data value issues (missing, infinite, etc.) using stratification and preprocessing of the dataset.
4) With a various set of features (the outcome of selection techniques and the number of features), we train and test nine ML algorithms in two setups (percentage split (70-30%) and 10-fold cross-validation) that can help to find the most suitable and robust model of FDIA detection for the smart power system.

This paper is organized as follows: Section II presents the related works. Section III presents the framework of the power system in which the events are recorded. Section IV explains different feature selection methods and presents the selected features as per rank. The result of various ML training and

[1]Data is made available https://github.com/urwithajit9/FDIA-classification.

testing are discussed in Section V. Lastly, the conclusion and future work is provided in Section VI.

## II. Related Work

The threat of a successful FDIA is critical and alarming, with the fast transition of the traditional grid to the smart grid. In recent work, Serkan et al. [5] have demonstrated the impact of FDIA through the insider attack model by modifying the memory address of Programmable Logic Controller (PLC). The authors demonstrated that change in PLC could directly affect bill management software and the final bill. The FDIA in the smart power system can understand in-depth through the recent (2015-20) survey works carried in this domain [6], [8]–[10]. These literature works explain various aspects of the FDIA, such as specific challenges posed by FDIA, available countermeasures, and their existing challenges. These works also discuss the current trend in detection technologies and big data, ML, and artificial intelligence for securing the smart power system.

In recent times, ML is used for FDIA detection in parallel with earlier state estimation and time-series analysis methods [11]. The performance of ML-based techniques has improved over time, and some interesting research is carried out in this domain. One-Class Support Vector Machine (OCSVM) was used to address the class labeling and availability of datasets for anomaly detection in Supervisory Control and Data Acquisition (SCADA) system [12]. The False data detection using anomaly detection (applying Principal Component Analysis (PCA)) and Distributed SVM given a new direction and addresses issues like stealthiness, and lower computation complexities for FDIA detection. The Distributed SVM has provable optimality and faster convergence rate so very suitable for larger size data and real-time detection [13]. Three supervised learning algorithms were used to detect the "direct" and "stealth" FDIA in the smart grid [14]. The balanced and imbalanced cases were considered, and experiments were done using the IEEE 30-bus system simulation. Under FDIA detection, Artificial Intelligence methods (ANN and ELM) were used to detect malicious meters [2]. The proposed methods were verified by using the dataset resulted from the mapping of NYISO load data on the IEEE 14 bus system. Yi Wang et at. [15] grouped FDIA into Cyber-space and Physical space and have proposed to use a Data-centric approach to detect and prevent FDIA using Big data and Margin Setting Algorithm (MSA) on synchronized Phasor Measurement Units (PMUs), in both simulated and Real-world, events dataset. Defu Wang et at. [16] applied ensemble ML techniques to detect FDIA attacks along with other cyber attacks (37-class fault) in the power system. The authors have used a publicly available dataset that contains readings of four PMUs and communication meta-data from IDS and firewall [7]. Cao et al. [11] discussed the scope and performance of ensemble learning for FDIA detection in great detail by explaining FDIA in the context of Cyber-Physical Power System (CPPS). The power system dataset [7] was used to train and test four ML algorithms (One R, J-Ripper, Random Forest (RF), and Naive Bayes (NB)) for the three-class problem, i.e., natural, no-event, and attack [17].

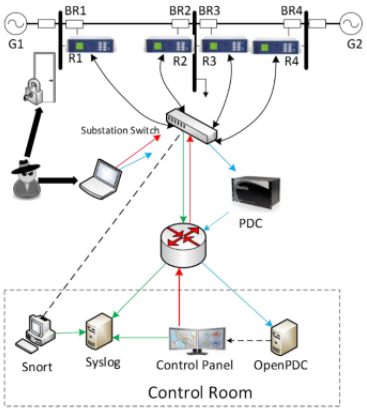


Fig. 1. The configuration of the power system framework for recording events [7].

The aforementioned related works on the FDIA and application of ML-based techniques indicate that ML-based solutions are good in resolving issues. In the future, the smart grid will have wide adoption of these solutions. It is also evident that most of the earlier works have used the power system dataset [7] and have considered event classification as a multi-class problem. In contrast to existing research work, the proposed work pre-processed the dataset suitable for binary classification. The original dataset also has a binary classification, but that is "Attack vs. Normal" events while the proposed work only selected "FDIA and non-FDIA" events, and explored the impact of feature selection to reduce the complexity (time and space) of ML-based classifiers. The proposed work also investigates the best performing classifiers with minimum number features and provides the reasoning of best-performing features.

## III. Power System Framework

A smart grid for the power system has a very complex and huge infrastructure, and due to 365 days X 24-hours usage, it is not possible to research and run new solutions directly to the infrastructure. The security and privacy issue also restrict access to data related to smart grid operations. To overcome these issues, most research uses Mississippi State University and Oak Ridge National Laboratory dataset[2] which contains the data of the power system having various attacks and normal events [7]. The experimental Power system framework configuration in which all the events were recorded is depicted in Fig. 1. In figure 1, the $G1$ and $G2$ are two power generators; R1-R4 are four Intelligent Electronic Devices (IEDs), which is used to switch on/off the breakers (BR1-BR4). The power system also has two lines between BR1-BR2 and BR3-BR4. Further, the power system is connected to a control room using substation switch and power distribution center (PDC).

[2]https://www.sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets

TABLE I
FEATURE DESCRIPTION WITH SIGNAL REFERENCE

| Feature | Description |
|---|---|
| PA1:VH – PA3:VH | Phase A - C Voltage Phase Angle |
| PM1: V – PM3: V | Phase A - C Voltage Phase Magnitude |
| PA4:IH – PA6:IH | Phase A - C Current Phase Angle |
| PM4: I – PM6: I | Phase A - C Current Phase Magnitude |
| PA7:VH – PA9:VH | Pos. – Neg. – Zero Voltage Phase Angle |
| PM7: V – PM9: V | Pos. – Neg. – Zero Voltage Phase Magnitude |
| PM7: V – PM9: V | Pos. – Neg. – Zero Current Phase Angle |
| PM10: V - PM12: V | Pos. – Neg. – Zero Current Phase Magnitude |
| F | Frequency for relays |
| DF | Frequency Delta (dF/dt) for relays |
| PA:Z | Appearance Impedance for relays |
| PA:ZH | Appearance Impedance Angle for relays |
| S | Appearance Impedance Angle for relays |

TABLE II
FILTER METHOD: TEN TOP AND BOTTOM FEATURES WITH INDIVIDUAL SCORE

| Top Features | | Bottom Features | |
|---|---|---|---|
| Features | Score | Features | Score |
| R2-PM11:I | 89.957696 | R3-PA4:IH | 0.506336 |
| R2-PM12:I | 77.299363 | R1-PM2:V | 0.410809 |
| R1-PM12:I | 76.535194 | snort_log2 | 0.396175 |
| R1-PM9:V | 72.015296 | R1-PA8:VH | 0.161182 |
| R3-PM11:I | 69.674591 | R1-PA:Z | 0.086954 |
| R2-PA6:IH | 68.635097 | R2-PA3:VH | 0.061740 |
| R1-PM8:V | 62.671352 | R2-PM2:V | 0.052888 |
| R2-PM8:V | 58.446877 | R3-PA3:VH | 0.052869 |
| R3-PM8:V | 58.252764 | R3-PM7:V | 0.012634 |
| R3-PA6:IH | 54.064154 | R1-PA9:VH | 0.006151 |

Further, in this section, a brief description of the features and dataset is presented that will help to understand the results of feature selection and ML algorithms training and testing.

### A. Features Description

The original dataset has 128 features, which come from two groups,1) PMUs measurement and 2) Control Room logs. There are 4 PMUs, and from each 29, measurements are collected, which contribute to total of 112 features and the control room logs are divided into the control panel, snort, and relay logs, each has 4 features, so it a total 12 features. The column naming in the dataset follows a naming convention, which helps to understand each feature better. Each PMUs measurement is represented as *R#-Signal Reference*, where R# represents the PMU number, which is R1, R2, R3, and R4 and *Signal Reference* is explained in Table I. The information in Table I is adopted from original dataset description document [3] and a detail explanation can be found from original document [3].

### B. Dataset Description

Pan et al. [7] has presented a very well explained and enriched dataset on the power system attack. The initial dataset is divided into 15 sets, and each has 37 power system event scenarios. These scenarios are further divided into *Natural Events*, *No Events*, and *Attack Events (data injection, command injection, etc.)* which has 8, 1 ,and 28 events respectively. Further, the initial dataset was randomly sampled and grouped into three datasets for (1) binary-class, (2) three-class, and (3) multi-class. The multi-class dataset has 37-classes representing each type of event as a class, and this dataset is in Attribute-Relation File Format (ARFF), which is suitable for the WEKA tool.

In this proposed work, the multi-class dataset is used after pre-processing (converting ARFF to CSV and filtering events). The scenarios related to normal (scenarios 1-6, 13, 14, 41) and data injection attacks (scenarios 7-12) were carved out from all 15 sets. Once all the events are filtered from each set, they are further merged and relabeled as 0 for the normal and 1 for the data injection scenario. After pre-processing the resulting dataset[1] has a total of 32296 samples and each has 128 features. The classes were distributed as 22714 normal events and 9582 data injection events, which indicate that the dataset is skewed towards normal events, and it is an imbalanced dataset. During pre-processing, the features (R1-PA:Z, R2-PA:Z, R3-PA:Z, R4-PA:Z) with infinite as value were replaced with 0.

[3]http://www.ece.uah.edu/~thm0009/icsdatasets/PowerSystem_Dataset_README.pdf

## IV. FEATURE SELECTION

The feature selection is the process of ranking and selecting features that contribute to predicting the output variable with high probability or support. It is one of the core components of the ML-based detection pipeline, and it has a major impact on the model in terms of both the performance and computational (time and space) requirements. Feature selection is grouped into three categories based on how it is processed: (1) filter method, (2) wrapper method, and (3) embedded method [18].

### A. Filter Method

The Filter method is also known as the univariate selection in which features are selected, having a strong relationship with the output variable. This relationship between features and output variables is measure through well established statistical tests such as chi-squared. Filter methods are fast and easy to interpret [18]. Table II lists out the top and bottom 10 features with their score that were selected using the filter method by applying ANOVA F-value. From Table II, we can observe that feature related to *Magnitude* of PMUs got higher ranking in filter method. The reason could be attributed to the fact that features related to *Magnitude* got higher values, while features related to *Angle* have smaller value and even negative value that impacts the ANOVA calculation for ranking these features.

### B. Wrapper Method

The wrapper method is a model-specific feature selection. In this method, features are selected as per their importance with a particular model's performance. It is obvious that feature rank varies with different models and features are tightly coupled with models. The wrapper methods are computationally

TABLE III
WRAPPER METHOD: TEN TOP AND BOTTOM FEATURES WITH ITS IMPORTANCE

| Top Features | | Bottom Features | |
|---|---|---|---|
| Features | Importance | Features | Importance |
| R4-PM2:V | 0.017745 | R4-PA9:VH | 0.000044 |
| R1-PM5:I | 0.016291 | R2-PA9:VH | 0.000041 |
| R4-PM5:I | 0.016251 | snort_log1 | 0.000024 |
| R2-PM5:I | 0.016208 | snort_log3 | 0.000021 |
| R3-PA7:VH | 0.015746 | control_panel_log3 | 0.000009 |
| R1-PM2:V | 0.015684 | control_panel_log2 | 0.000002 |
| R2-PA3:VH | 0.015622 | control_panel_log1 | 0.000000 |
| R3-PA3:VH | 0.015482 | control_panel_log4 | 0.000000 |
| R3-PM5:I | 0.015026 | snort_log2 | 0.000000 |
| R2-PA7:VH | 0.014930 | snort_log4 | 0.000000 |

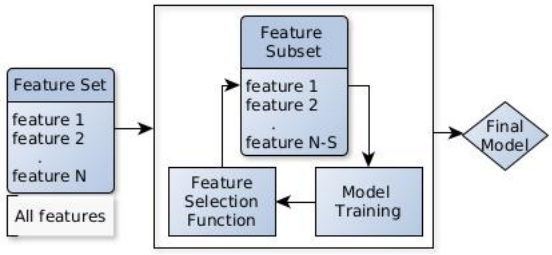


Fig. 2. Process of embedded feature selection.

expensive, and the greedy search used to find the best features is not optimal and often suspected of false starts [18]. Table III lists out the top and bottom 10 features with their importance that were selected using the wrapper method. From Table III it can be observe that both *Magnitude* and *Angle* related features equally (6 and 4 respectively) appeared in top 10 features based on their importance. The reasons behind this can be contributed to the fact that tree-based classifier is not affected by the number of features but affected by the participation of the value in classification accuracy. This can also be verified by the bottom 10 features in which mostly controls logs related features are listed, and these features are Boolean and very sparse. They do not contribute to classification (importance value equal to zero).

### C. Embedded Method

The embedded method is a hybrid approach that combines the filter and the wrapper method to get the best of these two methods (speed of filter) and better performance (as in wrapper) with the model. This is achieved by making the feature selection as part of the model training itself, i.e., the ML algorithm keeps selecting the features during the on-going model training process [18]. Many ML algorithms recursively perform embedded feature selection, i.e., after training model with all features, a subset of features are selected based on importance, and re-training repeats until the convergence condition is met. The LASSO and RIDGE regression are two common examples of embedded feature selection. Through block diagram Fig. 2 display the overall process of embedded feature selection.

## V. EXPERIMENTS AND RESULTS

This section presents results of all experiments carried out to train and test various ML algorithms on a different subset of features (ALL, Top10, Top20, Top30, Top40, and Top50).

### A. Experimental System

All experiments are carried out with the system that has the latest version of Ubuntu 20.04 LTS, 64-bit OS running on Intel i5-6200U CPU 2.30 GHz quad-core processor with 16 GB primary and 1 TB secondary memory. Python programming language with required modules (pandas, NumPy, matplotlib, CSV, etc.) is used to conduct all experiments. The Scikit-learn framework is used to carry out all ML-related tasks such as feature selection, model training, and testing.

### B. Machine Learning Algorithms

The working principle of each ML algorithm is different; some works on conditional probability while others use distance calculation. To have a better understanding of different learning techniques, different ML algorithms, i.e., Regression, Naive Bayes (NB), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM) and Ensemble methods (Bagging and Boosting), are selected for training and testing with the same feature set [19]. This experimental arrangement provides a common platform to compare the performance and help decide the best performing models.

### C. Training and Testing

The training and testing are performed with two setups: (1) percentage split (70-30) and (2) 10-fold cross-validation. The training and testing with cross-validation methods result in a more robust model and handle the overfitting issue of training [20].

*1) Percentage Split (70-30):* In percentage split setup, the initial dataset is split into training and testing part as per giving percentage. For example, using a 70-30 ratio, the training dataset will have $70\%$ of the randomly selected sample, and the rest $30\%$ will be treated as a testing dataset. This method gives an approx model and suffers from overfitting issue. First, all ML-algorithms were trained with a 70-30% split on all features. Fig. 3 shows the accuracy of all algorithms, and with $92\%$ accuracy Random Forest is the best performing model in the ensemble group, and with $85\%$ accuracy Decision Tree outperform the other individual model.

For the model's suitability to the smart grid environment, the time complexity is a major performance metric. So, the model building process (training and testing) is benchmarked with an execution time that will help to understand the time complexity of each algorithm. Table IV lists out the execution time of each algorithm in seconds. We can observe that Naive Bayes (NB) takes the minimum while Bagging with SVC takes the maximum time. The Naive Bayes is fast because it reuses the prior probability values for calculating the posterior, but only if it holds the conditional independence assumption. Bagging takes more time because it randomly re-sampled the dataset and generate multiple subsets of the dataset to train the base classifier.

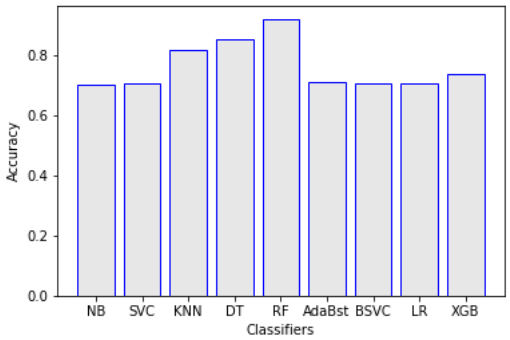


Fig. 3. Accuracy of ML algorithms on all features.

TABLE IV
TIME TAKEN BY ALL THE CLASSIFIERS FOR TRAINING AND TESTING

| Classifiers | Time Taken (Sec.) |
|---|---|
| NB | 0.132 |
| SVC | 9.770 |
| KNN | 7.280 |
| DT | 3.580 |
| RF | 17.900 |
| Adaboost | 11.800 |
| BSVC | 533.000 |
| LR | 1.140 |
| XGB | 62.000 |

*2) 10 Fold Cross-validation:* The training and testing with a percentage split given an initial estimation of algorithms' performance and a robust performance measuring the proposed work have trained and tested all algorithms with a 10-fold cross-validation method. After training and testing all algorithms with 10-fold cross-validation, its performance is compared with the percentage split methods. Table V and Fig. 4 shows the accuracy of all the algorithms in both percentage split and 10-fold cross-validation. It can be observed that the accuracy of many algorithms (RF, SVC, kNN, DT, and GB) gets reduced in 10-fold cross-validation. In contrast, other algorithms (LR, NB, Adaboost, and Bagging) either retained the same accuracy or only reduced slightly. The first set of algorithms has an inherent limitation of overfitting, while other sets of algorithms can handle overfitting.

From the complexity result, we can see that Bagging with SVC takes a longer time to train and test, so it was dropped from further experiments. After understanding the algorithms' performance with percentage split and 10-fold cross-validation, the proposed work aims to determine the performance variation of algorithms with a subset of features. These features were selected separately using filter and wrapper methods and grouped into the following sets (ALL, Top10, Top20, Top30, Top40, and Top50). So, we have 11 subsets of our initial dataset, and eight selected ML algorithms were trained and tested with 10-fold cross-validation. The accuracy of all algorithms on five feature sets selected using the filter method is shown in Fig. 5. Three kinds of effects, namely (1) no change, (2) increase, and (3) decrease in the accuracy, can be observed while the number of selected features increases from 10 to 50. The accuracy of SVC, Adaboost, and GB decreases while the accuracy of LR, kNN, and DT increases. The accuracy of RF and NB does not significantly change.

TABLE V
ACCURACY OF CLASSIFIERS IN TRAIN-TEST SPLIT AND 10-FOLD CROSS-VALIDATION

| Classifiers | Split (70-30) | 10-fold (CV) |
|---|---|---|
| Random Forest (RF) | 92.00 | 70.88 |
| Support Vector (SVC) | 70.74 | 58.40 |
| Linear Regression (LR) | 70.70 | 70.17 |
| Naive Bayes (NB) | 70.25 | 69.36 |
| Nearest Neighbors (kNN) | 81.69 | 62.76 |
| Decision Tree (DT) | 85.15 | 63.20 |
| Adaboost (Ada) | 70.86 | 69.26 |
| Gradient Boosting(GB) | 73.53 | 70.00 |
| Bagging SVC(BSVC) | 70.66 | 70.19 |

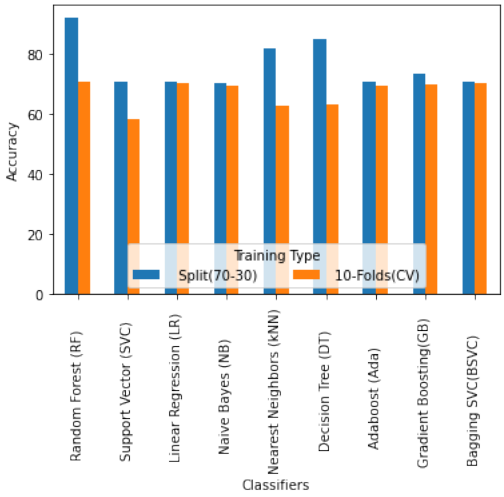


Fig. 4. Accuracy of classifiers in train-test split and 10-fold cross-validation.

Similar to the filter method, five different datasets are prepared using a subset of features, and all the eight algorithms are trained and tested with 10-fold cross-validation. The accuracy of these algorithms on all five datasets are shown in Fig. 6. LR has no change in its accuracy, while the accuracy of kNN increases with an increasing number of features. The NB recorded a decline in the accuracy and has a significant accuracy difference with the top10 and other features. The RF, SVC, DT, Adaboost, and GB have either mixed changed (increase with either end of feature set, i.e., top10 to top50) or no significant increase or decrease in accuracy with an increasing number of features. For example, the accuracy of RF and DT dropped and recorded a minimum with top30 features while it is approximately constant with other feature sets. In other groups, the change in accuracy of SVC and Adaboost is not related to an increase in the number of features.

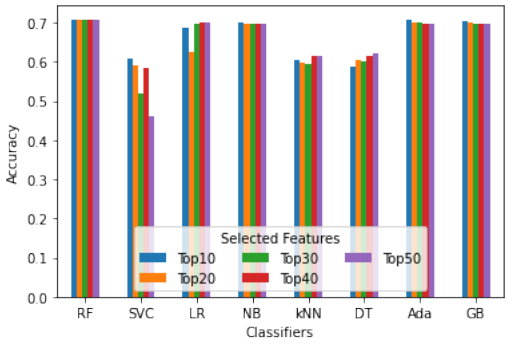


Fig. 5. Accuracy of classifiers with selected features using filter methods.

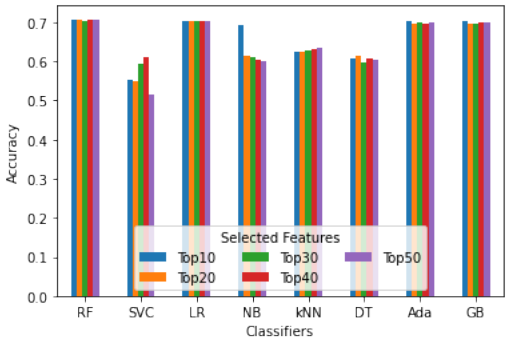


Fig. 6. Accuracy of classifiers with selected features using wrapper methods.

## VI. Conclusion and Future Scope

The proposed work aims to explore the suitability of ML for detecting FDIA in the power system. This was achieved by conducting various experiments with different feature selection methods and training and testing various ML algorithms with a different setup. Through experiments, it was found out that feature selection methods directly relate to the performance of ML algorithms. We observed that the algorithms' performance varies according to the type of selection. For example, the accuracy of NB is not affected by the increasing number of filter method features while decreasing with the wrapper method's features. Table II and Table III clearly indicate that output of selection methods results selecting different features. It was also observed that there is a significant dropped in performance (accuracy) of algorithms when trained with percentage split and 10-fold cross-validation. For example, the Random forest decreased from 90% to 70%. Based on experimental results and considering time and performance trade-off Random forest is the most suited model for FDIA detection in the power system. It should also be noted that its performance is constant with either filter or wrapper methods, and maximum accuracy is achieved with the minimum number of features. The performance enhancement is considered as a future work in which algorithms performance will be measure and compare on various other metrics such as Receiver Operating Characteristic (ROC), Area Under Curve (AUC), Partial AUC, False Positive Rate (FPR), True Positive Rate (TPR), etc.

## Acknowledgement

This research was supported by the National Research Foundation (NRF), Korea (2019R1C1C1007277) funded by the Ministry of Science and ICT (MSIT), Korea. This research was also supported by the Cardiff University HEFCW GCRF Small Project: Secure, Low-Cost, and Efficient Energy Solution (SP113).

## References

[1] E. R. Griffor, C. Greer, D. A. Wollman, and M. J. Burns, "Framework for cyber-physical systems: Volume 2, working group reports," 2017.

[2] K. Khanna, B. K. Panigrahi, and A. Joshi, "Ai-based approach to identify compromised meters in data integrity attacks on smart grid," *IET Generation, Transmission & Distribution*, vol. 12, no. 5, pp. 1052–1066, 2017.

[3] Y. Maleh, M. Shojafar, A. Darwish, and A. Haqiq, *Cybersecurity and Privacy in Cyber Physical Systems*. CRC Press, 2019.

[4] G. Liang, S. R. Weller, J. Zhao, F. Luo, and Z. Y. Dong, "The 2015 ukraine blackout: Implications for false data injection attacks," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3317–3318, 2016.

[5] S. Gönen, H. H. Sayan, E. N. Yılmaz, F. Üstünsoy, and G. Karacayılmaz, "False data injection attacks and the insider threat in smart systems," *Computers & Security*, p. 101955, 2020.

[6] S. Aoufi, A. Derhab, and M. Guerroumi, "Survey of false data injection in smart power grid: Attacks, countermeasures and challenges," *Journal of Information Security and Applications*, vol. 54, p. 102518, 2020.

[7] S. Pan, T. Morris, and U. Adhikari, "Developing a hybrid intrusion detection system using data mining for power systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 3104–3113, 2015.

[8] Z. Guan, N. Sun, Y. Xu, and T. Yang, "A comprehensive survey of false data injection in smart grid," *International Journal of Wireless and Mobile Computing*, vol. 8, no. 1, pp. 27–33, 2015.

[9] G. Liang, J. Zhao, F. Luo, S. R. Weller, and Z. Y. Dong, "A review of false data injection attacks against modern power systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1630–1638, 2016.

[10] A. S. Musleh, G. Chen, and Z. Y. Dong, "A survey on the detection algorithms for false data injection attacks in smart grids," *IEEE Transactions on Smart Grid*, vol. 11, no. 3, pp. 2218–2234, 2019.

[11] J. Cao, D. Wang, Z. Qu, M. Cui, P. Xu, K. Xue, and K. Hu, "A novel false data injection attack detection model of the cyber-physical power system," *IEEE Access*, vol. 8, pp. 95 109–95 125, 2020.

[12] L. A. Maglaras and J. Jiang, "Intrusion detection in scada systems using machine learning techniques," in *2014 Science and Information Conference*. IEEE, 2014, pp. 626–631.

[13] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, and Z. Han, "Detecting stealthy false data injection using machine learning in smart grid," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1644–1652, 2014.

[14] J. Yan, B. Tang, and H. He, "Detection of false data attacks in smart grid with supervised learning," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 1395–1402.

[15] Y. Wang, M. M. Amin, J. Fu, and H. B. Moussa, "A novel data analytical approach for false data injection cyber-physical attack mitigation in smart grids," *IEEE Access*, vol. 5, pp. 26 022–26 033, 2017.

[16] D. Wang, X. Wang, Y. Zhang, and L. Jin, "Detection of power grid disturbances and cyber-attacks based on machine learning," *Journal of Information Security and Applications*, vol. 46, pp. 42–52, 2019.

[17] M. Panthi, "Anomaly detection in smart grids using machine learning techniques," in *2020 First International Conference on Power, Control and Computing Technologies (ICPC2T)*. IEEE, 2020, pp. 220–222.

[18] S. Kotsiantis, "Feature selection for machine learning classification problems: a recent overview," *Artificial Intelligence Review*, vol. 42, no. 1, pp. 157–176, 2011.

[19] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[20] J. Shao, "Linear model selection by cross-validation," *Journal of the American statistical Association*, vol. 88, no. 422, pp. 486–494, 1993.