MACHINE LEARNING-ENHANCED INTRUSION DETECTION SYSTEM FOR ACCELERATED THREAT RESPONSE THROUGH FEATURE SPACE REDUCTION IN CRITICAL INFRASTRUCTURES

by

Kritika Sharma

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
August 2024

To my cherished family, dear friends, and wise mentors
this thesis is a tribute to your unwavering belief in me. Thank you
for being the guiding stars on my journey.

Table of Contents

List of Tables

List of Figures

Abstract

Critical Infrastructure (CI) forms the backbone of any nation, ensuring the seamless operation of various sectors such as manufacturing, gas pipeline systems, nuclear power plants, transportation, etc. The deployment of Industrial Control Systems (ICSs) and Supervisory Control and Data Acquisition (SCADA) systems facilitates the management and remote monitoring of the industrial processes. However, this advancement has also rendered ICSs vulnerable to numerous cyber-attacks. Security is crucial to prevent significant economic losses and potential loss of life and a highly responsive Intrusion Detection System (IDS) is vital for safeguarding CI. IDSs often rely on extensive network traffic that includes irrelevant features, leading to prolonged response time. To address these challenges, we propose a novel approach called Statistical Parameters - Selective Promising Feature Selection (SP-SPFS). This method ranks the most relevant features based on statistical parameters and selects the most effective features using a forward selection process. We evaluate SP-SPFS by comparing it with other feature ranking and selection methods, including Weighted Feature Importance (WFI) and Forward Feature Selection (FFS). Specifically, we analyze four combinations: SP-SPFS, SP-FFS, WFI-SPFS, and WFI-FFS. The effectiveness of these approaches is assessed using tree-based classifiers, namely, Decision Tree (DT), Random Forest (RF), Gradient Boost (GB), and Extreme Gradient Boost (XGB) on the Gas Pipeline dataset from Mississippi State University (MSU) and its three clusters namely, Command, Function, and Response. Performance metrics such as execution time, accuracy, f1-score, precision, and recall are evaluated using 10-fold cross-validation. Our findings show that SP-SPFS achieves the highest performance: 99.22% accuracy in 24.24 seconds with 14 features on the full dataset. For clusters, SP-SPFS-RF reaches 99.24% accuracy with 10 features in 179.13 seconds (Command), 99.61% with 11 features in 239.79 seconds (Function), and 98.62% with 7 features in 12.4 seconds (Response). Overall, SP-SPFS effectively reduces execution time while maintaining high performance.

List of Abbreviations Used

Acronyms

ARFF Attribute Relationship File Format

BATADAL Battle of the Attack Detection Algorithms

CCA Canonical Correlation Analysis

CD/CPE Combining Density and Class Probability Estimation

CFS Correlation Based Feature Selection

CI Critical Infrastructure

CRC Cyclic 21 Redundancy Code

CSV Comma Separated Value

CV Cross Validation

DDoS Distributed Denial of Service

eSNN evolving Spiking Neural Network

FFS Forward Feature Selection

FN False Negative

FP False Positive

GBFS Gradient Boosting Feature Selection

GB Gradient Boost

HMI Human Machine Interface

HNA-NN Hierarchical Neuron Architecture based Neural Network

IBL Instance-Based Learning

ICA Independent Component Analysis

ICS Industrial Control System

IDS Intrusion Detection System

IWP-CSO Intrusion Weighted Particle based Cuckoo Search Optimization

kNN k Nearest Neighbour

LRC Linear Redundancy Code

ML Machine Learning

MTU Master Terminal Unit

OCSVM One-Class Support Vector Machines

PCA Principal Component Analysis

PID Proportional-Integral-Derivative

PLC Programmable Logic Controllers

RBF Radial Basis Function

RF Random Forest

RM Ranking Mechanism

RTU Remote Terminal Unit

SCADA Supervisory Control and Data Acquisition

SPFS Selective Promising Feature Selection

SP Statistical Parameters

SVMNN Multilayer Perceptron Neural Network

SVM Support Vector Machine

SWAT Secure Water Treatment

TN True Negative

TP True Positive

WADI Water Distribution Testbed

WEKA h Waikato Environment for Knowledge Analysis

WFI Weighted Feature Importance

XGB Extreme Gradient Boost

Acknowledgements

I would like to extend my deepest gratitude to all those who have supported and guided me throughout the course of this endeavor. As someone very rightly said, “A mentor empowers a person to see a possible future and believe it can be obtained”. I believe Dr. Srinivas Sampalli, my supervisor, is a personification of this quote. I would like to thank him for his invaluable guidance, insightful feedback, and unwavering support which has been instrumental in the completion of this work. If I could embody even half of your persona, I would consider myself very fortunate.

I am profoundly grateful to Dr. Darshana Upadhyay, for her continuous encouragement, unending patience, keen understanding, and for fostering an environment of intellectual curiosity. Your expertise and dedication have been a constant source of inspiration now and in the future as well. You have been my compass in finding my way through this, without your guidance and help I would have been aimless.

My heartfelt thanks go to my family for their unconditional love and support. To my parents, your sacrifices, encouragement, and belief in my abilities have been the bedrock upon which I stand. Thank you for giving us the freedom, guidance, and unparalleled love to choose our destiny. To my sibling and his family, your constant support, love, and understanding have been a source of great strength.

I would also like to express my gratitude to my friends, Neha, Lav, Nupur, Sushumna, and Arpit, whose camaraderie and encouragement have been invaluable. Moving to a new country had its challenges but your support and love made this very difficult journey filled with love, laughter, and uncountable precious memories. I also want to thank my friends back home, Karan and Archana whose friendships truly know no bounds!

Chapter 1

Introduction

The advent of technology has ushered in an era of unparalleled convenience, yet it has also introduced a plethora of cyber-threats. This duality has fostered a complex relationship between humans and technology, characterized by both admiration and apprehension. Presently, cyber-threats are extending beyond individuals and infiltrating the organizations that underpin national infrastructure. These organizations form the Critical Infrastructures (CIs) of a nation, regulating essential services across diverse sectors such as agriculture, healthcare, nuclear energy, transportation, financial services, energy, civil and chemical engineering, gas pipeline systems, water plants, and research [3]. Consequently, any attack on these infrastructures has a profound impact on society at large.

Industrial Control Systems (ICSs) are pivotal in managing industrial processes within these CIs. Comprising hardware, software, operators, networks, links, etc., [4] ICSs facilitate seamless distribution and regulation of resources. ICSs traditionally were not integrated into the internet but with the advent of technology and the Internet, the transition to remote monitoring and surveillance took place. Supervisory Control and Data Acquisition (SCADA) systems specifically oversee the monitoring and control of ICSs. SCADA systems encompass various components, including Human-Machine Interface (HMI), Master Terminal Unit (MTU), Remote Terminal Unit (RTU), Communication Network, Programmable Logic Controller (PLC), Sensors, and Actuators [5]. The integration of SCADA systems enables continuous monitoring and timely data availability, which are crucial for real-time operations. As mentioned above, ICSs has various applications ranging from manufacturing, energy sector, water and wastewater treatments and distribution, transportation systems, chemical and pharmaceutical domain, automation, mining, telecommunications, etc. Historically, ICSs have been frequent targets of cyber-attacks, with an increasing number of incidents each year, as detailed in Chapter 3. Alimi et al. [6] discuss

the increase of cyber-attacks on ICSs. They also discuss several reported incidents of cyber-attacks on gas pipeline systems across the world such as the Night Dragon attack[7], Shamoon attack on the largest energy company in the world [7], and attack on Saipem company [7]. Given the significant financial and human impact of these attacks, it is imperative to address these vulnerabilities. To combat such issues an Intrusion Detection System (IDS) will be an optimum choice. IDSs play a crucial role in safeguarding the SCADA systems by monitoring and analyzing data to detect threats [8]. A lot of work has been done in developing IDS for ICSs. There exist different kinds of IDS that focus on different components to identify intrusions. Different categories of IDS include [9], [8]:

- Network-based SCADA IDS: In this approach, the data packets transmitted between various components within the SCADA system are utilized. These IDSs exhibit rapid computational performance by examining only the packet headers rather than the entire content. However, this methodology renders these IDSs vulnerable to cyberattacks that exploit malicious content embedded within the packet payloads.

- Application-based SCADA IDS: In this approach, the data generated by the sensors and actuators is utilized. Any deviation from the usual values will lead to it being inferred as a possible cyberattack.

- Signature-based SCADA IDS: In this approach, the intrusions are detected by matching them against a database of attack signatures of known attacks. However, it isn't possible to detect new attacks using such IDSs.

- Anomaly-based SCADA IDS: In this approach, the intrusions are detected by detecting abnormal behavior in system values. It is possible to detect new attacks as any deviation gets highlighted.

- Specification-based SCADA IDS [10]: This approach utilizes the comparison of the monitored values with predefined protocols for detecting intrusions.

There are various IDSs available for the SCADA system such as Snort, Suricata, and Bro [10]. Highly available IDSs are integral to the smooth working of ICSs by ensuring

continuous monitoring, data protection in terms of integrity and confidentiality, swift attack detection, and enhanced resilience [11].

IDS coupled with Machine Learning (ML) techniques has gained a lot of attraction in the past couple of years. The large volume of data generated by the SCADA sensors and other components can be used to develop a system or method capable of learning and identifying the patterns that help in anomaly detection [12]. Numerous publicly available datasets for different ICSs can be employed to develop mechanisms capable of detecting, predicting, and classifying potential intrusions. Some notable publicly available datasets include KDD98 [13], KDD99 [14], NSL-KDD [15], SWAT (Secure Water Treatment)[16], Gas Pipeline [17], WADI (Water Distribution Testbed)[16], Power System Attack dataset [18], BATADAL (Battle of the Attack Detection Algorithms) [16], etc. as mentioned in [19], [6].
There has been a constant and steady increase in the use of ML to detect intrusions in ICSs. The increase in the use of ML for SCADA security is highlighted in [6]. According to [10] ML covers over 38% of the total detection methodologies used in IDSs. For any IDS, a reduction in response time while identifying intrusions is of integral importance. This can be achieved in various ways with the use of techniques such as normalization, feature selection, and dimensionality reduction which are capable of spontaneous detection and a swift response. Given the substantial size and feature set of ICS datasets, normalizing data expedites ML model processing. Also, a huge feature set can serve as a liability to the ML models if they don't contribute towards deducing the prediction. Hence, selecting the features that enhance the prediction capability of the model and reduce the response time due to reducing the overall number of features needed for optimum performance [20] is an approach worth exploring. Researchers have used different Feature Selection techniques such as Principal Component Analysis (PCA), Canonical Correlation Analysis (CCA), Independent Component Analysis (ICA) [4], Functional Differential Analysis (FDA) and Cost Matrix [21], and as mentioned by [20]: Forward feature selection (FFSA), Modified mutual information-based feature selection (MMIFS), Linear correlation-based feature selection (LCFS), Minimal Redundancy Maximal Relevance (mRMR) and mRMR#, Fast Correlation Based Filter algorithm (FCBF) and FCBF#, Joint Mutual Information Maximisation (JMIM) and Normalized JMIM, and Euclidian

distance-based selection.

1.1 Motivation and Research Gaps

Our research focuses on the gas pipeline system in ICSs using the gas pipeline dataset by [22]. We have used the entire gas pipeline dataset as well as three clusters created from the original Gas Pipeline dataset namely Command, Function, and Response. The gas pipeline system is responsible for the smooth distribution of gas to the users through a complex network of pipelines. This system is monitored and controlled with the help of the SCADA systems. There has been work done for detecting anomalies using gas pipeline datasets however, there lies certain shortcomings in the present efforts:

- Many ML-based IDSs use entire datasets, resulting in lengthy response times and the inclusion of irrelevant features. Feature selection techniques are underutilized in addressing this issue.
- The performance of feature selection techniques can be improved further by using a feature ranking mechanism. However, that also remains an unexplored area.
- In the event of a cyberattack fast detection leading to a fast response is an integral component of any IDS. Yet response time optimization is often overlooked in ML classifiers used for IDSs.
- Publicly available datasets often suffer from imbalance, impacting model performance.

To address these issues, we focused on reducing the response time of ML classifiers using two feature ranking approaches namely a novel Statistical Parameter (SP) and Weighted Feature Importance (WFI) based on the work done in [23] to hierarchically arrange the features as well as reduce the dataset dimensionality by using the feature selection methods. In our work, we have also introduced a novel Feature Selection algorithm called Selective Promising Feature Selection (SPFS) and compared it with the Forward Feature Selection (FFS) algorithm used by [20]. We validated classifier performance using 10-fold cross-validation to select the optimal performer based on

total execution time, accuracy, F1 score, and other performance metrics to conduct a comparative analysis between different feature ranking mechanisms, feature selection algorithms, and multiple ML classifiers. Through this research, we make the following contributions:

1. We devised a methodology capable of distinguishing attack scenarios from normal operations while simultaneously enhancing performance metrics.

2. We assessed two ranking mechanisms, namely the SP approach and the WFI method, applied to three clusters of the Gas Pipeline dataset, both with and without preprocessing. This assessment yielded a hierarchically organized set of features, ranked by their significance within the dataset.

3. We introduced an innovative feature selection algorithm called SPFS, which selectively retains only those features that enhance classifier performance. This novel algorithm was benchmarked against the FFS algorithm using multiple machine learning classifiers on three clusters of the Gas Pipeline datasets, to identify the most effective feature selection methodology.

4. We used the SP-SPFS approach on the gas pipeline dataset to understand the impact of our proposed approach on the performance of the classifiers. We compared our results with other state-of-the-art work as well as against the performance of classifiers when no feature selection was employed.

5. We conducted a comparative analysis of various machine learning classifiers in each phase to ascertain the optimal combination for anomaly detection. The study incorporated 10-fold Cross-Validation to further substantiate our findings.

6. In our work, we have utilized the complete gas pipeline dataset and three clusters of the Gas Pipeline dataset, which has been previously employed in only one other research work.

1.2 Organization of the Thesis

The remaining sections of this paper include the following: Chapter 2 covers the background knowledge necessary for understanding the research. Chapter 3 reviews the literature and outlines our proposed work. Chapter 4 details the methodology employed, while Chapter 5 discusses the experimental results. Chapter 6 concludes the paper with insights drawn from the research.

Chapter 2

Background

2.1 Industrial Control Systems (ICSs) and its Applications

ICSs are used to monitor and control CIs such as nuclear and thermal power plants, water treatment plants, power grids, gas pipeline distribution systems, etc. It consists of wireless as well as control components which help to accomplish different industrial objectives [24]. CIs are the essential systems that form the foundation of a nation's functionality and development. They are indispensable for the seamless operation and progression of society and the nation as a whole. These infrastructures encompass various domains such as healthcare, transportation, communication, and defense systems making them integral to the functioning of society.

2.1.1 SCADA System and its architecture

The SCADA system is a supervisory module within ICSs that is pivotal in monitoring and managing these critical infrastructures. SCADA systems are employed to oversee power grids, gas pipeline systems, nuclear power plants, and healthcare facilities. They operate atop hardware modules, utilizing PLC to ensure efficient and reliable performance [25]. SCADA systems contain various components such as MTU, RTU, actuators and sensors, HMI, and Data Historian [3]. Fig 2.1 illustrates the architecture of SCADA and the various components involved in it. The section below explains these components in detail.

Actuators and Sensors

The actuators and sensors are responsible for gathering the data as well as performing a range of inspections, including verifying the presence, size, and color of parts, determining whether products are full or empty, and assessing compliance with quality standards.

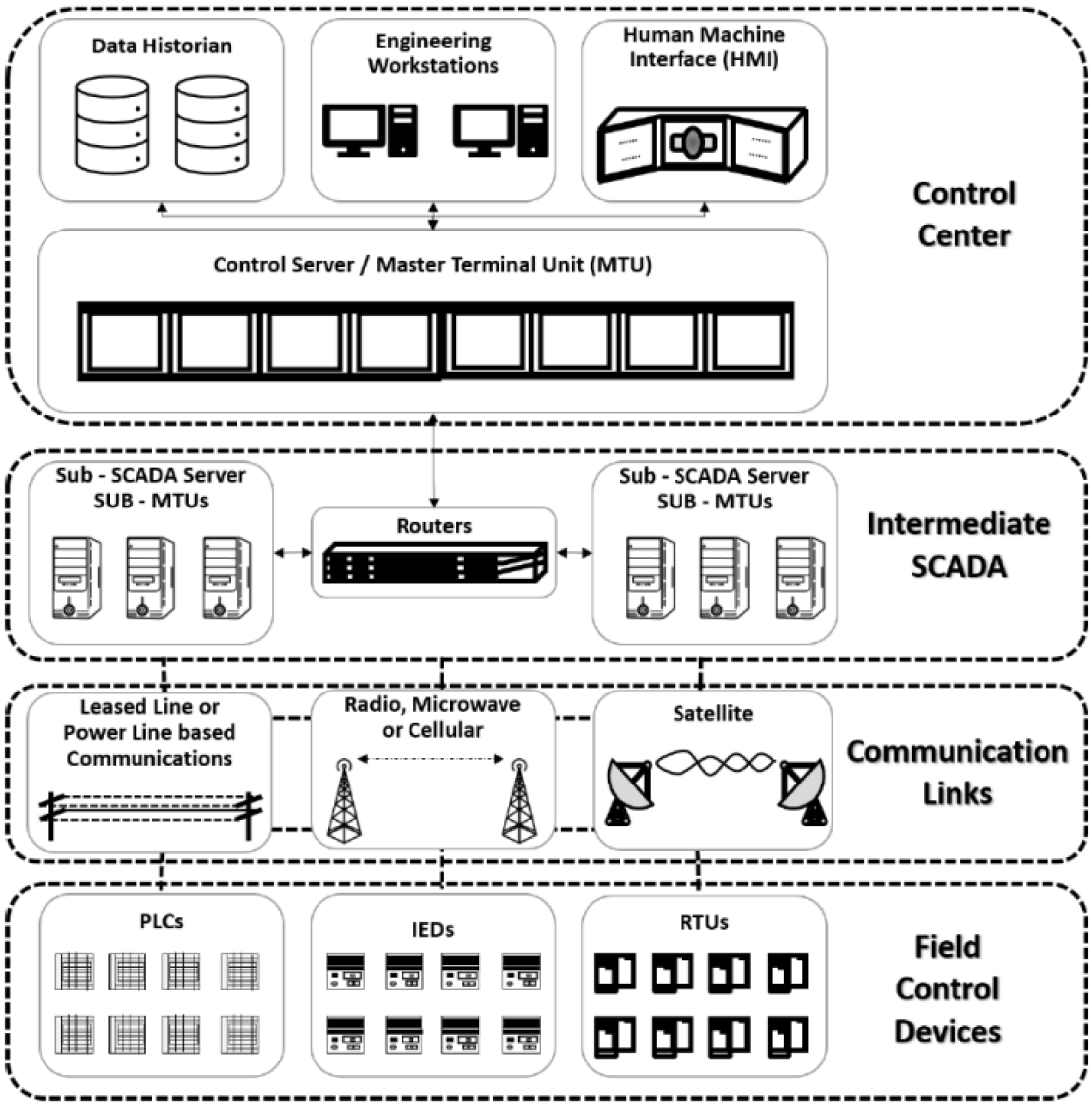


Figure 2.1: Block Diagram of SCADA Architecture [1]

Human Machine Interface (HMI)

HMI continuously monitors data and provides it to the human components. It is responsible for showcasing the information acquired in a human-comprehensible format such as graphs, text, statistics, etc [3].

Data Historian

The Data Historian is responsible for storing the data at regular intervals of time. It functions as a server at a distant location or a central database [3].

Master Terminal Unit (MTU)

It dispatches commands to field devices such as PLCs and RTUs located in remote areas, enabling the acquisition of essential data from the plant floor. Furthermore, the MTU processes this data, records vital status information in the data historian, and displays it in graphical, curve, and tabular formats on the HMI to support informed decision-making processes [5].

Remote Terminal Unit (RTU)

RTUs are responsible for the collection, and monitoring of data, and performing control functions. They act as an interface to actuators and sensors and send it to MTU for further processing. Also, PLCs and Intelligent Electronic Devices (IED) are used as an interface to actuators and sensors[3]. [5].

2.2 Feature Engineering

Feature Engineering is a process of improving the predictive capabilities of a model by improving existing features or creating new ones. The more relevant the features, the less complex the dataset and the better the model's performance. There are different techniques for feature Engineering namely:

2.2.1 Feature Scaling

This method allows a dataset's numerical values to be rescaled within a certain limit for more uniformity [26]. This provides easier and more accurate predictions. The techniques used for feature scaling are:

- Normalization: This technique takes the minimum and maximum values in a dataset and rescales all values according to that. The resultant is, every value lies between [0,1]. It is also called Min-Max scaling [27]. The mathematical representation is:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{2.1}$$

  Here, x' is the normalized value, x is the original data, x_{min} is the minimum value, and x_{max} is the maximum value.

- Standardization: This technique uses the mean and standard deviation of the dataset to rescale the values [27]. The mathematical representation is:

$$z = \frac{x - \mu}{\sigma} \tag{2.2}$$

  Here, z is the standardized value, x is the data value, μ is the mean of the dataset, and σ is the standard deviation of the dataset.

2.2.2 Dimensionality Reduction

Dimensionality reduction is reducing the overall size of the dataset by removing less relevant features from it. Reducing the features decreases the complexity of the dataset and can aid in better performance of the model. This can be performed using feature selection and feature extraction. This thesis does not cover the scope of feature extraction therefore, we will give a brief description of feature selection methods.

Feature Selection

There are different techniques that help in identifying the optimal feature subset in the dataset to improve the performance of the ML models [26]. Different techniques used for feature selection are:

- Univariate Method: This method uses statistical tests to infer the relationship between features [28].
- Feature Importance: In this method, feature importance is computed, and based on that features are selected [28].
- Correlation Matrix: In this method, the correlation between different features and the correlation of features with the target variable is used to select the feature subset [28].

2.3 Statistical Parameters

Statistics is a branch of mathematics that facilitates the visualization and comprehension of data. A thorough understanding of data is essential for obtaining credible

results and performing reliable analysis. Several parameters are employed to achieve this understanding, some of which are discussed below:

2.3.1 Standard Deviation

Standard Deviation is a measure of dispersion of values from a mean value in a dataset. This in turn helps identify the variability in a dataset [29]. The mathematical representation of the standard deviation is given below:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{2.3}$$

Here, σ is Standard Deviation, N is Total data points, μ is Mean of the data points, and x_i is Individual data point

2.3.2 Absolute Difference of Mean and Median

Here, the absolute difference between mean and median is computed in a dataset. The mathematical representation is as follows:

$$abs\_diff\_mean\_median = |\text{mean}(X) - \text{median}(X)| \tag{2.4}$$

2.3.3 Skewness

Skewness quantifies the asymmetry of a feature's probability distribution. Distribution can be deemed symmetric if it is evenly distributed around the mean. However, when the distribution strays from that to extend towards right or left it is referred to as right-skewed or left-skewed [30]. The mathematical representation of Skewness is given below:

$$S = \frac{N}{(N-1)(N-2)}\sum_{i=1}^{N}\left(\frac{x_i - \mu}{\sigma}\right)^3 \tag{2.5}$$

Here, S is Skewness, σ is Standard Deviation, N is Total data points, μ is Mean of the data points, and x_i is Individual data point

2.3.4 Kurtosis

Kurtosis is a measure of the data points residing in the tails of a distribution. The higher the value of Kurtosis, the more data points reside in the tail, and the lower

the value corresponds to fewer data points in the tail section [30]. The mathematical representation of Kurtosis is given below:

$$K = \frac{N(N+1)}{(N-1)(N-2)(N-3)} \sum_{i=1}^{N} \left(\frac{x_i - \mu}{\sigma} \right)^4 - \frac{3(N-1)^2}{(N-2)(N-3)} \tag{2.6}$$

Here, K is Kurtosis, σ is Standard Deviation, N is Total data points, μ is Mean of the data points, and x_i is Individual data point

2.4 Machine Learning and different Classifiers

ML is a technological paradigm where machines acquire the capability to perform tasks without being explicitly programmed to do so. This is achieved through the analysis of large datasets and the application of pattern recognition techniques [31]. Based on the dataset properties there can be multiple techniques that can be applied:

- Supervised Machine Learning
- Unsupervised Machine Learning
- Semi-Supervised Machine Learning

These approaches used different classifiers to make the predictions. Below we briefly explain the classifiers used in our research work.

- Decision Tree (DT): DT is a hierarchical approach to data prediction. It predicts the value of a target variable by deriving simple decision rules from the features of the data [32].
- Random Forest (RF): A RF is an ensemble learning method that constructs multiple decision tree classifiers on various sub-samples of the dataset and employs averaging to enhance predictive accuracy and mitigate over-fitting [33].
- Naive Bayes (NB): NB methods constitute a collection of supervised learning algorithms that leverage Bayes' theorem, incorporating the "naive" assumption that all features are conditionally independent of each other given the class variable [34].

- Gradient Boost (GB) GB is an iterative classifier that takes the prediction from multiple base learners to devise the final prediction. It uses the concept of gradient descent and minimizes the loss function in the model. At each iteration, the algorithm computes the gradient of the loss function with respect to the model's predictions and fits a new model to this gradient [35].

- Extreme Gradient Boost (XGB): XGB is a distributed Gradient Boost Decision Tree (GBDT) ML technique. XGB performs parallel processing on Decision Trees which makes it highly scalable [36].

2.5 Performance Metrics

Performance Metrics are different markers used to measure the performance of an ML approach. It helps the user make sense of how reliable the acquired results are, how efficiently a particular approach performs for a given dataset, comparison of different approaches, etc. There are different kinds of performance metrics providing different insights into the performance of a model. In the following sections, we discuss the performance metrics we have used:

2.5.1 Confusion Matrix

Confusion Matrix is a matrix constructed based on the combination of correct and incorrect predictions made by the classifier [37]. Fig 2.2 illustrated a confusion matrix. Below we discuss various combinations produced as a result of classification:

- True Positive (TP): The classifier correctly predicts the positive outcome [38].

- True Negative (TN): The classifier correctly predicts the negative outcome [38].

- False Positive (FP): The model incorrectly identified a negative outcome as a positive outcome [38].

- False Negative (FN): The model incorrectly identified a positive outcome as a negative outcome [38].

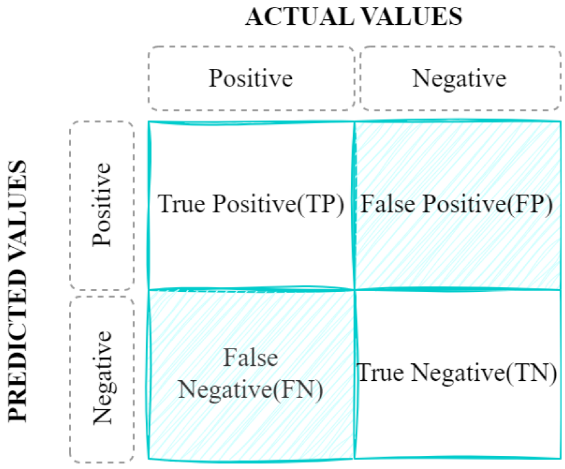


Figure 2.2: Confusion Matrix

2.5.2 Accuracy

Accuracy is a measure of total true predictions i.e. TP and TN to that of overall predictions [38]. It is mathematically represented as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.7}$$

2.5.3 Precision

Precision is a ratio of correct positive predictions w.r.t all positive predictions made by the model [38]. It is mathematically represented as:

$$Precision = \frac{TP}{TP + FP} \tag{2.8}$$

2.5.4 Recall or Sensitivity

Recall quantifies the fraction of correctly identified positive predictions out of all actual positive instances in the dataset [37]. It is mathematically represented as:

$$Recall = \frac{TP}{TP + FN} \tag{2.9}$$

2.5.5 F1-Score

F1-Score is used to evaluate the entire model. It is the harmonic mean of Precision and Recall [38]. It is mathematically represented as:

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2.10}$$

Chapter 3

Literature Review

3.1 Security threats in ICS

With the advancement in the internet and technology, there has been an increase in cyber attacks as well. Now the attack paradigm has somewhat shifted from people to infrastructures, it poses a threat to the very functioning of society and the nation as a whole. In this section, we discuss some of the cyber attacks that have taken place on ICS. The complexity of the attacks has grown over the years leaving a crippling effect on society. Below are documented attacks that have happened since 2010.
Stuxnet is a computer worm-based attack that happened in 2010. This attack was aimed at damaging the nuclear power centrifuges in Iran. It was created to specifically target the SCADA systems [3]. Night Dragon is an attack that happened on gas, oil, and petrochemical industries in 2011. The attackers penetrated the systems to get access to confidential documents and files [7].
Duqu is a malware-based attack that exploited the zero-day vulnerability in Microsoft Word in 2011. This attack targeted the control systems in SCADA systems [7]. Shamoon - Saudi Aramco and RasGas refers to the attack that happened on Saudi Aramaco, the largest energy company in the world, and Ras Gas, the second largest Liquified Natural Gas (LNG) company in 2012. The attack was implemented using a Malware called Shamoon that removed data from infected devices [7], [39].
Target Store attack in 2013 targeted the HVAC control systems in the target stores. The attackers used malicious credit card stealing software to steal credit card information of customers at Targets' chain of stores. Close to 70 million customers were affected in this incident [40]. New York Dam is an attack that happened on Bowman Dam in 2013. Here the attackers got access to the SCADA system which was under status monitoring only at the time [3]. Havex is a malware that targeted specifically SCADA systems in 2013. Godzilla Attack in 2014 [39], [3] was a malware based attack that impacted various ICS. German Steel Mill attack happened in 2014

on a steel plant in Germany. The hackers acquired access to the steel plant via its business network. They tempered with the control system resulting in the furnace not shutting down the way it should have. As a result of this, the company faced a huge loss [39], [3].

BlackEnergy was a malware-based attack in 2014. Its target was Ukraines electrical system which found its mark leading to a blackout in various areas. It also affected the HMI in SCADA systems [40]. The Ukraine Power Grid attack happened in 2015. The attackers used BlackEnergy3 malware on SCADA systems leading to a power shut-off in 30 substations leaving nearly a quarter million Ukranians in the dark [7], [3], [40]. "Kemuri" Water Company was an attack that happened on the SCADA system of a water company in 2016. The name of the water company was kept confidential and named "Kemuri" for reference purposes. In this attack, the attackers targeted the water district valve and flow controls affecting the chemicals going into the water. This attack was disclosed by Verizon Security Solutions [40]. Operation Ghoul was a spear-phishing and malware-based attack that happened in 2016. The attackers here targeted various computer systems in the SCADA network [3].

CRASHOVERRIDE was a malware-based attack that happened in 2017. This attack was specifically targeting the power grids resulting in the de-energization of substations [40]. Dragonfly 2.0 was a phishing and malware-based attack that happened in 2017. It targeted the software used to send commands to energy equipment leading to compromised equipment in various energy companies [7]. Triton (Trisis) was a malware-based attack that happened in 2017. It targeted the safety system of Triconex Safety Instrumented System (SIS) of a petrochemical company in Saudi Arabia [41]. SamSam was a ransomware attack that happened in 2018. It targeted the Department of transportation in Atlanta leading to website outages, blocked tickets, ticket processing, etc [40]. Saipem Company in Italy is an Oil and Gas company. It was affected by a malware attack in 2018 [7]. Iran APT33 (Elfin) Attack was a malware and spear-phishing attack that happened in 2019. It affected various ICSs such as aviation, energy, etc [42]. Norsk Hydro Attack was a ransomware(LockerGoga) attack that happened in 2019. It targeted the ICS for Aluminum manufacturing in Norway [43]. Oldsmar Water Treatment Plant attack is an unauthorized remote

access that happened in 2021. The attack happened at a water treatment facility in Florida, USA. The hacker attempted to alter the chemical levels in the water supply leading to what could have been a life-threatening situation [44].

Colonial Pipeline attack was a ransomware(Darkside) attack that happened in 2021. The attack affected the Fuel Supply chain of east USA by disruptions in fuel supply leading to a shortage of fuel [45]. JBS S.A. is the worlds' largest meat processing company that faced a ransomware (REvil) attack in 2021. The attack targeted the meat production facilities leading to disruption in the global food supply chain. The attack disrupted meat production facilities in the USA, Canada, and Australia, impacting the global food supply chain [46]. Table 3.1 shows the attacks that have happened from 2010-2021 on ICSs.

Table 3.1: Details of attacks launched on ICS from 2010-2021

Ref	**Name of Attack**	**Year**	**Location**	**ICS/SCADA System Targeted**	**Attack Vector**	**Affected Population/ Industry**
[3]	Stuxnet	2010	Iran	Siemens Step7 software on PLCs	Malware introduced via infected USB drives	Nuclear enrichment facilities
[7]	Night Dragon	2011	Global (primarily USA)	Various systems in oil, gas, petrochemical industries	Spear-phishing and remote access tools	Oil, gas, and petrochemical industries
[7]	Duqu	2011	Global	SCADA systems	Malware exploiting zero-day vulnerability in Microsoft Word	Various industries
[39], [3]	South Houston Water Treatment Plant Hack	2011	USA (Texas)	SCADA system of water treatment plant	Default password exploitation	Municipal water services
[7], [39]	Shamoon	2012	Saudi Arabia	Ras Gas a Liquified Natural Gas (LNG) company	Malware (Disttrack)	Service disruption and data loss
[3]	Auto Manufacturer Hack	2012	USA	Various systems in auto manufacturing	Spear-phishing and malware	Auto manufacturing
[40]	Target Store Attack	2013	USA	HVAC control systems in retail stores	Malware targeting point-of-sale systems	Retail (Target stores, 70 million customers)
[3]	New York Dam	2013	USA (New York)	SCADA system of Bowman Dam	Unauthorized remote access	Water management
[39], [3]	Havex	2013	Various countries	Various SCADA systems	Malware distributed via watering hole attacks and phishing emails	Various industries
[47]	Godzilla Attack	2014	USA	Various ICS systems	Malware	Various industries
[39], [3]	German Steel Mill Attack	2014	Germany	Industrial control systems	Spear-phishing emails and malware	Steel production
[40]	BlackEnergy	2014	Ukraine	Electrical and Power grids	Malware (BlackEnergy) and remote access tools	Power distribution (230,000 customers)
[7], [3], [40]	BlackEnergy3	2015	Ukraine	Electrical and Power grids	Malware (BlackEnergy) and remote access tools	Shut-off of 30 substations
[40]	Kemuri	2016	Global	Water Company	Not made public	Water company facilities

[3]	Operation Ghoul	2016	United Arab Emirates (UAE)	Computer systems in SCADA network	Spear-phishing and malware-based attack	Industrial, engineering, manufacturing, and transportation sectors.
[40]	Industroyer/CrashOverride	2016	Ukraine	Electric power transmission networks	Malware specifically designed for power grid systems	Power distribution
[7]	Dragonfly 2.0	2017	United States and Europe	Energy sector	Phishing and malware-based attack	Data in the form of sensitive information compromised
[41]	Triton/Trisis	2017	Saudi Arabia	Safety instrumented systems (SIS) in industrial plants	Malware targeting Schneider Electric's Triconex SIS	Petrochemical industry
[40]	SamSam	2018	USA (Atlanta)	Department of transportation	Ransomware	Website outages, blocked tickets, ticket processing
[7]	Saipem Company	2018	Italy	Oil and Gas company	Malware attack	Significant disruption by wiping data
[42]	APT33 (Elfin) Attack	2019	Iran	Aviation Sector	Malware and spear-phishing attack	Operation disruption and data loss
[43]	LockerGoga	2019	Norway	Aluminum manufacturing industry	Ransomware	Operation disruption and files being encrypted
[44]	Oldsmar Water Treatment	2021	USA (Florida)	SCADA system of water treatment plant	Unauthorized remote access via TeamViewer	Municipal water services
[45]	Colonial Pipeline attack	2021	USA	Fuel supply chain	Ransomware (Darkside)	Disruptions in fuel supply leading to a shortage of fuel.
[46]	Revil attack	2021	USA, Canada, and Australia	Food supply chain	Ransomware	Impacted JBS S.A.

3.2 Use of Machine Learning in IDS for ICS

As shown in the previous section, the advent of technology and reliance on machines has brought about a new array of cyber attacks. This paradigm of cyber attacks has motivated researchers to try and combat such attacks with the use of ML in IDSs. Below are some of the works carried out using ML in IDS for ICSs. Various supervised learning techniques have been used to detect attacks in SCADA systems using various publicly available datasets.Guamei et al. [23] have worked on detecting attacks on smart grids. The researchers have reduced the feature space using correlation-based feature selection (CFS). The reduced feature set is then used in an Instance-Based Learning (IBL) algorithm which classifies normal and cyber-attacks followed by a 10-fold cross-validation. Alimi et al. [48] researched the use of ML in detecting intrusions in a power system network. They have used five ML classifiers i.e. K-nearest neighbors (kNN), DT, NB, RF, and AdaBoost on a simulated voltage dataset. According to their findings, they found AdaBoost to perform the best in terms of accuracy and kNN to perform the best in terms of Training time.

Upadhyay et al. [49] have used ML classifiers in the attack detection in Smart Grids. They have used a novel feature selection algorithm called Gradient Boosting Feature Selection (GBFS) using the Weighted Feature Importance (WFI) extraction technique before the actual classification. They have focused on using different DT-based classifiers for detection and accessing them on various performance parameters such as accuracy and execution time. Alimi et al. [6] created a hybrid approach using a Support Vector Machine and a Multilayer Perceptron Neural Network (SVMNN) algorithm. They tested out their hybrid algorithm on the power system networks dataset attaining an increased performance to other schemes.

Arora et al. [50] have explored the performance of different ML classifiers such as RF, Support Vector Machine (SVM), DT, Artificial Neural Networks (ANN), K-Nearest Neighbors (KNN) and NB for detecting an attack on ICSs using a SCADA attack dataset. According to their research, RF performs the best among the classifiers used. Turkoz et al. [51] have worked on improving anomaly detection with the use of the SVM classifier. They have created a Generalized Support Vector Data Description (GSVDD) which uses a hypersphere for class separation. They also introduced a generalized Bayesian framework which highlights the relationship between modes

rather than just using the prior information.

Alhaidari et al. [52] improves the SCADA system framework against Distributed Denial of Service (DDoS) using ML classifiers. The research is done using J48, NB, and RF on the KDDCup'99 dataset. Based on their findings, RF attained a near-perfect accuracy becoming the best performer in the collection. Robles et al. [53] research work was based on proposing a real-time anomaly intrusion detection system for water supply systems. They have created a novel dataset from conducting the attacks on the testbed. They have used five ML classifiers to detect these attacks: K-Nearest Neighbour, SVM, DT, NB, and Multilayer Perceptron. They have judged the performance of classifiers based on online and offline training which concluded kNN and SVM as best performers.

Shitharth et al. [54] researched on detecting and classifying intrusion in the SCADA system based on optimization. The researchers have proposed Intrusion Weighted Particle-based Cuckoo Search Optimization (IWP-CSO) and Hierarchical Neuron Architecture based Neural Network (HNA-NN) techniques. Here IWP-CSO algorithm is used to select the best attributes which are then classified using the HNA-NN algorithm on the ADFA-LD dataset.

3.3 Related Work based on Gas Pipeline systems

As discussed in the last section about the emergence of ML in IDSs tailored for ICSs we can conclude that ML can prove pivotal for creating and efficiently using the IDSs. This section specifically targets the use of ML in IDS based on the SCADA system Gas-Pipleine dataset. Below are the research works done using the Gas-Pipeline dataset. Table 3.2 shows the details of the attacks mentioned below.

Khan et al. [55] performed a binary and multi-class classification on the subset of the Gas Pipeline dataset i.e. Command, Function, and Response datasets. They used classification algorithms such as NB, PART, and RF to classify the attacks. According to their acquired results, the researchers concluded that RF classifier performed the best.

Demertzis et al. [56] performed Spiking One Class Anomaly Detection Framework using a novel One Class Classification algorithm. This framework was evaluated on three distinct datasets: an electrical, water tower, and gas pipeline dataset. They

demonstrated that the eSNN (evolving Spiking Neural Network) algorithm performed superior to SVM and CD/CPE (Combining Density and Class Probability Estimation) in the obtained results.

Tammy et al. [57] have used four supervised learning techniques on the "10% Random Sample Gas Pipeline Dataset" to detect attacks on SCADA systems. In their experimentation, they have used NB, SVM, Trees J48, and RF ML models. According to their work, RF has performed the best out of all however, it also takes the longest time to execute.

Perez et al. [58] have worked on using ML for IDS in ICSs. For their research, they have used the gas pipeline system dataset. They have used a multi-step process that involves using four techniques for data estimation as well as normalization of the dataset followed by the use of two ML classifiers namely RF and SVM. Based on their findings RF performed better than SVM.

Khan et al. [4] have worked towards improving anomaly detection for an ICS by exploiting the communication patterns in ICS environments. Their methodology includes extensive pre-processing and feature selection on the dataset using Principal Component Analysis (PCA), Canonical Correlation Analysis (CCA), and Independent Component Analysis (ICA) before prediction. They developed a hybrid model that balances the dataset through an edited nearest-neighbor rule in K-Nearest Neighbors (KNN). They also construct a signature database using a Bloom filter during periods devoid of abnormalities.

Prisco et al. [59] employed ML techniques to detect attacks on SCADA systems, specifically utilizing the One-Class Support Vector Machines (OCSVM) algorithm. This method involves training with only normal data to identify anomalies, followed by cross-validation. Data pre-processing and normalization were performed using the Radial Basis Function (RBF) kernel. Their findings indicate a complete identification of attack data. Additionally, their approach demonstrated superior response times compared to other existing methods.

Al-Asiri et al. [60] examined Intrusion Detection Systems (IDS) utilizing physical metrics within SCADA systems. Their case study employed a gas-pipeline dataset to assess the impact of incorporating physical metrics into an IDS. They utilized the DT classifier in their analysis. Their findings indicate that integrating physical

metrics with additional parameters enhances detection capabilities, whereas relying solely on network traffic data may not produce similarly effective results.

Paramkusem et al. [61] conducted a study on the classification and detection of malicious command and response packets within SCADA networks utilizing a Big Data framework. They employed the k-means clustering algorithm to analyze packet history and attribute differences. Additionally, they utilized the RF classifier to detect attacks, achieving improved results compared to previous work." Choubineh et al. [21] used cost-sensitive learning and Fisher's discriminant analysis (FDA) for dimensionality reduction followed by five classifiers namely: HoeffdingTree, RandomTree, OneR, NB, and BayesNet. In their work they have tried to address the issue of class imbalance found in SCADA system datasets.

Table 3.2: Research Work done using Machine Learning for attack detection in Gas Pipeline dataset.

Ref.	Normalization	Clustering	Feature Selection	Models	Dataset	Research Gaps
[58]	Yes, Min-Max, Mean Standard Deviation	No	Not Done	SVM, RF	Entire Gas Pipeline Dataset	The running time of the model isn't taken into account. Use of entire Feature space. Broader range of ML algorithms can be explored.
[55]	Not Mentioned	Yes	Not Mentioned	Naïve Bayes, PART, RF	Clustered dataset: Command, Response, Function	The running time of the model isn't taken into account. Use of entire Feature space. No normalization performed on datasets. Scope for increasing accuracy in Command and Response dataset.
[61]	No	Yes	1. Current-Previous value 2. k-means(k=2) algorithm	RF	Entire Gas Pipeline Dataset, Attribute Difference Dataset, 2-means Dataset	The running time of the model isn't taken into account. No normalization performed on datasets. Broader range of ML algorithms can be explored.
[57]	No	No	No	RF, SVM, NB, J48	10% Random Sample Gas Pipeline Dataset	No normalization performed on datasets. No Feature Selection methods employed. The running time of the model isn't taken into account.
[4]	Yes, Standardization	Yes	Yes using PCA, CCA, and ICA	KNN, RF, AdaBoost, Net (MLP), Quadratic Discriminant Analysis	Entire Gas Pipeline Dataset	The running time of the model isn't taken into account. Other Feature Selection techniques can be explored.
[59]	Yes, Min-Max Normalization, RBF kernel	No	No	One-Class Support Vector Machines (OCSVM)	Entire Gas Pipeline Dataset	No Feature Selection methods employed. The response time of the model isn't taken into account. Broader range of ML algorithms can be explored.
[60]	No	No	No	DT, 10-fold Cross Validation	Entire Gas Pipeline Dataset	No normalization performed on datasets. No Feature Selection methods employed. The response time of the model isn't taken into account. Broader range of ML algorithms can be explored.
[56]	Yes	No	No	OCC-eSNN, OCC-SVM, OCC-CD/CPE	Entire Gas Pipeline Dataset	No Feature Selection methods employed. The response time of the model isn't taken into account. Broader range of ML algorithms can be explored.
[21]	No	No	Yes, using FDA and Cost Matrix	HoeffdingTree, RandomTree, OneR, NaiveBayes, and BayesNet	Entire Gas Pipeline Dataset	The response time of the model isn't taken into account. Broader range of ML algorithms can be explored.

3.4 Research Gaps, Novelty, & Proposed work

As shown in the previous sections there has been work done using Gas-Pipeline Dataset created by [22] in IDS for ICSs. However, there are certain shortcomings in the present efforts:

- Many ML-based IDSs use entire datasets, resulting in lengthy response times and the inclusion of irrelevant features. Feature selection techniques are underutilized in addressing this issue.
- The performance of Feature Selection techniques can be improved further by using a feature ranking mechanism. However, that also remains an unexplored area.
- In the event of a cyber-attack fast detection leading to a fast response is an integral component for any IDS. Yet response time optimization is often overlooked in ML classifiers.
- To the best of our knowledge, we have found just one research work pertaining to the use of clusters devised from the gas pipeline dataset.
- Publicly available datasets often suffer from imbalance, impacting model performance.

To address these issues, we focused on reducing the response time of ML classifiers using two feature ranking approaches based on the work done in [23] to hierarchically arrange the features as well as reduce the dataset dimensionality by using the Feature Selection methods. In our work, we have introduced a novel Feature Selection algorithm called Selective Promising Feature Selection (SPFS) and compared it with the Forward Feature Selection (FFS) algorithm used by [20]. We validated classifier performance using 10-fold cross-validation to select the optimal performer based on total execution time, accuracy, F1 score, and other performance metrics. Through this research, we make the following contributions:

1. We devised a methodology capable of distinguishing attack scenarios from normal operations while simultaneously enhancing performance metrics.

2. We assessed two ranking mechanisms, namely the Statistical Parameter (SP) approach and the Weighted Feature Importance (WFI) method, applied to three clusters of the Gas Pipeline dataset, both with and without preprocessing. This assessment yielded a hierarchically organized set of features, ranked by their significance within the dataset.

3. We introduced an innovative Feature Selection algorithm, the Selective Promising Feature Selection (SPFS), which selectively retains only those features that enhance classifier performance. This novel algorithm was benchmarked against the Forward Feature Selection (FFS) algorithm using multiple ML classifiers on three clusters of the Gas Pipeline datasets, to identify the most effective feature selection methodology.

4. We conducted a comparative analysis of various ML classifiers in each phase to ascertain the optimal combination for anomaly detection. The study incorporated 10-fold Cross-Validation to further substantiate our findings.

5. In our work, we have utilized the three clusters of the Gas Pipeline dataset, which has been previously employed in only one other research work.

As our primary objective is to reduce the response time in IDSs for ICSs we have experimented with the clusters of gas pipeline dataset as well as the whole dataset to understand the time complexity of the classifiers w.r.t. the dataset in consideration. Therefore, in our work to narrow down the dimensions of the dataset we opted for two methods:

1. Using the deduced clusters [61] from original dataset.

2. Narrowing down the dimensions further with the help of Feature Ranking and Feature Selection algorithms. Therefore, the dataset have shrank in both rows as well as columns.

There are several benefits to using the dataset clusters as opposed to the entire dataset such as:

- Focused Analysis: Analyzing the clusters allow a more focused understanding of the structure of gas pipeline. Hence, the predictions can be more targeted and specific.

- Enhanced Anomaly Detection: It is easier to identify the localized patterns in clusters hence, even a slight deviation from normal behavior can be captured which might get overlooked in a bigger dataset.

- Improved Data Management: As our primary target is to reduce the response time in an IDS, smaller datasets aid towards that goal. It allows for faster processing, lesser computational resources as well as independent computation on the clusters rather than using a huge dataset.

- Enhanced Monitoring and Control: As each cluster pertains to specific monitoring and control strategies inside the pipeline. Therefore, they provide a very precise view of how things should proceed ideally. The segregation of different control parameters makes it easier to monitor them.

With our research we aim to address the following questions:

- Research Question 1: The impact of pre-processing, feature ranking, and feature selection on ML classifier performance.

- Research Question 2: Accuracy of the current approach in detecting anomalies.

- Research Question 3: The advantages of our proposed methods compared to existing techniques.

- Research Question 4: Based on the comparative analysis, what is the best approach to be followed for IDS targeting time-sensitive and performance-sensitive systems?

Chapter 4

Methodology

The research gaps identified in the previous sections helped us deduce that a new efficient way of handling cyber-attacks for CI is minimizing the response time by using ML. Therefore, in this section, we discuss the overall methodology, proposed ranking mechanisms, and proposed feature selection algorithm.

4.1 Dataset Description

Mississippi State University's in-house SCADA lab features a gas pipeline system utilized for collecting data pertinent to cyber attack research, illustrated in Fig 4.1. The system comprises three primary components: sensors and actuators, a communication network, and supervisory control, as depicted in Fig 4.1. This segment provides an overview of the testbed components and outlines the methodologies employed for dataset acquisition using this testbed. For more detailed insights, please refer to [2]. The gas pipeline includes two actuators, a pump and solenoid, and a pressure sensor. These actuators are responsible for maintaining pressure levels regulated by the supervisory control system. The gas pipeline operates in three primary system modes: automatic, manual, and off. In automated mode, two schemes are utilized: the pump mode, which toggles the pump to regulate pressure in the pipe at the designated set point, and the solenoid mode, where a relief valve, controlled by a solenoid, modulates pressure levels. Both the pump and solenoid modes employ a Proportional-Integral-Derivative (PID) control scheme. In manual mode, operators are required to manually oversee the pump and solenoid operations.
The next component is the communication network in which the protocol used is serial Modbus RTU. Modbus packets include a header and a payload. For Modbus over a Serial Line, a packet includes a device address, function code, payload, and a cyclic 21 redundancy code (CRC) or linear redundancy code (LRC).

The gas pipeline cyber attack database was initially developed by Tommy Morris

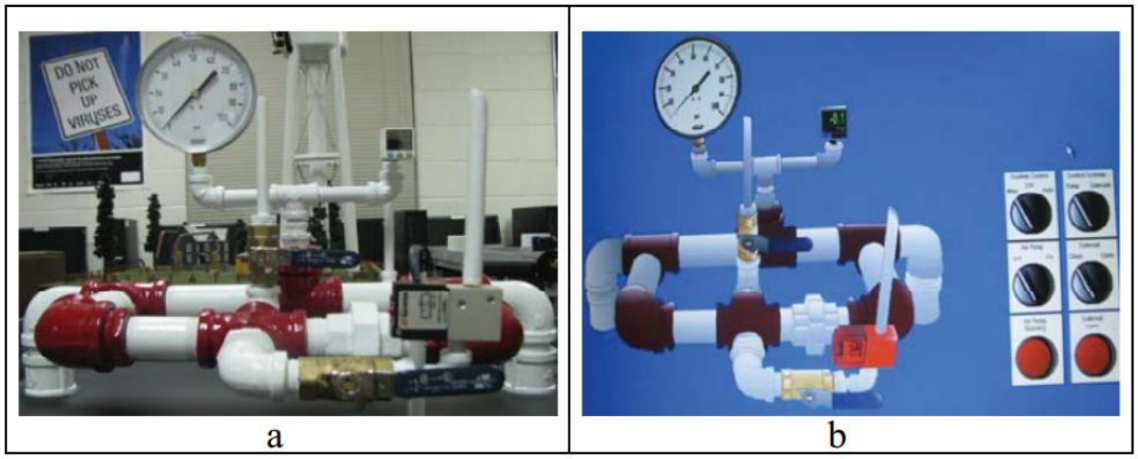


Figure 4.1: Gas Pipeline Testbed [2]

and Wei Gao in 2012 [22]. However, upon examination of the original dataset, significant flaws were identified, rendering it unsuitable for ML research [2]. Subsequently, Ian Turnipseed of Mississippi State University compiled a more realistic cyber attack gas pipeline dataset, known as the "new gas pipeline dataset," and made it publicly accessible for the research community. In our study, we utilize the new gas pipeline dataset.

The "new gas pipeline" dataset is available in two formats: a comma-separated value (CSV) text file and an Attribute Relationship File Format (ARFF). The ARFF dataset was specifically formatted to align with Waikato Environment for Knowledge Analysis (WEKA), a tool widely used by researchers worldwide for testing ML algorithms [62]. Although we do not employ WEKA, we preprocessed the ARFF dataset to prepare it for training various classification models. Each record or instance in the dataset corresponds to one packet delivered to either the MTU or the RTU, containing network traffic and payload information. SCADA systems possess fixed network topologies, and transactions between components are repetitive and regular, in contrast to the dynamic and irregular nature of IT network traffic data. The payload information in the dataset reveals details about the gas pipeline's state, settings, and parameters, crucial for detecting anomalies resulting from system malfunctions or malicious activities by cyber attackers. The dataset comprises a total of 274,628 instances or rows, with each row containing twenty columns, commonly

referred to as features. These features are summarized with a brief description in Table 4.1.

Feature Id	Feature Symbol	Type	Description
1	address	real	The address of the slave device. Each slave device in the Modbus is assigned an 8-bit address to identify the slave device the master is communicating to and from.
2	function	real	The function codes are primarily used in the gas pipeline to indicate a read (0x03) and write commands (0x16). But there are possibilities of a total of 256 such commands. A denial of service attack can be launched by setting a function code of 0x08 which corresponds to diagnostic mode where the device would be always in listening mode.
3	length	real	Length of the Modbus frame. This is fixed for each command and response frame. Frames that are not of specific length can be easily detected as attacks.
4	setpoint	real	This value controls the pressure in the gas pipeline.
5	gain	real	Gain parameter of the PID controller.
6	reset rate	real	Reset rate parameter of the PID controller.
7	deadband	real	Deadband parameter of the PID controller.

Feature Id	Feature Symbol	Type	Description
8	cycle time	real	Cycle time parameter of the PID controller.
9	rate	real	Rate parameter of the PID controller.
10	system mode	{0,1,2}	Controls the duty cycle of the system. The following modes are valid: 0 – Off, 1 – Manual, 2 – Automatic.
11	control scheme	{0,1}	The control scheme in the gas pipeline determines whether the system will be controlled by the pump or by the solenoid. There are two schemes: 0 – Pump, 1 – Solenoid.
12	pump	{0,1}	This is the state of the pump when system mode is set to manual. There are two possible values: 0 – Off, 1 – On.
13	solenoid	{0,1}	This represents the state of the solenoid valve. There are two possible values: 0 – Closed, 1 – Open.
14	pressure	real	The current pressure measurement from the gas pipeline.
15	crc rate	real	The Cyclic Redundancy Check (CRC) allows the system to check errors within a Modbus frame.
16	command response	{0,1}	This value allows the IDS to learn about the command and response frame. Two possible values: 0 – Response, 1 – Command.
17	time	real	Timestamp of the instance.
18	binary result	{0,1}	Labels to indicate either attack (1) or normal (0) instance.

Feature Id	Feature Symbol	Type	Description

Table 4.1: Description of features from New Gas Pipeline dataset

This section details the derivation, size, number of fields, and output labels for three clustered datasets derived from an original SCADA system dataset. The original dataset, containing 274,628 data instances (rows) and 18 features (columns), was pre-processed and divided into three sub-datasets based on dataset types: Command (C), Function (F), and Response (R). Each sub-dataset then underwent further cleaning to remove missing values, resulting in the reduced datasets: Command Dataset, Function Dataset, and Response Dataset.

The following code snippet filters the original dataset based on specific criteria to create the three clustered datasets: Command, Response, and Function. The code is structured to filter the dataset according to the type specified by the 'filetype' variable. The code block for creating the three sub-datasets is as shown below: In the

```
if (filetype=='C'):
    ##### Command Data Set Section 3.2, Paramkeusem (2018) #####
    data2=data[(data['function']==16) & (data['length']==90)]
elif (filetype=='R'):
    ##### Response Data Set Section 3.3, Paramkeusem (2018) #####
    data2=data[(data['function']==3) & (data['length']==46)]
else:
    ##### Rest of the data ######
    data2=data[~((data['function']==16) & (data['length']==90)) &
~((data['function']==3) & (data['length']==46))]
```

Figure 4.2: Code block for creating the three sub-datasets

first condition, "if (filetype=='C')", the code filters the dataset for command data by selecting rows where the function code is 16 and the Modbus frame length is 90, as described in Section 3.2 of [61]. In the second condition, "elif (filetype=='R')", the code filters the dataset for response data by selecting rows where the function code is 3 and the Modbus frame length is 46, as detailed in Section 3.3 of [61]. Finally, the else condition captures all remaining data that do not meet the criteria for the command or response datasets, ensuring that the function dataset includes all other instances. This approach ensures that the data is accurately segmented according to

the specified criteria for each dataset type. The features of the processed command, response, and function datasets are described in more detail below.

Command Dataset Command Dataset was derived by filtering the original dataset for instances where the dataset type is 'C'. According to Section 3.2 of [61], this involved selecting data where the function code equals 16 and the Modbus frame length equals 90. Initially, this dataset contained 64,100 rows and 20 columns. After removing missing values, the dataset size remained 64,100 rows but reduced to 17 columns. Following further cleaning, the final dataset comprised 64,100 rows and 14 columns. The input features for this dataset included setpoint, gain, reset_rate, deadband, cycle_time, rate, system_mode, control_scheme, pump, solenoid, pressure_measurement, crc_rate, and time. The output label for the dataset is binary_result, and the cleaned data was saved as NewGasFilteredCommand.csv.

Response Dataset The Response Dataset was derived by filtering the original dataset for instances where the dataset type is 'R'. As outlined in Section 3.3 of [61], this involved selecting data where the function code equals 3 and the Modbus frame length equals 46. Initially, this dataset contained 68,848 rows and 20 columns. After further processing, the dataset size remained at 68,848 rows but was reduced to 14 columns. The input features for this dataset included setpoint, gain, reset_rate, deadband, cycle_time, rate, system_mode, control_scheme, pump, solenoid, pressure_measurement, crc_rate, and time. The output label for the dataset is binary_result, and the cleaned data was saved as NewGasFilteredResponse.csv.

Function Dataset The Function Dataset was derived by excluding data where the function code equals 16 and length equals 90, and where the function code equals 3 and length equals 46 (i.e. excluding the command and response data). Initially, this dataset contained 141,680 rows and 20 columns. After further processing, the dataset size remained at 141,680 rows but was reduced to 18 columns. The input features for this dataset included address, function, length, setpoint, gain, reset_rate, deadband, cycle_time, rate, system_mode, control_scheme, pump, solenoid, pressure_measurement, crc_rate, command and time. The output label for the dataset is binary_result, and the cleaned data was saved as NewGasFilteredFunction.csv.

4.2 Proposed Framework

This section will provide you with a basic overview of the methodology used to address the research problem in consideration before diving into details. A series of experiments were performed to identify the best possible parameters and classifiers to attain an optimized result. We have done our research analysis as three-part experiments. The three experiments are implemented using the Gas Pipeline dataset as well as the clusters of the Gas Pipeline dataset [61] namely Command, Function, and Response. Through the experimentation on the Gas Pipeline dataset, we acquired promising ML classifiers that can be used for the three clusters as well. Fig 4.3 shows the list of experiments performed and their respective deliverables. Here each experiment serves as a stepping stone for the next experiment. The following provides the experiments and sub experiments performed:

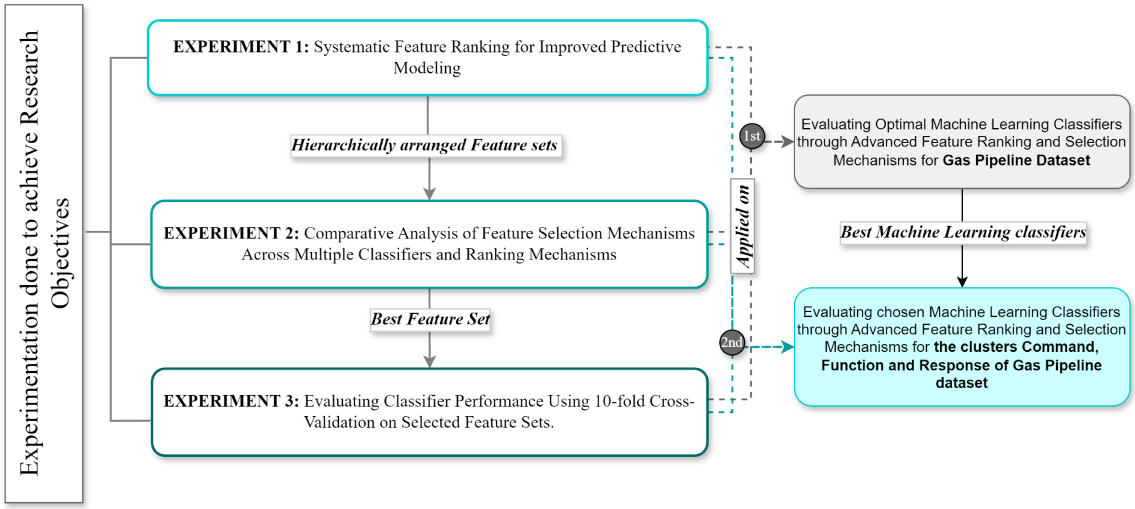


Figure 4.3: Experiments performed and their deliverables

1. Systematic feature ranking for improved predictive modeling using SP and WFI.

2. Comparative analysis of feature selection mechanisms across multiple classifiers and ranking mechanisms using FFS and SPFS.

3. Evaluating Classifier performance using cross-validation on selected feature sets using RF, GB, and XGB.

4. Evaluating optimal ML classifiers for the Gas Pipeline Dataset Analysis using SP and SPFS algorithm on five ML classifiers namely DT, RF, GB, XGB, and NB.

5. A comparative analysis of the implementation of the proposed novel approach SP-SPFS on the gas pipeline dataset against the approach without feature ranking and feature selection.

6. Evaluating chosen ML Classifiers through advanced feature ranking and selection mechanisms for the clusters Command, Function, and Response of the Gas Pipeline dataset.

Experiments represent the overall structure of the research whereas, each experiment contains a set of steps carried out in a sequential manner consisting of the following phases: pre-processing, feature ranking, feature selection, cross-validation, and prediction. Figure 4.4 provides more details regarding the overall process.

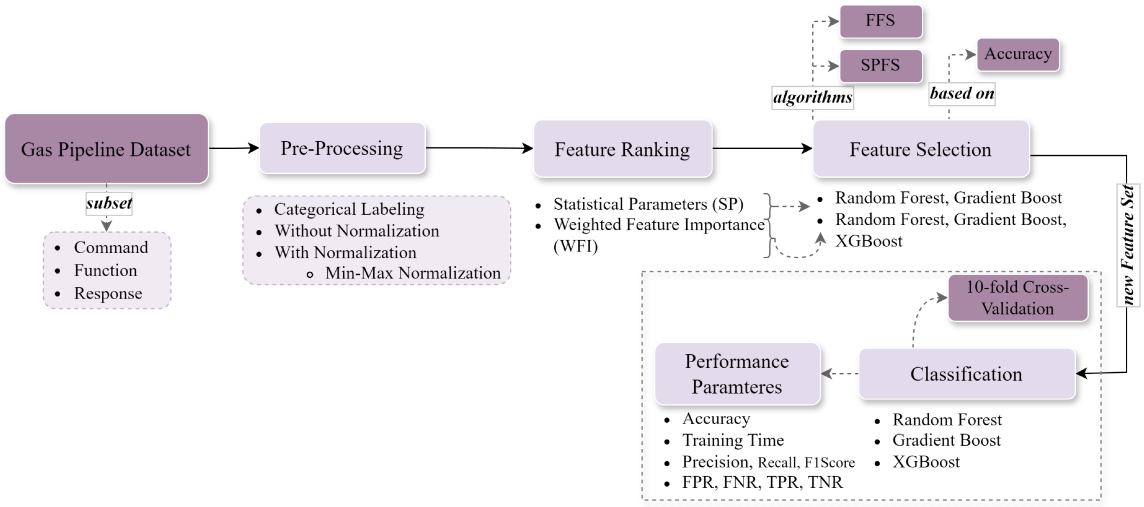


Figure 4.4: Block diagram of the Methodology

4.3 Pre-Processing

To prepare the dataset for further processing we performed the data pre-processing as follows:

- We separated all the numerical features from the dataset.

- The numerical features are then scaled using the "RobustScaler" from the scikit-learn library to account for the presence of outliers.
- The acquired numerical features are now transformed using the "MinMaxScaler" from the scikit-learn library in the range of 0 to 1.

4.4 Choosing the Machine Learning Classifier for Gas Pipeline dataset

In this experiment, we used the pre-processed as well as non-pre-processed dataset to determine the best ML classifier. We used the entire gas pipeline dataset for the classification process. As shown in Fig. 4.5 we used five classifiers to determine the top three performing classifiers. The classifiers used were: DT, RF, GB, XGB, and NB.

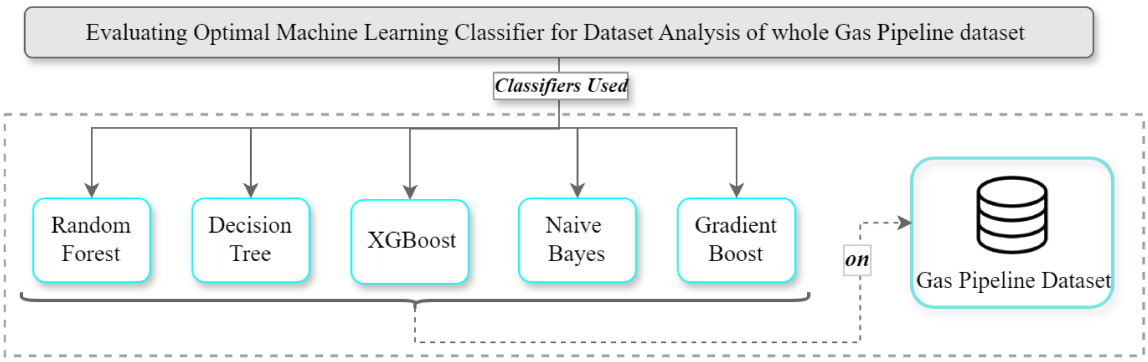


Figure 4.5: Classifiers and Dataset used for Gas Pipeline dataset

4.5 Comparative Analysis of Feature Selection and Non-Feature Selection Techniques on the Gas Pipeline Dataset

In this experiment, our objective was to investigate the influence of feature selection on anomaly detection efficacy within an IDS tailored for ICSs. We have utilized the gas pipeline dataset with a novel approach called SP-SPFS which utilizes the SP technique for feature ranking and the SPFS algorithm for feature selection. The outcomes derived from this approach were subsequently compared to those obtained from scenarios devoid of feature ranking and selection in the gas pipeline dataset. We have used the same ML classifiers mentioned in the section above followed by a

10-fold cross-validation to validate the results. Fig 4.6 illustrates the methodology used for the experiments performed

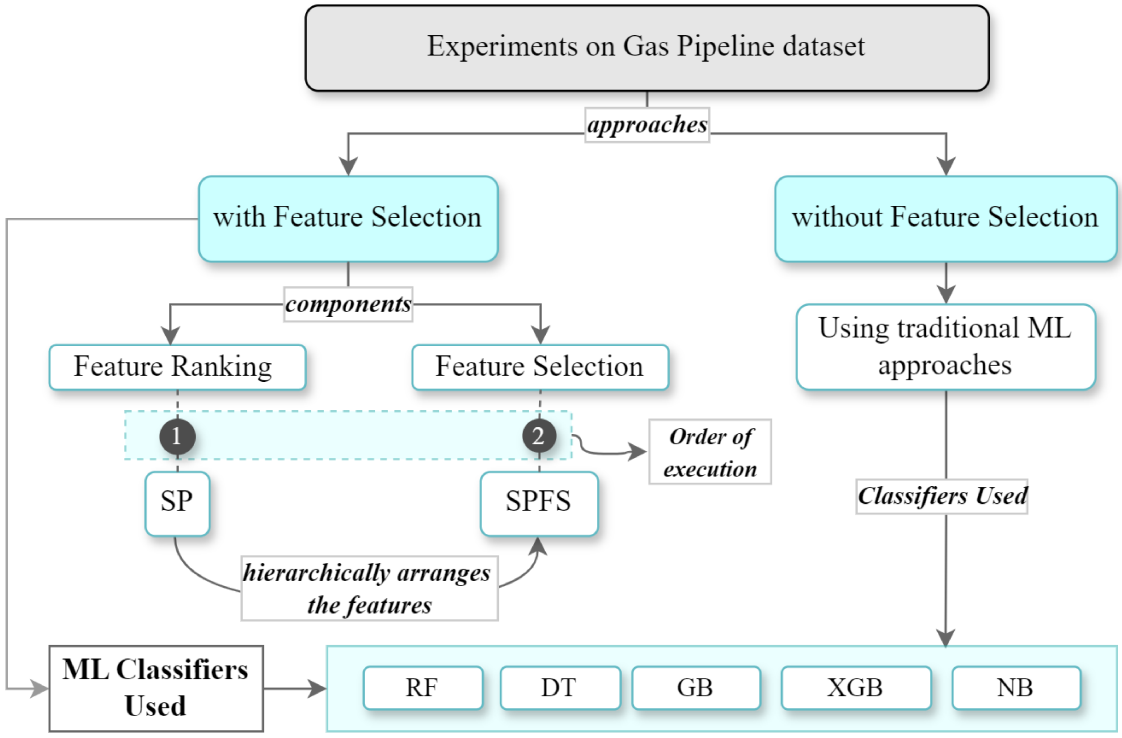


Figure 4.6: Block diagram of methodology for experimentation performed on the Gas Pipeline dataset

4.6 Feature Ranking

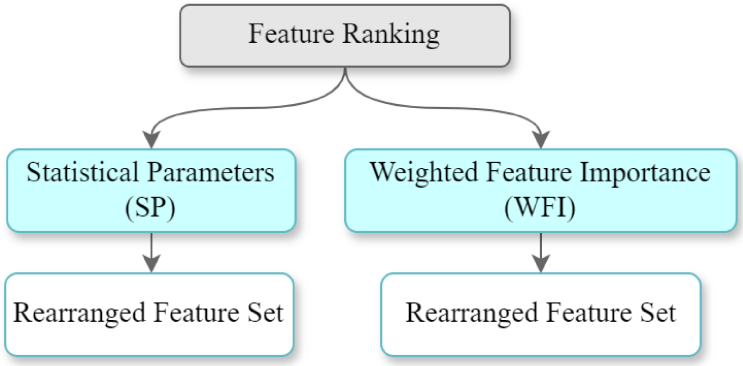


Figure 4.7: Feature Ranking Methods

The pre-processed dataset is sent over to the next phase i.e. “Feature Ranking”.

In this phase, the datasets features are ranked according to their possible significance in the dataset. Through this, we aim to reduce the feature space by eliminating the lesser important features. Fig 4.7 provides a visualization of the steps taken in the process of Feature Ranking. The rank of the features is determined using two mechanisms:

- Statistical Parameters (SP)
- Weighted Feature Importance (WFI)

For each feature in the feature list, we apply SP and WFI to yield a set of features ranked using the values of the feature in consideration. The features now having acquired their respective rank specific to the chosen mechanism are now combined to provide "$Combined\_Feature\_Rank$". In the following sections, we explain the working of SP and WFI in detail.

4.6.1 Statistical Parameters (SP)

This is the novel method proposed by us for ranking features by utilizing numerical characteristics of the dataset. For our experimentation, we have considered four SP namely:

- Standard Deviation
- Absolute Difference
- Skewness
- Kurtosis

Through consideration of multiple such characteristics, it devises a ranking amongst the features in the dataset. The SP used to compute the overall rank as well as the order of feature arrangement associated with it are shown in the Table 4.2.

To determine the order in 4.2 we conducted multiple experiments to ensure optimal results. Fig 4.8 below provides a block diagram of the experiments carried out. To understand the effect of SP on the overall model performance we carried out a three-part experimentation considering different scenarios:

Stastical Metrics	Order
1. Standard Deviation	Ascending
2. Absolute Difference	Ascending
3. Skewness	Descending
4. Kurtosis	Ascending

Table 4.2: Statistical Parameters and order of arrangement

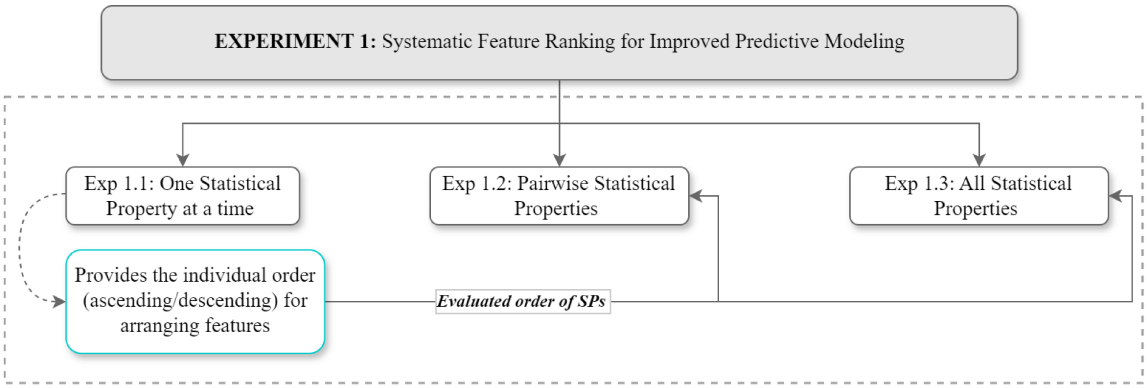


Figure 4.8: Experiments to infer the best combination of SP

- *Taking one SP at a time:* This experiment helped us understand the importance of individual SP as well as the best possible order of arrangement of the features i.e. ascending/descending for a paragon performance. The results acquired here were used as a pedestal for the other two experiments.

- *Taking pairwise SP:* Through this experiment, we tried to understand if any of the two SP aid towards better performance. Table 4.3 shows all the pairs of SP we used in our research.

Pairwise SP					
Standard Deviation, Absolute Difference	Standard Deviation, Skewness	Standard Deviation, Kurtosis	Absolute Difference, Skewness	Absolute Difference, Kurtosis	Skewness, Kurtosis

Table 4.3: Pairs of statistical parameters used in the Experiment 1.2

- *Taking all SP:* Here, we considered all the mentioned SP to compute the model performance.

We compared the accuracy acquired across these experiments to deduce the best possible combination that prompts our research goal. Based on the results, we observed that the ranking devised using all the SP performs the best.

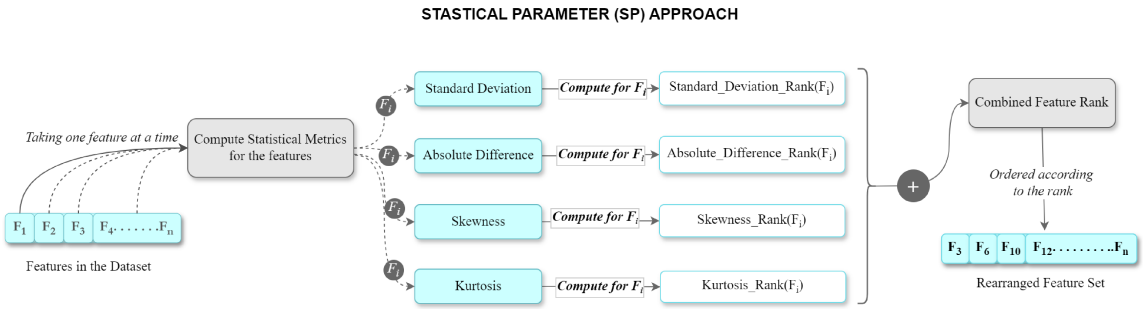


Figure 4.9: Process of computing Feature Rank using Statistical Parameters

The block diagram in Fig 4.9 provides a detailed view of the entire procedure. As illustrated, all features except the dependent variable in the dataset are considered for computation. For the entire feature list, we start by computing all the SP. Using the values acquired, the rank for each parameter is computed along with the ordering of the features. This gives us a feature rank across four SP i.e. $Standard\_Deviation\_Rank$, $Absolute\_Difference\_Rank$, $Skewness\_Rank$, and $Kurtosis\_Rank$. These individual ranks are then aggregated to obtain a combined feature rank. The ordering based on this combined rank yields a new list of features arranged by their importance in the dataset. Our analysis indicates that the ordering of features according to their combined feature rank significantly enhances the overall performance of the classifiers. Consequently, the features are ordered in ascending order of their rank. The algorithm followed is presented below:

Algorithm 1 Computation of Feature Ranks based on Statistical Parameters

Input: List containing features in a dataset D.

1. **Initialize**: Define a Feature set list $F \leftarrow [F_1, F_2, \ldots, F_n]$

2. **Compute the value of Statistical Parameters as well as resultant Feature rank**: For each feature F_i where $i = [1 \text{ to } n]$, compute the following statistical parameters:

   (a) Standard Deviation (σ)
   $StdDev\_rank(F_i) \leftarrow rank(\sigma(F_i))$
   Order $StdDev\_rank(F_i)$ in ascending order.

   (b) Absolute Difference (Δ)
   $AbsDiff\_rank(F_i) \leftarrow rank(abs(mean(F_i) - median(F_i))$
   Order $AbsDiff\_rank(F_i)$ in ascending order

   (c) Skewness (γ)
   $Skew\_rank(F_i) \leftarrow rank(skewness(F_i))$
   Order $Skew\_rank(F_i)$ in descending order

   (d) Kurtosis (κ)
   $Kurt\_rank(F_i) \leftarrow rank(kurtosis(F_i))$
   Order $Kurt\_rank(F_i)$ in ascending order

3. **Compute Overall Rank**:
   $combined\_feature\_rank = \sum_{i=1}^{n}(rank(statisticalmetrics(F_i)))$
   or
   $combined\_feature\_rank = StdDev\_rank(F_i) + AbsDiff\_rank(F_i) + Skew\_rank(F_i) + Kurt\_rank(F_i)$

4. **Output**: Reordered feature set, $F_{\text{new}} \leftarrow [F_3, F_6, \ldots, F_n]$

4.6.2 Weighted Feature Importance (WFI)

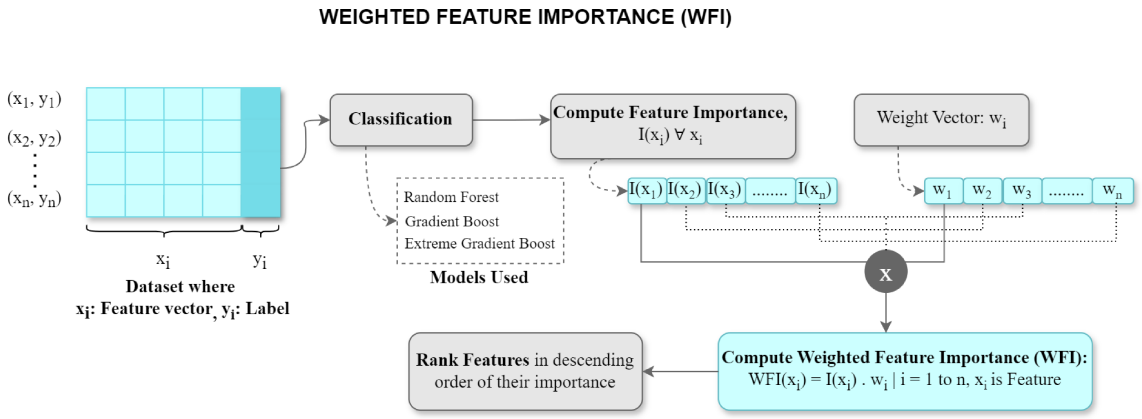


Figure 4.10: Process of computing Feature Ranking using WFI mechanism

WFI is the second-ranking mechanism we have utilized for feature ranking. This method computes the WFI scores for the features in the dataset. Fig 4.10 illustrates how WFI approach computes feature rank. Given a dataset, we represent it as (x_i, y_i) where x_i represents the Feature vector and y_i is the target label. This dataset is fed into a classifier to compute the Importance of features i.e. $I(x_i)$. Along with this, we also have a weight vector containing weights for each feature. Having acquired these two vectors, we have computed the WFI scores by multiplying Feature Importance with its weight as shown in Fig 4.10. The features are then arranged in descending order of their WFI scores to obtain a newly arranged feature set. The algorithm followed is presented below:

Algorithm 2 Compute Weighted Feature Importance (WFI) using RF classifier

Training data $D = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where x_i are feature vectors and y_i are labels.

Result: Weighted Feature Importance (WFI) scores for each feature

Step 1: Train Random Forest classifier

Train a Random Forest classifier on the training dataset D. Random Forests are ensemble learning methods that build multiple decision trees and combine their predictions to improve accuracy and robustness.

Step 2: Compute feature importance scores

After training, compute the feature importance scores $I(x_i)$ for each feature x_i. Feature importance in Random Forests is typically measured by how much each feature contributes to reducing impurity (e.g., Gini impurity) across all decision trees in the forest.

Weight vector $W = \{w_1, w_2, ..., w_m\}$ where w_i are weights for each feature **Step 3: Calculate Weighted Feature Importance (WFI) scores**

Initialize an array WFI_scores with zeros. For each feature x_i:

1. Multiply the feature importance score $I(x_i)$ by its corresponding weight w_i from the weight vector W.

2. Store the result in the WFI_scores array at index i.

Step 4: Rank features based on WFI scores

Sort the WFI_scores array in descending order. The higher the WFI score for a feature, the more important it is considered to be in predicting the target variable.

WFI_scores: Array of weighted feature importance scores, sorted in descending order

4.7 Feature Selection

This phase deals with the selection of features that aid in increasing the accuracy of the classification model with a minimal number of features. It takes the new rearranged feature set derived in the last phase i.e. feature ranking as an input. To eliminate features, we have used two algorithms namely:

- **Selective Promising Feature Selection (SPFS)**
- Forward Feature Selection (FFS)

The performance of both algorithms is assessed using "accuracy" as a metric. The output of SPFS and FFS consists of a new, enhanced feature set designed to improve the overall performance of the classification algorithm. The SPFS algorithm is our innovative contribution to feature elimination methodologies. Detailed explanations of these algorithms are provided in the following sections.

4.7.1 Selective Promising Feature Selection (SPFS)

As the name suggests, this algorithm is designed to retain the most promising features in the feature set. To accumulate the "promising" features that fit the desired criteria, we have used accuracy as a filtering parameter. This process eliminates the unnecessary features that don't increase the overall performance of the classifiers thereby reducing the feature space which is integral for an IDS for CI. Fig 4.11 is a flowchart of the series of steps taken in SPFS. The steps below outline the operation of SPFS.

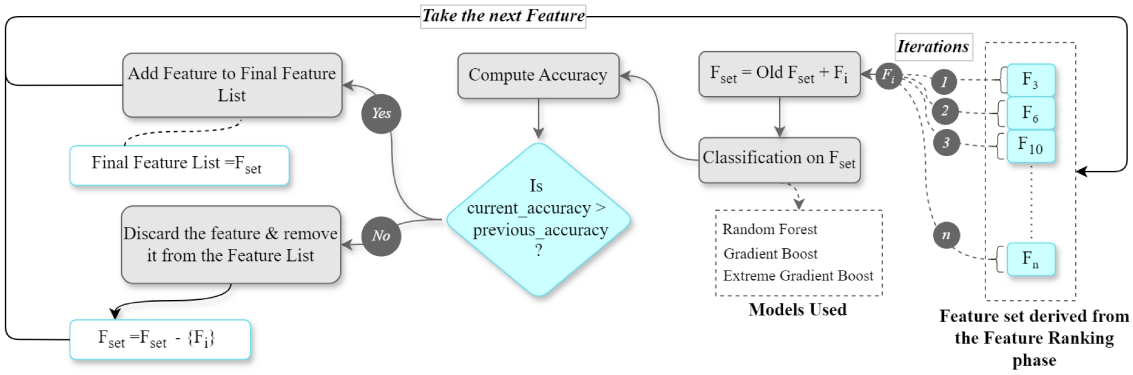


Figure 4.11: Flowchart illustrating the operation of SPFS

- Input: The rearranged feature set containing features arranged according to their importance from the previous phase.

- Iterative Feature Evaluation Module: Features are evaluated one at a time in an iterative manner. In each iteration, one feature F_i is taken from the list of features to be added to the F_{set}. For future iterations, this subset will contain all the features that have met the performance criteria along with the feature F_i in consideration. This subset is then passed to the next module for further computation.

- Model Training and Testing Module: The F_{set} acts as an input for the classification model to compute the accuracy, f1 score, and training time of the model. The model uses 80% of the data for training and 20% for testing.

- Feature Selection Module: The accuracy of the model with the currently considered feature is compared against the previously attained maximum accuracy. If the inclusion of the feature improves accuracy, it is retained as part of the final set or else, it is discarded.

- Output: Set of features that improve the performance of the model.

An algorithm 4 that explains the series of steps performed in SPFS is provided at the end of this section.

4.7.2 Forward Feature Selection (FFS)

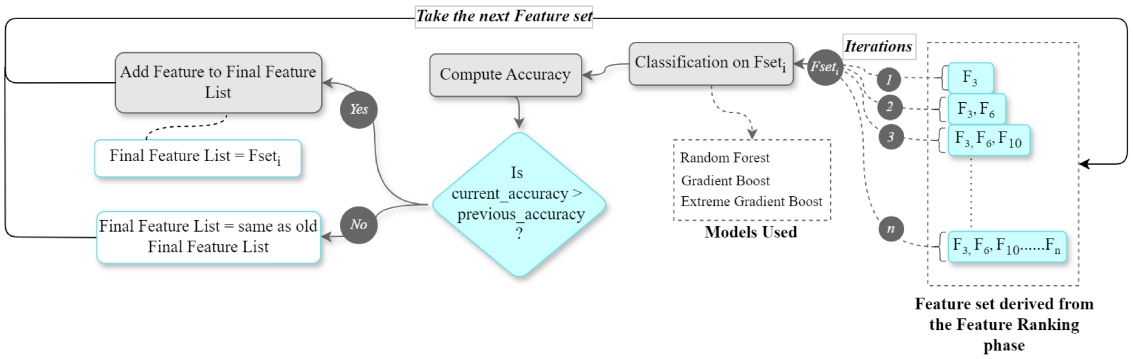


Figure 4.12: Flowchart illustrating the operation of FFS

The FFS algorithm evaluates subsets of the feature list iteratively, in contrast to the individual feature evaluation approach of SPFS. Fig 4.12 is a flowchart of the steps performed in FFS. FFS is designed to achieve the maximum attainable accuracy for the subset under consideration. Unlike SPFS, it does not discard any features; instead, it continually aggregates previously selected features into the newly considered subset for classifier evaluation. The FFS algorithm also employs "accuracy" as a performance metric to determine the optimal feature set. The steps below describe the operation of FFS.

- Input: The rearranged feature set containing features arranged according to their importance from the previous phase.
- Iterative Feature Evaluation Module: Different subsets of features derived from the feature list are evaluated iteratively. In each iteration, the current subset being taken is added to the $Fset_i$. $Fset_i$ is now ready to be passed on to the next step.
- Model Training and Testing Module: The $Fset_i$ acts as an input for the classifier to compute the accuracy, F1 score, and training time of the model. The model uses 80% of the data for training and 20% for testing.
- Feature Selection Module: The accuracy attained from the classifier using the current feature set is compared against the previously attained maximum accuracy. It records the maximum accuracy attained with the given feature subset.
- Output: Set of features that improve the performance of the model.

An algorithm 3 that explains the series of steps performed in FFS is provided at the end of this section.

4.7.3 ML Algorithms used in Feature Selection

The feature selection works in conjunction with different classification algorithms to determine the credibility of a certain feature. Figure 4.13 shows in detail the classification models used for each of the ranking mechanisms. The entire Feature Selection process is carried out as shown in Fig 4.13. Experiment 2: Comparative

Analysis of Feature Selection Mechanisms Across Multiple Classifiers and Ranking Mechanisms uses both the Feature Selection algorithms (SPFS, FFS) and RF, GB, and XGB on the feature sets acquired using SP and WFI. We have applied the feature selection algorithms on the feature sets acquired using the ranking mechanisms (SP, WFI) on the Command, Function, and Response datasets. To acquire the best feature set we have used the classifiers: RF, GB, and XGB. Table 4.4 provides details regarding the algorithms, classifiers, and their hyper-parameters used in this experiment.

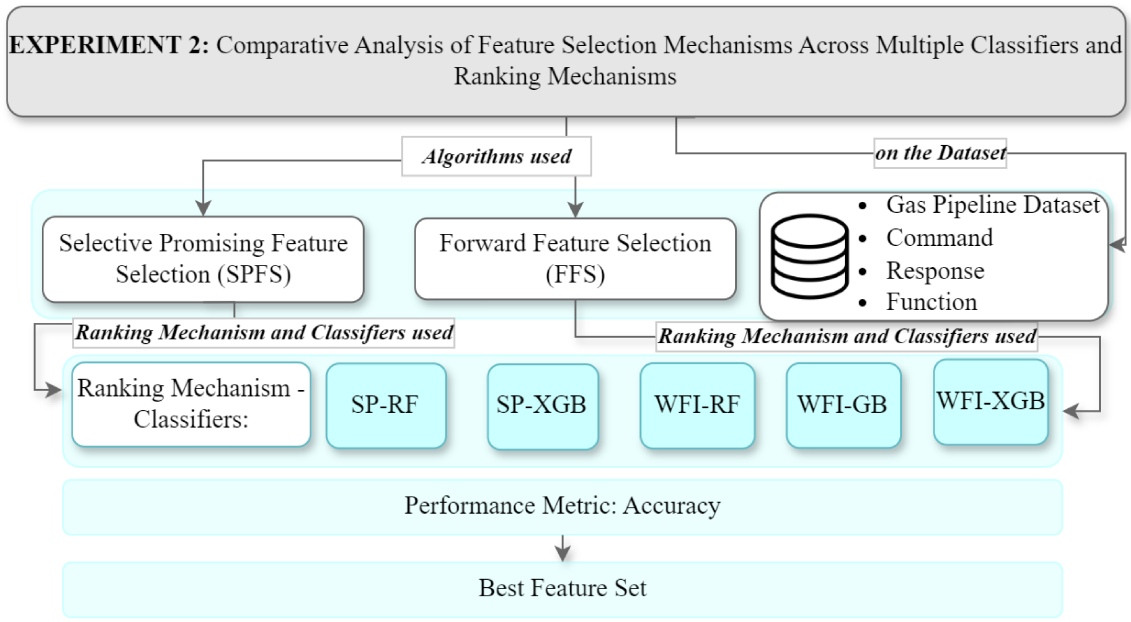


Figure 4.13: Flowchart of Experiment 2

Random Forest (RF)

RF is an ensemble method based on the logic of the DT. It builds multiple DT using different samples of the data from the dataset using averaging to enhance predictive accuracy and mitigate over-fitting [33]. We performed multiple experiments to select the best hyperparameters. The final hyperparameters selected were n_estimators: 50 and random_state=42. Based on the earlier experimentation, we decided to use a feature set generated by both the ranking mechanisms (SP, WFI) as inputs for RF classification. Following are the two experiments performed using the RF classifier:

- SP-RF: Classification using RF based on ranking derived from SP.

- WFI-RF: Classification using RF based on ranking derived from WFI.

Gradient Boosting (GB)

GB is an ensemble technique used for classification as well as regression problems in supervised learning. It improves the model performance by iteratively reducing the loss by using Gradient Descent Optimization [35]. We performed multiple experiments to select the best hyperparameters. The final hyperparameters selected were n_estimators: 50 and random_state=42. Based on our earlier experiments, we performed classification using GB only on the feature set acquired using WFI i.e. WFI-GB.

XGBoost (XGB)

XGB is an optimized version of Gradient Boost algorithm. Since it provides parallel tree boosting, it is capable of solving ML problems more accurately and faster [63]. We performed multiple experiments to select the best hyperparameters. The final hyperparameters selected were n_estimators: 200 and random_state=42. Based on the earlier experimentation, we concluded to use feature set generated by both the ranking mechanisms (SP, WFI) as inputs for XGB classification. Following are the two experiments performed using the XGB classifier:

- SP-XGB: Classification using XGBoost based on ranking derived from SP.
- WFI-XGB: Classification using XGBoost based on ranking derived from WFI.

Dataset	RM	FS Algo	ML Classifier	Hyperparameters
Command, Function, and Response	SP	SPFS, FFS	RF	n_estimators=50, random_state=42
		SPFS, FFS	XGB	n_estimators=200, random_state=42
	WFI	SPFS, FFS	RF	n_estimators=50, random_state=42
		SPFS, FFS	GB	n_estimators=50, random_state=42
		SPFS, FFS	XGB	n_estimators=200, random_state=42

Table 4.4: Datasets, Ranking Mechanism, Feature Selection Algorithm, Classifiers, and Hyperparameters used in Feature Selection Phase.

4.8 Classification using Cross-Validation

Algorithm 3 FFS using SP Ranking and classification using Cross-Validation

1: Read the data from the CSV file:
$df \leftarrow pd.read\_csv('NewGasFilteredCommand\_Preprocessed.csv')$
2: Replace and remove infinite values with NaN:
$df.replace([np.inf, -np.inf], np.nan, inplace = True)$, $df \leftarrow df.dropna()$
3: Convert Labels 'Normal' to 0 and 'Attack' to 1:
$df['Label'] \leftarrow df['Label'].replace('0' : 0,' 1' : 1)$
4: Extract features and Labels:
$X \leftarrow df.drop(columns = ['Label'])$, $y \leftarrow df['Label']$
5: Compute and rank the features based on standard deviation (σ):
$std\_dev \leftarrow X.std()$, $std\_dev\_rank \leftarrow std\_dev.rank(ascending = True)$
6: Compute the absolute difference (D) of mean and median of the features:
$abs\_diff\_mean\_median \leftarrow np.abs(X.mean() - X.median())$,
$abs\_diff\_rank \leftarrow abs\_diff\_mean\_median.rank(ascending = True)$
7: Define a function to compute the rank of features based on skewness:
$skewness \leftarrow X.apply(skew)$, $skew\_rank \leftarrow skew.rank(ascending = False)$
8: Define a function to compute the rank of features based on kurtosis:
$kurt \leftarrow X.apply(kurtosis)$, $kurt\_rank \leftarrow kurt.rank(ascending = True)$
9: Compute combined feature rank as
$comb\_feature\_rank \leftarrow std\_dev\_rank + abs\_diff\_rank + skew\_rank + kurt\_rank$
10: Sequential feature selection based on the current accuracy is better
11: Current subset ← combined feature rank sorted in ascending order
12: **for** each feature in current subset **do**
13: Previous subset ← current subset, Select one feature to the current subset
14: Split the data into train and test sets and evaluate a classifier
15: **if** current accuracy > previous accuracy **then**
16: Update previous accuracy, F1 score and Final set ← current subset
17: **end if**
18: **end for**
19: Output the final selected features: Final set
20: Select features based on the Final set
21: Initialize KFold cross-validation with 10 folds
22: Compute accuracy, precision, recall, F1 score, confusion matrix, execution time.

With the new feature set acquired, we proceeded to the final experiment of our study. In this phase, we conducted a 10-fold cross-validation to obtain a more reliable estimate of the models' performance. The cross-validation was performed on the Gas pipeline dataset as well as the Command, Function, and Response datasets. The classifiers selected were based on the feature selection mechanism (ranking mechanism and classifier used for feature selection). Table 4.5 provides details regarding the classifiers used, the ranking mechanisms that supplied the feature set, and the hyperparameters of the respective classifiers. We considered accuracy, f1 score, execution time, total features, precision, recall, FPR, FNR, TPR, and TNR as performance metrics in this phase. Figure 4.14 illustrates the classification models, input feature sets, and performance metrics.

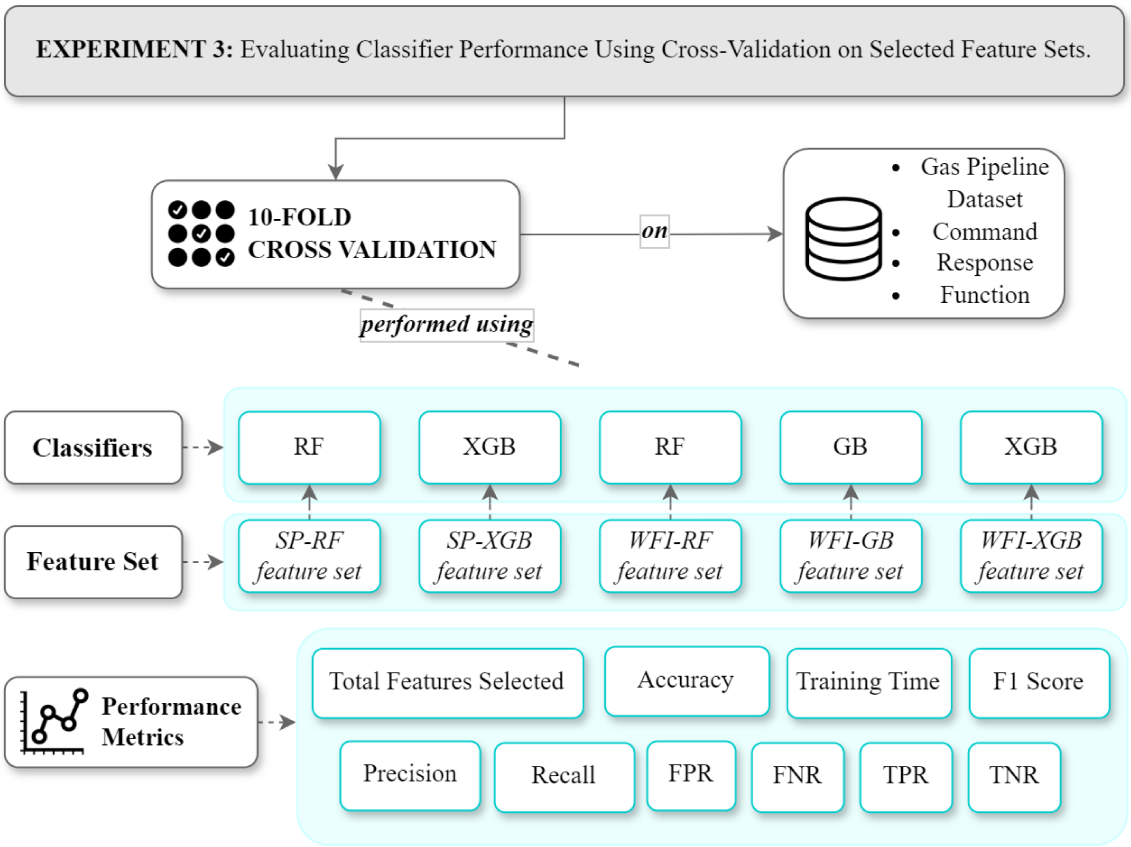


Figure 4.14: Classification Models and Performance Metrics for 10 fold Cross Validation

Classifier	**Ranking Mechanism**	**Hyper Parameters**
Random Forest (RF)	WFI	n_estimators= 200, random_state=42
	SP	n_estimators= 200, random_state=42
Gradient Boost (GB)	WFI	n_estimators= 200, random_state=42
Extreme Gradient Boost (XGB)	WFI	n_estimators= 200, random_state=42
	SP	n_estimators= 200, random_state=42

Table 4.5: Classifier, Ranking Mechanism, and Hyper Parameters in 10-fold Cross Validation

Algorithm 4 SP Ranking using SPFS for Binary classification using Cross-Validation

1: Read the data from the CSV file.
2: Replace and remove infinite values with NaN:
$df.replace([np.inf, -np.inf], np.nan, inplace = True)$, $df \leftarrow df.dropna()$
3: Convert Labels 'Normal' to 0 and 'Attack' to 1:
$df['Label'] \leftarrow df['Label'].replace('0' : 0, '1' : 1)$
4: Extract features and Labels: $X \leftarrow df.drop(columns = ['Label'])$, $y \leftarrow df['Label']$
5: Compute and rank the features based on standard deviation (σ):
$std\_dev \leftarrow X.std()$, $std\_dev\_rank \leftarrow std\_dev.rank(ascending = True)$
6: Compute the absolute difference (D) of mean and median of the features:
$abs\_diff\_mean\_median \leftarrow np.abs(X.mean() - X.median())$,
$abs\_diff\_rank \leftarrow abs\_diff\_mean\_median.rank(ascending = True)$
7: Define a function to compute the rank of features based on skewness:
$skewness \leftarrow X.apply(skew)$, $skew\_rank \leftarrow skew.rank(ascending = False)$
8: Define a function to compute the rank of features based on kurtosis:
$kurt \leftarrow X.apply(kurtosis)$, $kurt\_rank \leftarrow kurt.rank(ascending = True)$
9: Compute combined feature rank and order it in ascending order as
$comb\_feature\_rank \leftarrow std\_dev\_rank + abs\_diff\_rank + skew\_rank + kurt\_rank$
10: Previous accuracy $\leftarrow$ 0, Final set $\leftarrow$ [], Current subset $\leftarrow$ $comb\_feature\_rank$
11: **for** each feature in current subset1 **do**
12: Add one feature to the Final set.
13: Previous subset $\leftarrow$ Finalset, current subset $\leftarrow$ Previous subset
14: Split the data into train and test sets to Train and Evaluate the classifier
15: Compute execution time, accuracy and f1-score
16: **if** current accuracy > previous accuracy **then**
17: Update previous accuracy, F1 score
18: **else**
19: Remove current feature from Final set.
20: **end if**
21: **end for**
22: Perform K-Fold cross-validation with 10 folds on the Final Set
23: Compute and output execution time accuracy, precision, recall, F1 score, and confusion matrix.

Chapter 5

Results and Discussions

5.1 Overview

In this section, we will systematically discuss and analyze the results obtained from our experimental investigations. As mentioned in the preceding sections, our research encompassed three distinct experiments. Consequently, the results are organized according to the specific experiments conducted. Sections 5.2, 5.4, 5.5, 5.6 present the results and analyses of Experiments 1 through 3, as well as results acquired from Gas Pipeline dataset and the Command, Function, and Response clusters. Section 5.7 synthesizes the outcomes derived from the aggregated data of all experiments.

5.2 Evaluating Optimal Machine Learning Classifier for Gas Pipeline Dataset Analysis

Following a comprehensive understanding of the dataset and initial pre-processing steps, our objective was to identify the most effective machine learning classifiers for optimal performance. Here we used the complete Gas Pipeline dataset to determine the top three performing classifiers for subsequent experiments using all three clusters of the gas pipeline dataset. We tried five machine learning classifiers namely RF, DT, GB, XGB, and NB using the SP-SPFS approach. Table 5.1 contains the accuracy and total execution time for all ML classifiers. As shown in Fig 5.1 our findings indicate that RF, DT, and XGB are the most effective classifiers, achieving an accuracy of 98.91% with 16 features, 99.14% with 14 features, and 97.95% using 17 features, respectively on a Non-Normalized dataset. From the results acquired, it can be deduced that RF, DT as well as XGB perform better when the dataset is not normalized. The graph in Fig 5.2 shows the performance of all classifiers for the non-normalized and normalized gas pipeline dataset. As inferred from the results above, the tree-based classifiers perform better than the others. Therefore, we

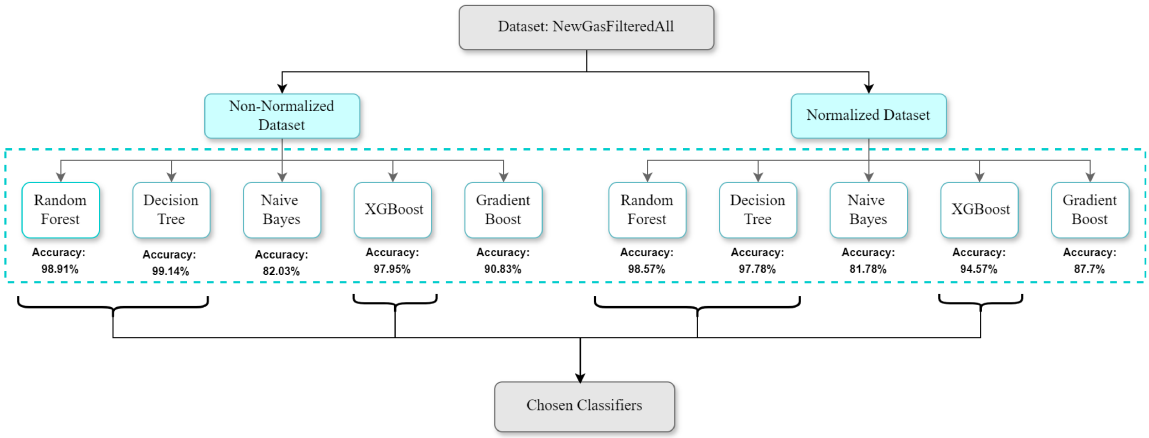


Figure 5.1: Performance accuracy of different Machine Learning Classifiers for NewGasFilteredAll dataset with and without min-max normalization.

Table 5.1: Performance metrics obtained with 10-fold cross-validation of all ML classifiers for Gas Pipeline dataset using SP-SPFS approach.

Classifier	Gas Pipeline Dataset			Normalized Gas Pipeline Dataset		
	Accuracy	Total_Time	F1-Score	Accuracy	Total_Time	F1-Score
DT	**99.22**	**24.43**	**0.98**	**96.71**	**32.888**	**0.949**
RF	**99.1**	**1067.26**	**0.974**	**98.69**	**1277.27**	**0.966**
GB	92.3	646.44	0.734	88.2	712.25	0.606
XGB	97.96	51.69	0.951	94.41	51.9	0.864
NB	81.64	0.989	0.364	81.52	1.075	0.386

have used the tree-based classifiers for the analysis of the Command, Function, and Response clusters.

5.3 Comparative Analysis of Feature Selection and Non-Feature Selection Techniques on the Gas Pipeline Dataset

In this sub-experiment, we conducted a comparative analysis to understand the impact of feature ranking and feature selection on anomaly detection. We have compared the performance of the SP-SPFS methodology applied to the gas pipeline dataset against scenarios where feature ranking and selection were not employed. Fig 5.3 shows an elaborate view of the performance of ML classifiers on a non-normalized dataset using no feature selection or feature ranking mechanisms. As inferred from the results, the RF classifier attains a maximum accuracy of 99.1%, f1-score of 0.986, precision of 0.991, and recall of 0.981 outperforming other methods. Fig 5.5 (a) shows

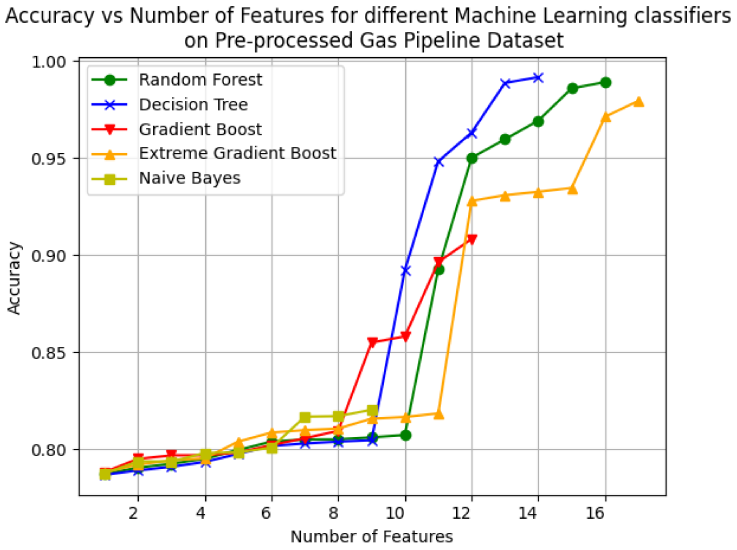


Fig (a)

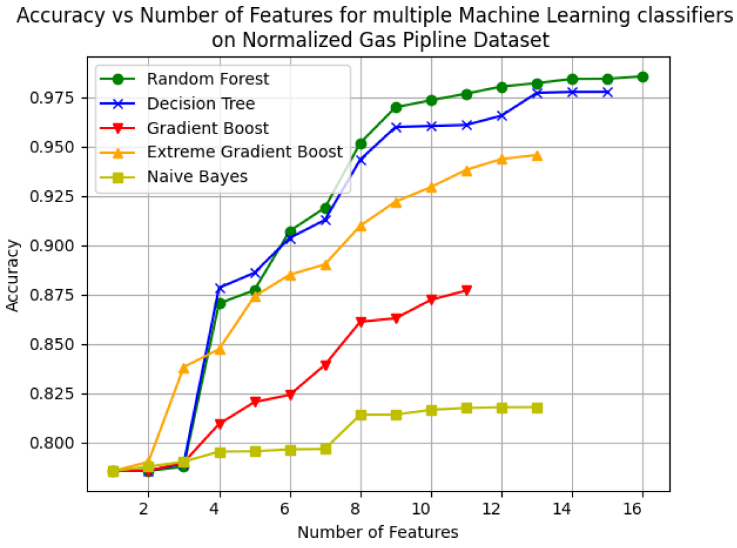


Fig (b)

Figure 5.2: Fig (a): Accuracy vs Number of features of ML classifiers on non-normalized Gas Pipeline dataset. Fig (b): Accuracy vs Number of features of ML classifiers on normalized Gas Pipeline dataset.

the total execution time of the classifiers in seconds. As observed, the total execution time for RF is 4110.27s. According to the majority of performance metrics, RF is the best approach for a non-normalized gas pipeline dataset. Fig 5.4 shows an elaborate view of the performance of ML classifiers on a normalized dataset using no feature selection or feature ranking mechanisms. As inferred from the results, the RF classifier attains a maximum accuracy of 96.9%, f1-score of 0.953, precision of 0.962, and recall of 0.945 outperforming other methods. Fig 5.5 (b) shows the

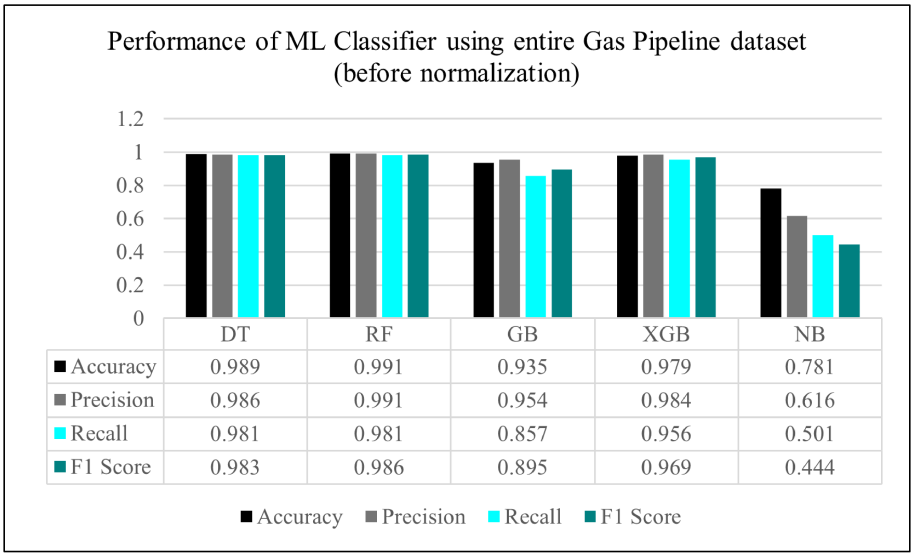


Figure 5.3: Performance of ML classifiers on gas pipeline dataset with pre-processing and no feature selection

total execution time of the classifiers in seconds. As observed, the total execution time for RF is 3911.049s. According to the majority of performance metrics, RF is the best approach for a normalized gas pipeline dataset as well. Based on the

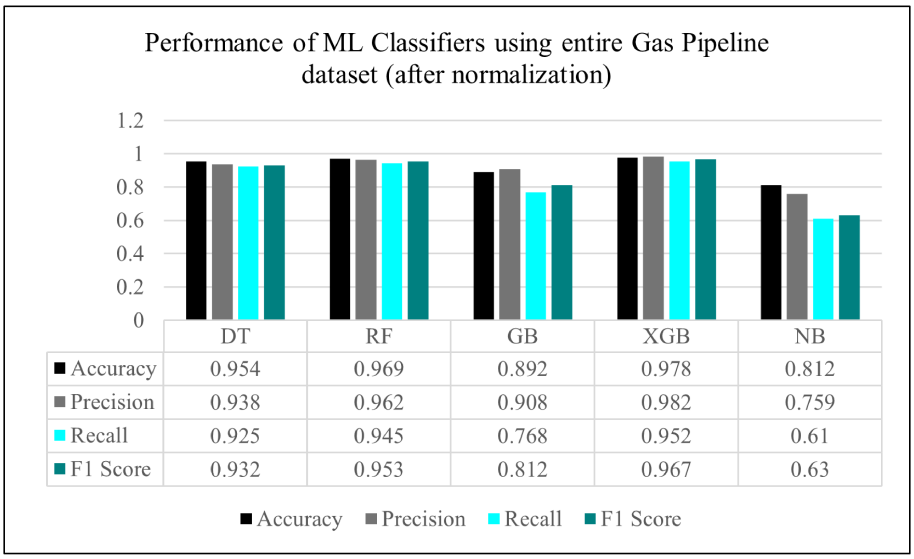


Figure 5.4: Performance of ML classifiers on gas pipeline dataset with no feature selection and no pre-processing

data presented above and data acquired from 5.2 the application of the SP-SPFS-DT approach it can be inferred that SP-SPFS-DT outperforms the RF classifier used here. The SP-SPFS-DT attains an accuracy of 99.22% in 24.43s after using 10-fold

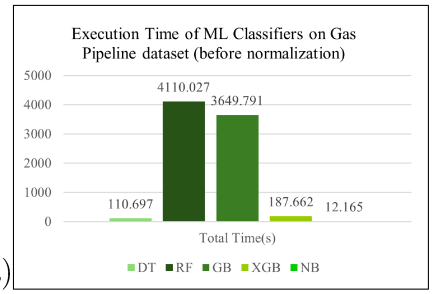
Fig (a) 

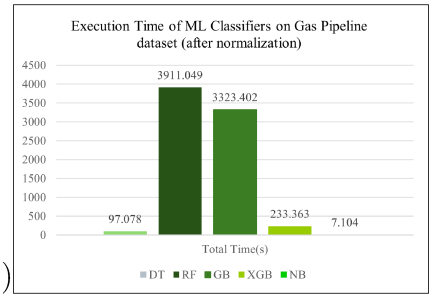
Fig (b) 

Figure 5.5: Fig (a): Execution time of ML classifiers before normalization Fig (b): Execution time of ML classifiers after normalization

cross-validation, which is far less in comparison to the execution time of 4110.27s by the RF classifier. Hence, for the gas pipeline dataset, the feature ranking and feature selection have managed to increase the performance metrics while decreasing the overall execution time.

5.4 Results for Systematic Feature Ranking for Improved Predictive Modeling

To improve the performance of the classifiers further, we employed the feature ranking techniques (SP, WFI). This section elucidates the results that prompted the underlying feature arrangement hierarchy. As mentioned in the earlier chapters, we carried out three sub-experiments to understand the impact on classifiers' performance for different statistical properties given a dataset. The dataset used in these experiments is the gas pipeline dataset. The subsequent sections present the experimental results and outline the inferences utilized to determine the optimal feature hierarchy.

5.4.1 Analysing impact of individual statistical parameters

To ensure the optimal performance of the classifiers we analyzed the impact that each statistical property induces on the overall performance. Table 5.2 and Table 5.3 show the performance of Random Forest based on features ranked according to the mentioned statistical parameter. Our goal is to determine the tuple i.e. accuracy and number of features that simulates the optimal performance of the classifiers. Fig 5.6 is a comparative plot showing accuracy acquired by using RF for all statistical

parameters based on the ordering of features.

Feature Rank order: Ascending		
Statistical Parameter	**Accuracy (%)**	**Number of Features**
Standard Deviation	96.59	14
Absolute Difference	96.63	15
Skewness	96.56	16
Kurtosis	96.66	16

Table 5.2: Performance of Statistical properties taken one at a time arranged in Ascending order of value

Feature Rank order: Descending		
Statistical Parameter	**Accuracy (%)**	**Number of Features**
Standard Deviation	96.63	16
Absolute Difference	96.59	16
Skewness	96.63	13
Kurtosis	96.63	13

Table 5.3: Performance of Statistical properties taken one at a time arranged in Descending order of value

In Table 5.2 and 5.3 we can see that when using Standard Deviation, Absolute Difference, and Kurtosis the classifier attains an accuracy of 96.59% using 14 features, 96.63% using 15 features and 96.66% using 16 features respectively. Although for statistical parameters accuracy is better when features are ranked in descending order however, it comes at the cost of using more features. Hence, we concluded that Standard Deviation, Absolute Difference as well as Kurtosis contribute better results when features are arranged in ascending order of *std_dev_rank*, *abs_rank*, and *kurt_rank* respectively. Whereas, Skewness contributes to better results when features are arranged in descending order of *skew_rank*. Table 4.2 summarizes the order associated with each statistical parameter.

5.4.2 Analysing impact of pairs of statistical parameters

After deducing the best possible ordering associated with each statistical parameter, we implemented these findings in this as well as further experiments. In this experiment, we considered the statistical parameters in pairs as shown in Table 4.3.

Fig (a)

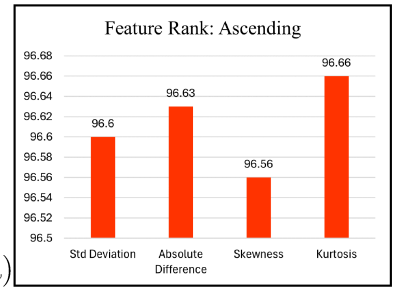


Fig (b)

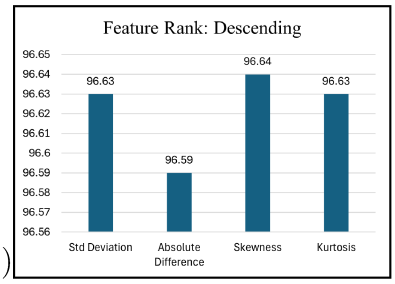


Figure 5.6: Fig (a): Performance of Random Forest when features are arranged in ascending order of the statistical parameters. Fig (b): Performance of Random Forest when features are arranged in descending order of the statistical parameters

We used Random Forest to understand how the statistical pairs influence the overall performance. Table 5.4 shows the accuracy and number of features taken to achieve that accuracy for different pairs of statistical parameters. Fig 5.7 shows a graphical representation of the results shown in the Table 5.4. We have observed that the

Pairs of Statistical Parameters	**Accuracy (%)**	**Feature #**
Std. Deviation, Absolute Difference	98.61	15
Std. Deviation, Skewness	98.57	16
Std. Deviation, Kurtosis	98.60	13
Absolute Difference, Skewness	98.58	15
Absolute Difference, Kurtosis	98.55	14
Skewness, Kurtosis	98.59	14

Table 5.4: Pairs of Statistical Parameters with Corresponding Accuracy and Feature Number

pair Standard Deviation and absolute Difference provide the maximum accuracy of 98.61% among all. However, the difference between the accuracy of other pairs and the highest accuracy attained isn't significant enough to declare the supremacy of one pair over the other. Another thing that we noticed is, that the pairs of statistical parameters aren't performing better than the scenario in which one statistical parameter is taken into account.

5.4.3 Analysing impact of taking all statistical parameters

In this experiment, we analyzed the impact of taking all statistical properties on the performance of Random Forest for the gas pipeline dataset. The results acquired from using all the SP in RF classification are presented in 5.1. Based on the results

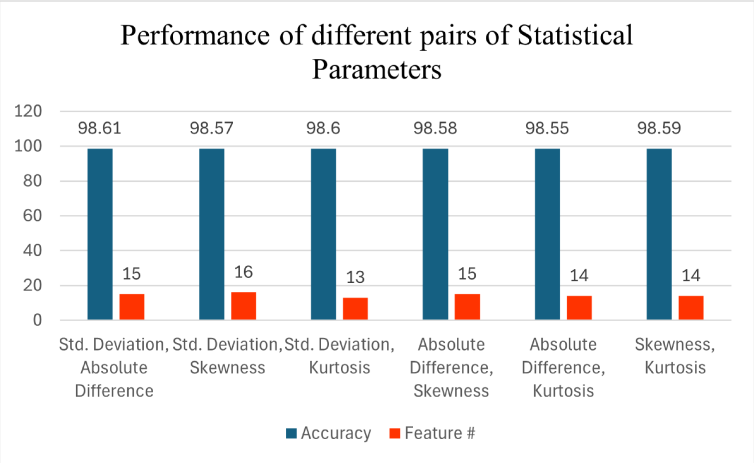


Figure 5.7: Graph representing the performance of different statistical pairs using Random Forest Classifier.

obtained, we concluded that the optimal performance was achieved when all statistical parameters were taken into account. Therefore, for subsequent experiments, we incorporated all statistical parameters.

5.5 Results for Comparative Analysis of Feature Selection Mechanisms Across Multiple Classifiers and Ranking Mechanisms

In the previous experiment, we established a hierarchical arrangement of features based on their importance using SP and WFI techniques. These methods were applied to three datasets: Command, Function, and Response, resulting in ranked feature sets for each. Both normalized and non-normalized datasets were considered for the final results. From these sets, to select the best possible sets of features that aid the performance of chosen classifiers (RF, XGB, and GB) we used:

- Feature Selection using SPFS
- Feature Selection using FFS

5.5.1 Feature Selection using SPFS

The SPFS algorithm is our novel contribution developed in this study. This algorithm synergizes with machine learning classifiers to deliver optimal performance by identifying and providing the most effective feature set. The subsequent sections present

a comprehensive analysis and the associated data pertaining to the application of SPFS on Command, Function, and Response datasets.

Command Dataset

As mentioned above, we used tree-based classifiers namely RF, XGB, and GB to analyze the command dataset. The evaluation was based on performance metrics such as accuracy and F1 score, which are critical in assessing predictive models. The ensuing Table 5.5 presents the detailed results obtained, offering insights into the effectiveness of each classifier in predicting the desired output for the dataset. This table presents the two sets of accuracy and f1-score values based on the normalization or non-normalization of the dataset. The values are associated with a specific ranking mechanism and classifier used.

As seen from the table, the RF classifier, when applied with the SP ranking

RM	**Classifier**	**Non-Normalized Dataset**		**Normalized Dataset**	
		ACC (%)	**F1-Score**	**ACC (%)**	**F1-Score**
SP	RF	**98.87**	**0.98**	99.21	0.98
	XGB	98.29	0.96	98.53	0.97
WFI	RF	98.81	0.97	**99.27**	**0.98**
	XGB	98.28	0.96	98.1	0.96
	GB	91.15	0.77	91.15	0.77

Table 5.5: Performance Metrics for Different Ranking Mechanisms and Classifiers for Command Dataset

mechanism, achieves superior results with an accuracy of 98.87% and an F1-Score of 0.98 on the non-normalized dataset. Similarly, employing RF with the WFI ranking mechanism yields even higher performance, achieving an accuracy of 99.27% and an F1-Score of 0.98, surpassing other configurations. Additionally, it is noteworthy that in context of normalized dataset the RF classifier exhibits consistent F1-Score performance across SP implementation, with a minor 0.06 difference in accuracy observed. Figure 5.8 illustrates the performance of classifiers in terms of Accuracy and F1-Score as each feature is incrementally added and evaluated using the SPFS algorithm on the non-normalized Command dataset. From Figure 5.8, it is evident that there is a significant increase in both Accuracy and F1-Score with the addition of the first four features in graphs (a) and (b). However, in graphs (c) and (d), it

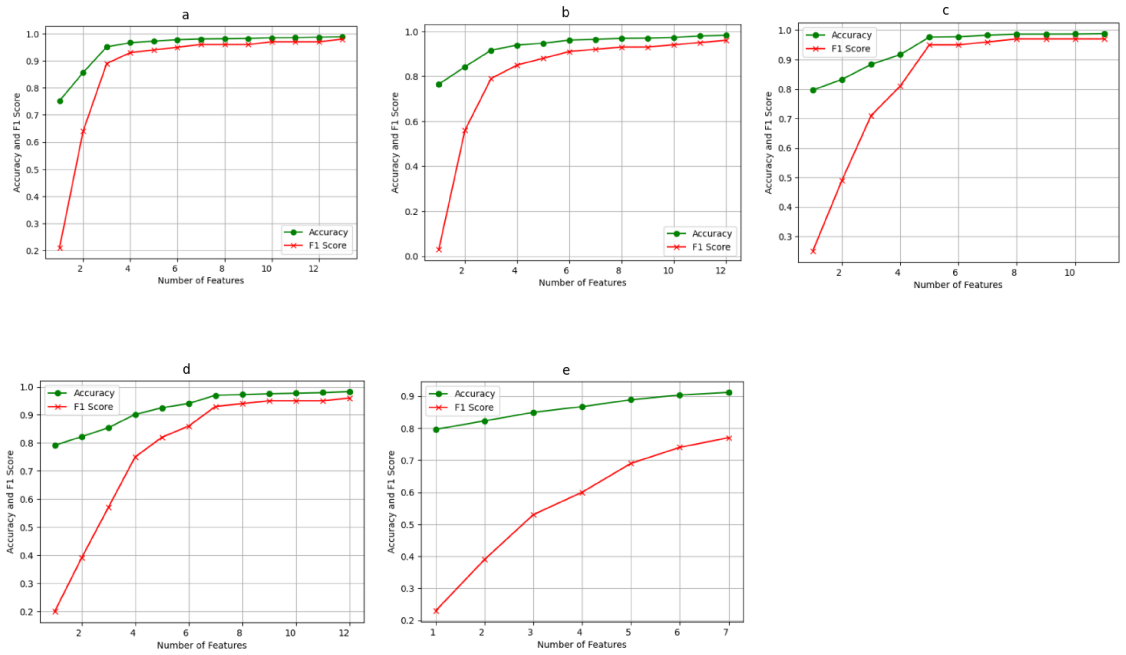


Figure 5.8: From left to right Graphs represent a.SP-SPFS-RF, b.SP-SPFS-XGB, c.WFI-SPFS-RF, d.WFI-SPFS-XGB, and e.WFI-SPFS-GB plot for Acc vs F1-Score on non-normalized command dataset.

takes six and seven features, respectively, before the Accuracy and F1-Score stabilize. This indicates that the major improvement in Accuracy and F1-Score occurs with the first five to six features. Beyond this point, adding more features does not significantly enhance the performance metrics. Specifically, the graph labeled (e) shows that the model achieves high Accuracy but a low F1-Score. Figure 5.9 presents the performance of classifiers based on Accuracy and F1-Score as each feature is added, evaluated using the SPFS algorithm on the normalized Command dataset. As depicted in Figure 5.9, for graphs (a) and (b), approximately five features are required before the Accuracy and F1-Score stabilize. In graphs (c) and (d), this stabilization occurs after the addition of about six features. Beyond this point, the performance metrics do not show a significant increase. Graph (e), representing the WFI-SPFS-GB model, demonstrates high Accuracy but a low F1-Score, suggesting that this approach may not be the most suitable for the normalized Command dataset.

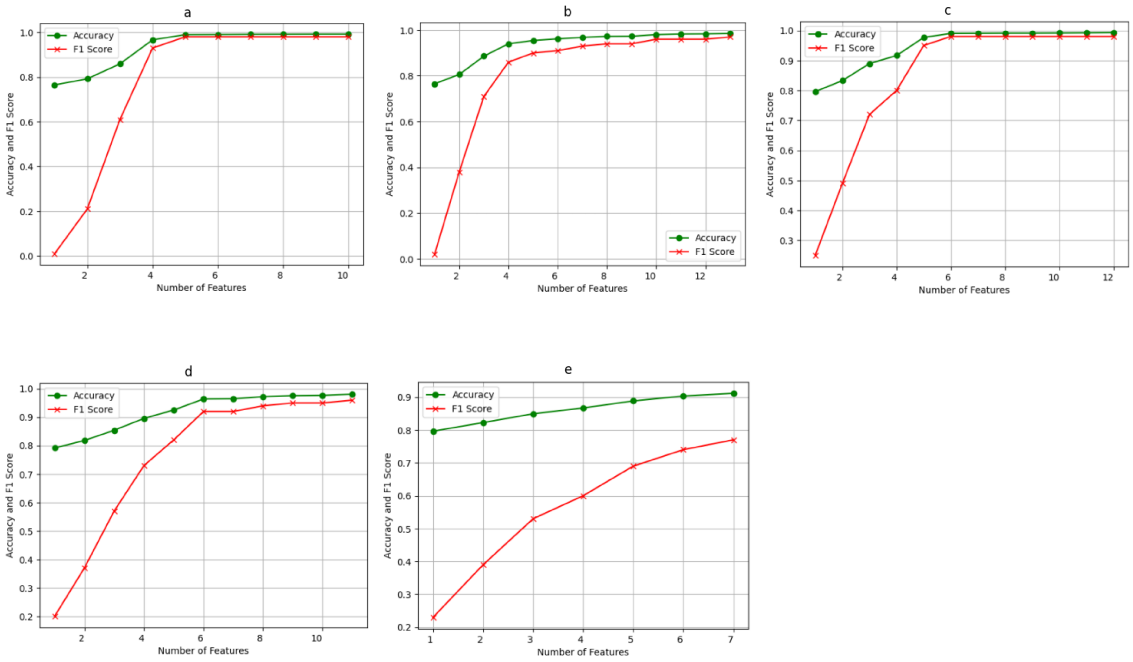


Figure 5.9: From left to right Graphs represent a.SP-SPFS-RF, b.SP-SPFS-XGB, c.WFI-SPFS-RF, d.WFI-SPFS-XGB, and e.WFI-SPFS-GB plot for Acc vs F1-Score on normalized command dataset.

Function Dataset

As mentioned above, we used tree-based classifiers namely RF, XGB, and GB to analyze the Function dataset. The evaluation was based on performance metrics such as accuracy and F1 score, which are critical in assessing predictive models. The ensuing Table 5.6 presents the detailed results obtained, offering insights into the effectiveness of each classifier in predicting the desired output for the dataset. This table presents the two sets of accuracy and f1-score values based on the normalization or non-normalization of the dataset. The values are associated with a specific ranking mechanism and classifier used.

As seen from the table, the RF classifier, when applied with the SP as well as WFI ranking mechanism, achieves superior results with an accuracy of 99.25% and an F1-Score of 0.98 on the non-normalized dataset. Similarly, employing RF with the WFI ranking mechanism yields even higher performance, achieving an accuracy of 99.53% and an F1-Score of 0.99, surpassing other configurations. For the dataset

RM	Classifier	Non-Normalized Dataset		Normalized Dataset	
		ACC (%)	F1-Score	ACC (%)	F1-Score
SP	RF	**99.25**	**0.98**	**99.43**	**0.98**
	XGB	98.87	0.97	98.15	0.94
WFI	RF	**99.25**	**0.98**	**99.53**	**0.99**
	XGB	98.65	0.99	98.59	0.96
	GB	95.29	0.84	95.29	0.84

Table 5.6: Performance Metrics for Different Ranking Mechanisms and Classifiers for Function Dataset

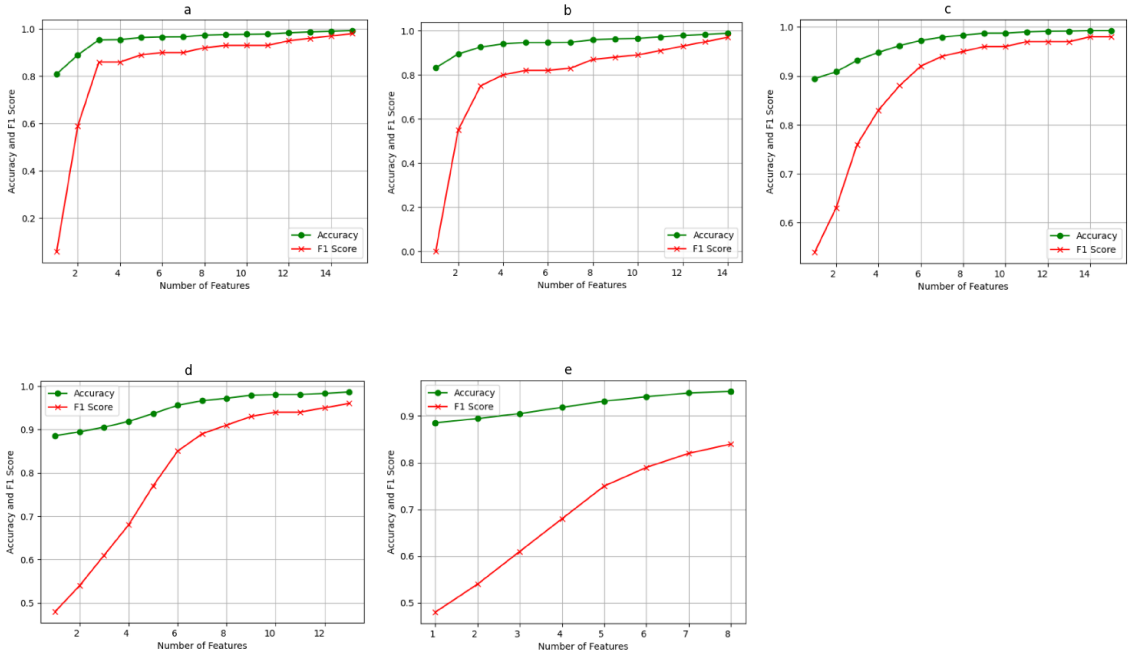


Figure 5.10: From left to right Graphs represent a.SP-SPFS-RF, b.SP-SPFS-XGB, c.WFI-SPFS-RF, d.WFI-SPFS-XGB, and e.WFI-SPFS-GB plot for Acc vs F1-Score on non-normalized Function dataset.

Function, RF seems to be the best classifier amongst the other classifiers providing high accuracy and F1-Score. Figure 5.10 illustrates the performance of classifiers based on Accuracy and F1-Score as each feature is incrementally added, evaluated using the SPFS algorithm on the non-normalized Function dataset. As inferred from Figure 5.10, graph (a) demonstrates that the most significant increase in performance parameters occurs with the addition of three features, beyond which there is no substantial improvement in performance. In graph (b), while Accuracy is high with

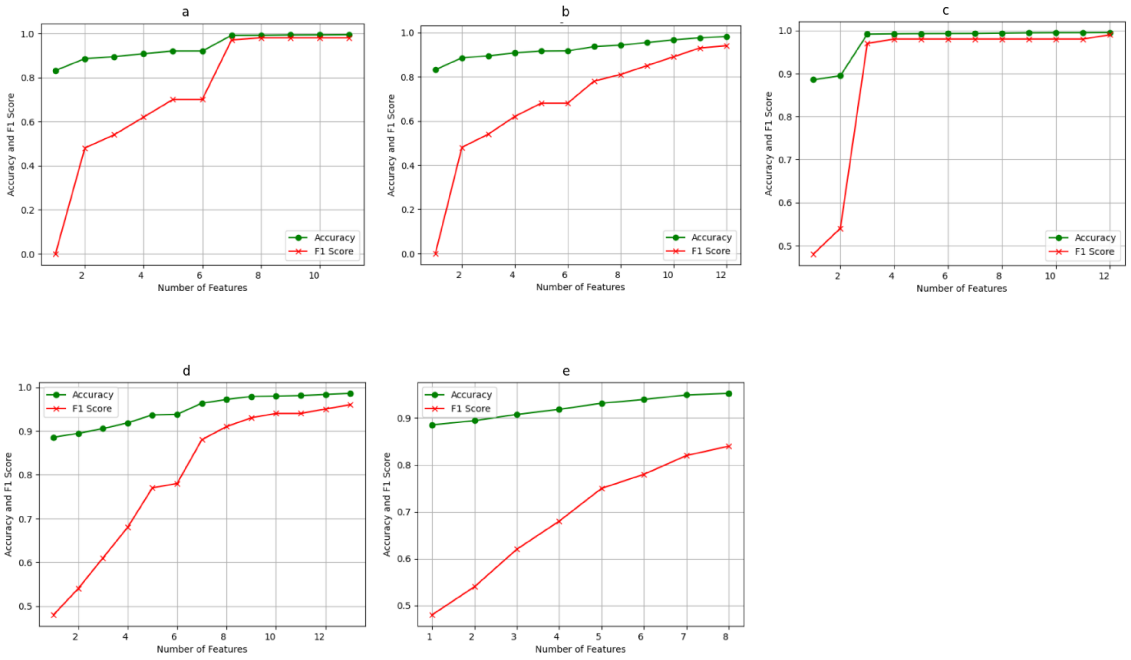


Figure 5.11: From left to right Graphs represent a.SP-SPFS-RF, b.SP-SPFS-XGB, c.WFI-SPFS-RF, d.WFI-SPFS-XGB, and e.WFI-SPFS-GB plot for Acc vs F1-Score on normalized Function dataset.

just the initial feature, it takes approximately six to seven features for the F1-Score to increase and stabilize. Graphs (c) and (d) show that it takes around six to eight features for the performance parameters to stabilize. Graph (e) reveals high Accuracy but a low F1-Score, indicating poor performance of the classifier. Figure 5.11 depicts the performance of classifiers based on Accuracy and F1-Score as each feature is added, and evaluated using the SPFS algorithm on the normalized Function dataset. As observed from Figure 5.11, graphs (a) shows an increase in accuracy and F1-Score with seven features whereas graph (b) shows no particular spike in the accuracy on the addition of any feature but the first and a steady increase in F1-Score. In graph (c), both performance metrics exhibit a maximum spike with just three features and stabilize thereafter. Graph (d) indicates that the first seven features contribute to the increase in both Accuracy and F1-Score. However, compared to other approaches, graphs (a), (b), and (c) show a better increase in F1-Score. Graph (e) displays poor performance in terms of F1-Score specifically.

Response Dataset

As mentioned above, we used tree-based classifiers namely RF, XGB, and GB to analyze the Response dataset. The evaluation was based on performance metrics such as accuracy and F1 score, which are critical in assessing predictive models. The ensuing Table 5.7 presents the detailed results obtained, offering insights into the effectiveness of each classifier in predicting the desired output for the dataset. This table presents the two sets of accuracy and f1-score values based on the normalization or non-normalization of the dataset. The values are associated with a specific ranking mechanism and classifier used. As seen from the table, the RF classifier, when

RM	**Classifier**	**Non-Normalized Dataset**		**Normalized Dataset**	
		ACC (%)	**F1-Score**	**ACC (%)**	**F1-Score**
SP	RF	**98.97**	**0.98**	97.73	0.96
	XGB	98.51	0.98	**98.62**	**0.98**
WFI	RF	**98.96**	**0.98**	97.99	0.97
	XGB	98.71	0.98	**98.67**	**0.98**
	GB	91.05	0.83	71.13	0.09

Table 5.7: Performance Metrics for Different Ranking Mechanisms and Classifiers for Response Dataset

applied with the SP as well as WFI ranking mechanism, achieves superior results with an accuracy of 98.97% and 98.96% respectively as well as an F1-Score of 0.98 on the non-normalized dataset. Also, employing XGB with the SP and WFI ranking mechanism yields an accuracy of 98.62% and 98.67% respectively and an F1-Score of 0.98, surpassing other configurations.

Figure 5.12 shows the performance of classifiers based on Accuracy and F1-Score as each feature is added, and evaluated using the SPFS algorithm on the non-normalized Response dataset. As inferred from Figure 5.12, graph (a) reveals a constant increase in Accuracy with five features, while the F1-Score stabilizes after three features. Graphs (b), (c), and (d) exhibit a significant increase in performance with the addition of the first three features. Graph (e) demonstrates that the approach considered only two features, resulting in very poor performance. Figure 5.13 presents the performance of classifiers based on Accuracy and F1-Score as each feature is added, and evaluated using the SPFS algorithm on the normalized Response dataset. As inferred from Figure 5.13, graph (a) shows a spike in performance with the first

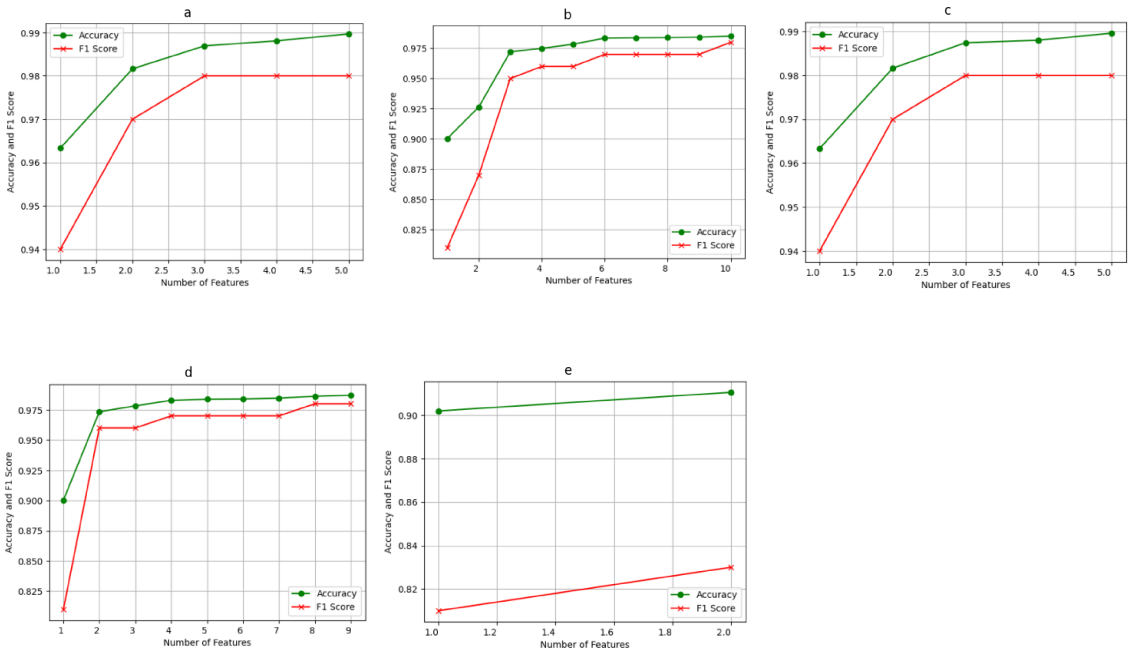


Figure 5.12: From left to right Graphs represent a. SP-SPFS-RF, b. SP-SPFS-XGB, c. WFI-SPFS-RF, d. WFI-SPFS-XGB, and e. WFI-SPFS-GB plot for Acc vs F1-Score on non-normalized Response dataset.

three features, followed by a stabilization of performance parameters. In graphs (b) and (d), the first four features contribute to the performance increase. In graphs (c) and (e), although Accuracy is high, the F1-Score remains low, indicating poor performance of the classifier.

5.5.2 Feature Selection using FFS

The FFS algorithm records the maximum attainable accuracy for a given subset of the feature set. It considers all the features in the feature set to attain that accuracy as opposed to SPFS, which terminates the features that don't meet the performance criteria. The subsequent sections present a comprehensive analysis and the associated data pertaining to the application of FFS on Command, Function, and Response datasets.

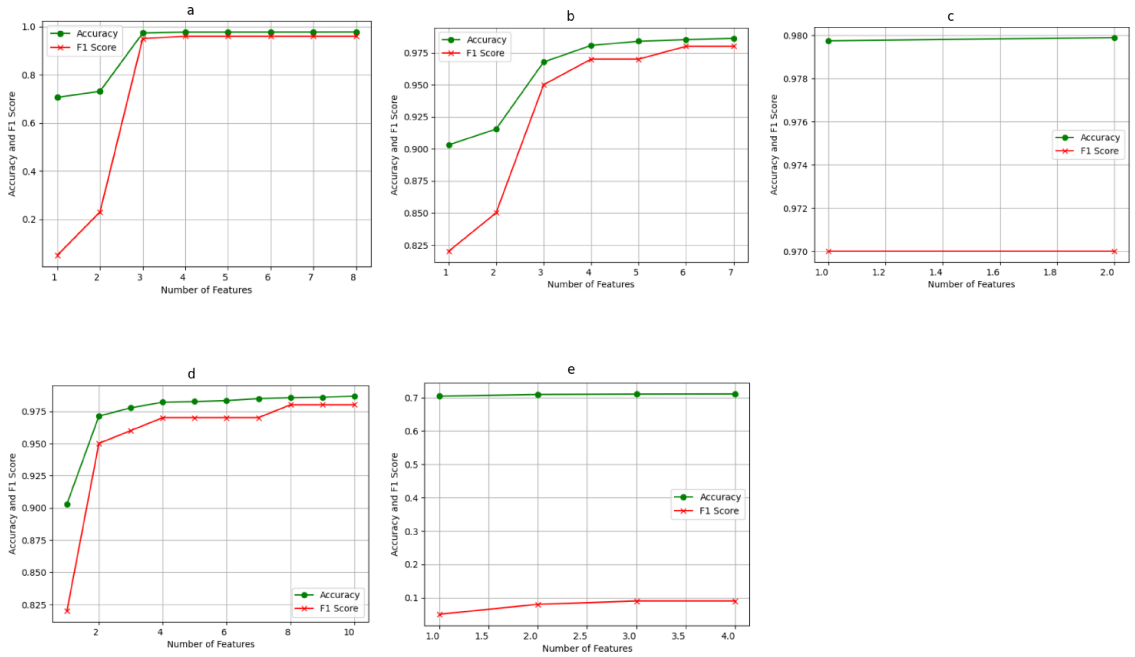


Figure 5.13: From left to right Graphs represent a. SP-SPFS-RF, b. SP-SPFS-XGB, c. WFI-SPFS-RF, d. WFI-SPFS-XGB, and e. WFI-SPFS-GB plot for Acc vs F1-Score on normalized Response dataset.

Command Dataset

Similar to SPFS, we employed tree-based classifiers namely RF, XGB, and GB to analyze the Command dataset using the FFS algorithm. The evaluation was based on performance metrics such as accuracy and F1 score, which are critical in assessing predictive models. The ensuing Table 5.8 presents the detailed results obtained, offering insights into the effectiveness of each classifier in predicting the desired output for the dataset. This table presents the two sets of Accuracy and F1-Score values based on the normalization or non-normalization of the dataset. The values are associated with a specific ranking mechanism and classifier used. As illustrated in the table, the Random Forest (RF) classifier demonstrates exceptional performance when utilized with both the SP and WFI ranking mechanisms, achieving an impressive accuracy of 98.87% and 98.97% respectively, alongside an F1-Score of 0.98 on the non-normalized dataset. Similarly, for the normalized Command dataset, the RF classifier, when applied to the feature sets derived from SP and WFI, performs

RM	Classifier	Non-Normalized Dataset		Normalized Dataset	
		ACC (%)	F1-Score	ACC (%)	F1-Score
SP	RF	**98.87**	**0.98**	**99.3**	**0.98**
	XGB	98.53	0.97	98.53	0.97
WFI	RF	**98.97**	**0.98**	**99.3**	**0.98**
	XGB	95.09	0.88	98.62	0.98
	GB	91.15	0.77	91.15	0.77

Table 5.8: Performance Metrics for Non-Normalized and Normalized Command Dataset using FFS

remarkably well, attaining an accuracy of 99.3% and an F1-Score of 0.98. Consequently, it can be inferred that the RF classifier yields the most promising results for the Command dataset.

Figure 5.14 shows the performance of classifiers based on Accuracy and F1-Score as

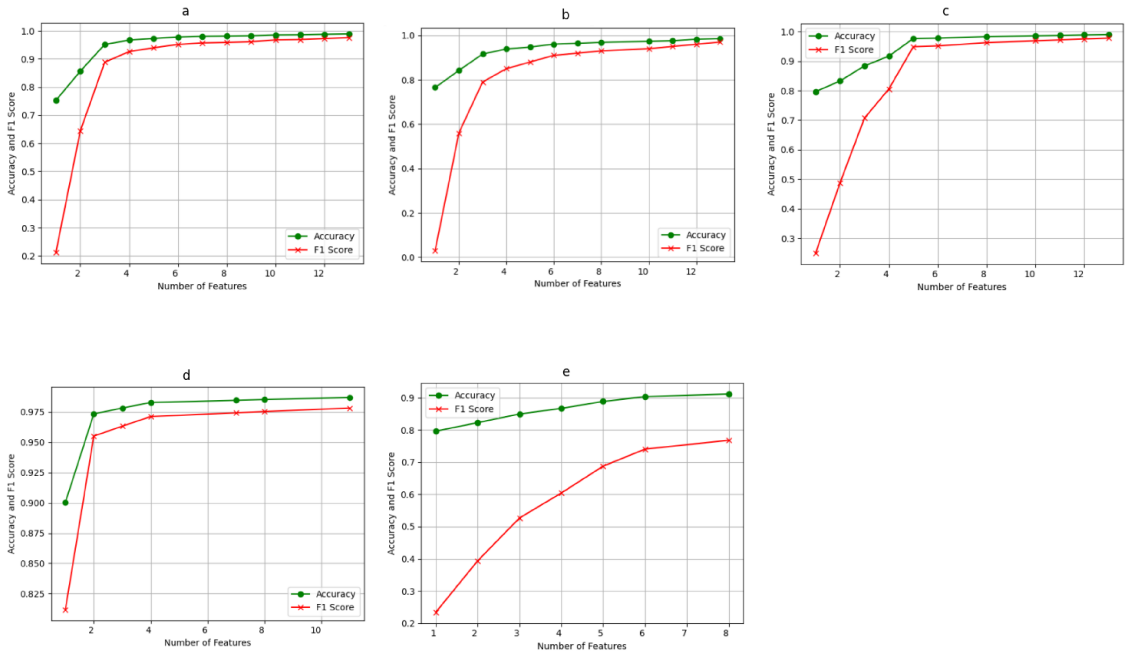


Figure 5.14: From left to right Graphs represent a. SP-FFS-RF, b. SP-FFS-XGB, c. WFI-FFS-RF, d. WFI-FFS-XGB, and e. WFI-FFS-GB plot for Acc vs F1-Score on non-normalized Command dataset using FFS

each feature is added, and evaluated using the FFS algorithm on the non-normalized Command dataset. As inferred from Figure 5.14, graphs (a), (b), and (c) reveal a constant increase in Accuracy with three features, while the F1-Score stabilizes

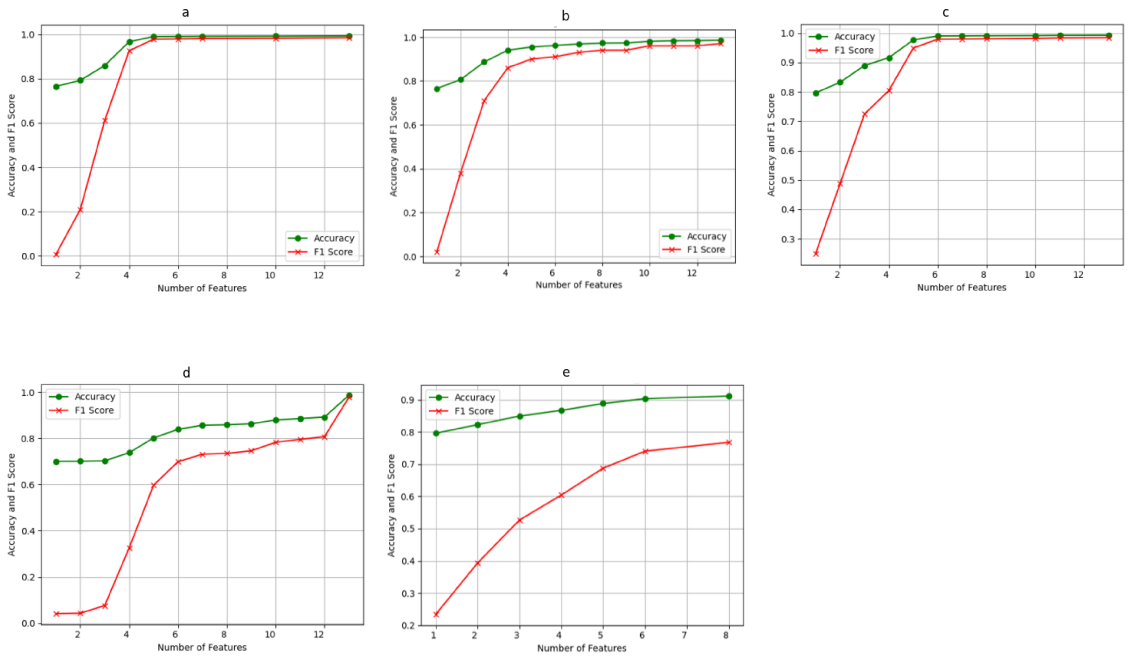


Figure 5.15: From left to right Graphs represent a.SP-FFS-RF, b.SP-FFS-XGB, c.WFI-FFS-RF, d.WFI-FFS-XGB, and e.WFI-FFS-GB plot for Acc vs F1-Score on normalized Command dataset using FFS

after three features. In graph (d), the Accuracy increase is significant for the first two features whereas, F1-Score increases for the first four features before stabilizing. Graph (e) demonstrates that even though accuracy is increasing lag in F1-Score is too prominent to be overlooked. Figure 5.15 presents the performance of classifiers based on Accuracy and F1-Score as each feature is added, and evaluated using the FFS algorithm on the normalized Command dataset. As inferred from Figure 5.15, graphs (a) and (b) show a spike in performance with the first three features, followed by a stabilization of performance parameters. In graph (c), the prominent increase in the performance parameters can be seen for the first five features and a plateau post that. Graph (d) shows a consistent spike in the performance parameters for the first six features and the last two features indicating that even though the features are arranged according to their importance from high to low they can still make a difference. In graph (e), although Accuracy is high, the F1-Score remains low, indicating poor performance of the classifier.

Function Dataset

We employed tree-based classifiers namely RF, XGB, and GB to analyze the Function dataset using the FFS algorithm. The evaluation was based on performance metrics such as accuracy and F1 score, which are critical in assessing predictive models. The ensuing Table 5.9 presents the detailed results obtained, offering insights into the effectiveness of each classifier in predicting the desired output for the dataset. This table presents the two sets of accuracy and f1-score values based on the normalization or non-normalization of the dataset. The values are associated with a specific ranking mechanism and classifier used. As illustrated in the table, the Random Forest (RF)

RM	**Classifier**	**Non-Normalized Dataset**		**Normalized Dataset**	
		ACC (%)	**F1-Score**	**ACC (%)**	**F1-Score**
SP	RF	**99.24**	**0.98**	**99.56**	**0.99**
	XGB	98.9	0.97	98.9	0.97
WFI	RF	**99.27**	**0.98**	**99.57**	**0.99**
	XGB	98.86	0.97	98.91	0.97
	GB	95.13	0.83	95.14	0.83

Table 5.9: Performance Metrics for Non-Normalized and Normalized Function Dataset using FFS

classifier demonstrates exceptional performance when utilized with both the SP and WFI ranking mechanisms, achieving an impressive accuracy of 99.24% and 99.27% respectively, alongside an F1-Score of 0.98 on the non-normalized Function dataset. Similarly, for the normalized Function dataset, the RF classifier, when applied to the feature sets derived from SP and WFI, performs remarkably well, attaining an accuracy of 99.56% and 99.57% with an F1-Score of 0.99. Consequently, it can be inferred that the RF classifier yields the most promising results for the Function dataset. Figure 5.16 shows the performance of classifiers based on accuracy and f1-score as each feature is added, and evaluated using the FFS algorithm on the non-normalized Function dataset. As inferred from Figure 5.16, graphs (a) and (b) reveal a constant increase in Accuracy with three features, while the f1-score in (b) is consistently increasing for almost all features. In graphs (c) and (d), we can see an increase in Accuracy for the first six to seven features whereas, for the f1-score it takes six to seven features for performance to plateau. Graph (e) shows an increase in accuracy as features are added but not so promising increase in F1-Score.

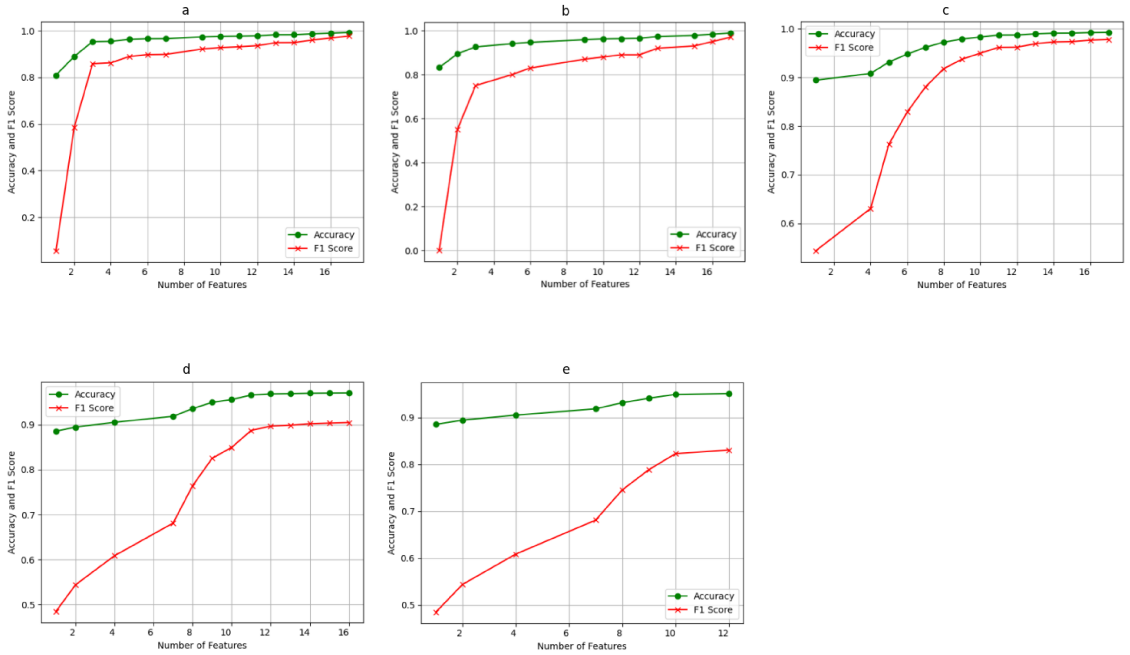


Figure 5.16: From left to right Graphs represent a.SP-FFS-RF, b.SP-FFS-XGB, c.WFI-FFS-RF, d.WFI-FFS-XGB, and e.WFI-FFS-GB plot for Acc vs F1-Score on non-normalized Function dataset using FFS.

Figure 5.17 presents the performance of classifiers based on accuracy and f1-Score as each feature is added, and evaluated using the FFS algorithm on the normalized Function dataset. As inferred from Figure 5.17, graphs (a), and (b) show a spike in performance with the first three features, followed by a stabilization of performance parameters. However, in the graph (b) it takes nine features for performance to stabilize. Even though (a), (b), and (c) attain similar kinds of accuracy and f1-score, the number of features it took to attain that varies in all approaches. In graph (d), it takes about seven to eight features to see a continuous increase, especially in f1-score before the performance plateaus. In graph (e), although Accuracy is high, the f1-score remains low, indicating poor performance of the classifier.

Response Dataset

We employed tree-based classifiers namely RF, XGB, and GB to analyze the Response dataset using the FFS algorithm. The evaluation was based on performance metrics such as accuracy and F1 score, which are critical in assessing predictive

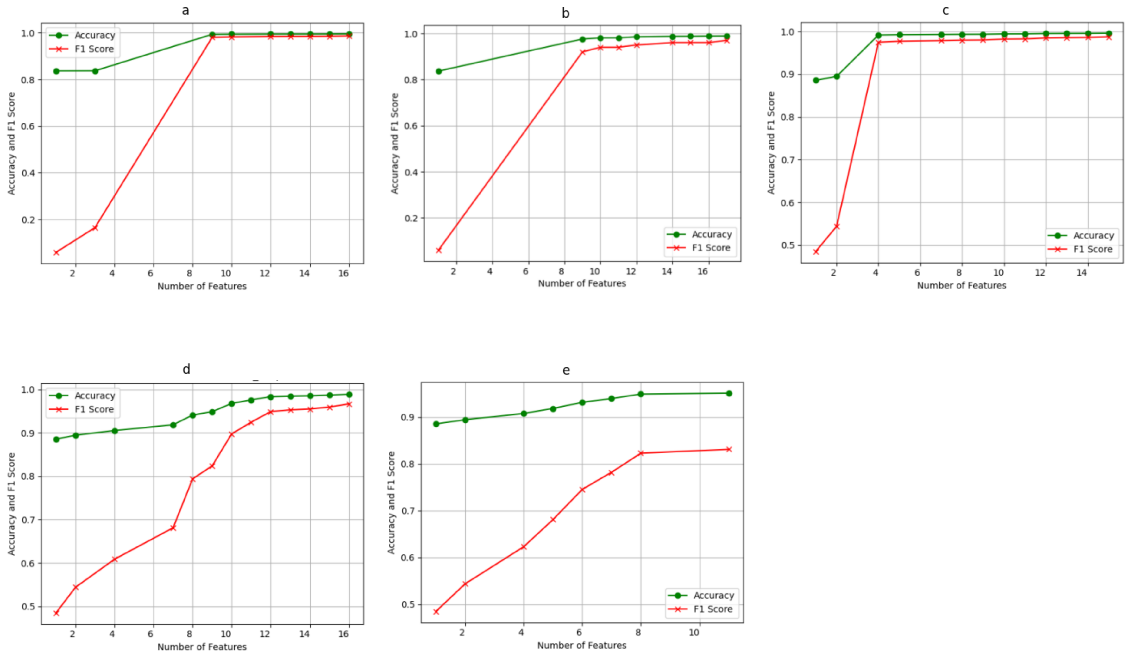


Figure 5.17: From left to right Graphs represent a. SP-FFS-RF, b. SP-FFS-XGB, c. WFI-FFS-RF, d. WFI-FFS-XGB, and e. WFI-FFS-GB plot for Acc vs F1-Score on normalized Function dataset using FFS

models. The ensuing Table 5.10 presents the detailed results obtained, offering insights into the effectiveness of each classifier in predicting the desired output for the dataset. This table presents the two sets of accuracy and f1-score values based on the normalization or non-normalization of the dataset. The values are associated with a specific ranking mechanism and classifier used. As illustrated in the table,

RM	**Classifier**	**Non-Normalized Dataset**		**Normalized Dataset**	
		ACC (%)	**F1-Score**	**ACC (%)**	**F1-Score**
SP	RF	98.65	0.98	97.79	0.96
	XGB	98.58	0.98	**98.7**	**0.98**
WFI	RF	**98.84**	**0.98**	97.97	0.97
	XGB	**98.72**	**0.98**	**98.72**	**0.98**
	GB	90.19	0.81	71.07	0.09

Table 5.10: Performance Metrics for Non-Normalized and Normalized Response Dataset using FFS

the RF and XGB classifier demonstrates exceptional performance when utilized with both the WFI ranking mechanisms, achieving an accuracy of 98.84% and 98.72%

respectively, alongside an f1-score of 0.98 on the non-normalized Response dataset. Similarly, for the normalized Response dataset, the performance of the XGB classifier using SP and WFI ranking mechanism appears most promising. It attains nearly the same accuracy of 98.7% and 98.72% with an F1-Score of 0.98.

Figure 5.18 shows the performance of classifiers based on accuracy and f1-score as

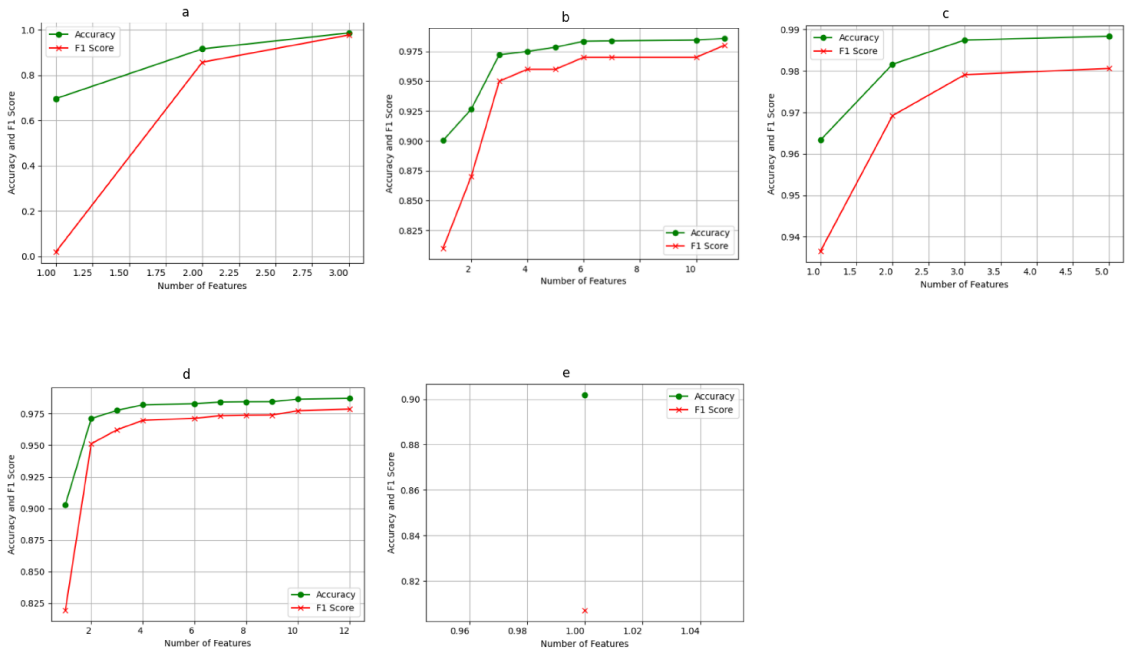


Figure 5.18: From left to right Graphs represent a.SP-FFS-RF, b.SP-FFS-XGB, c.WFI-FFS-RF, d.WFI-FFS-XGB, and e.WFI-FFS-GB plot for Acc vs F1-Score on non-normalized Response dataset using FFS

each feature is added, and evaluated using the FFS algorithm on the non-normalized Response dataset. As inferred from Figure 5.18, graphs (a) and (c) show a considerable performance with just three features. As seen in graphs (b) and (d), the Accuracy and F1-Score keep increasing for the first three features and plateaus post that. Notably, WFI-FFS-GB considers only a single feature, thus it could not be included in these graphical representations. Figure 5.19 presents the performance of classifiers based on Accuracy and F1-Score as each feature is added, and evaluated using the FFS algorithm on the normalized Response dataset. As inferred from Figure 5.19, graph (a) shows a considerable increase in Accuracy and F1-Score with just two features. Even though SP-FFS-RF performs using just four features, it performs

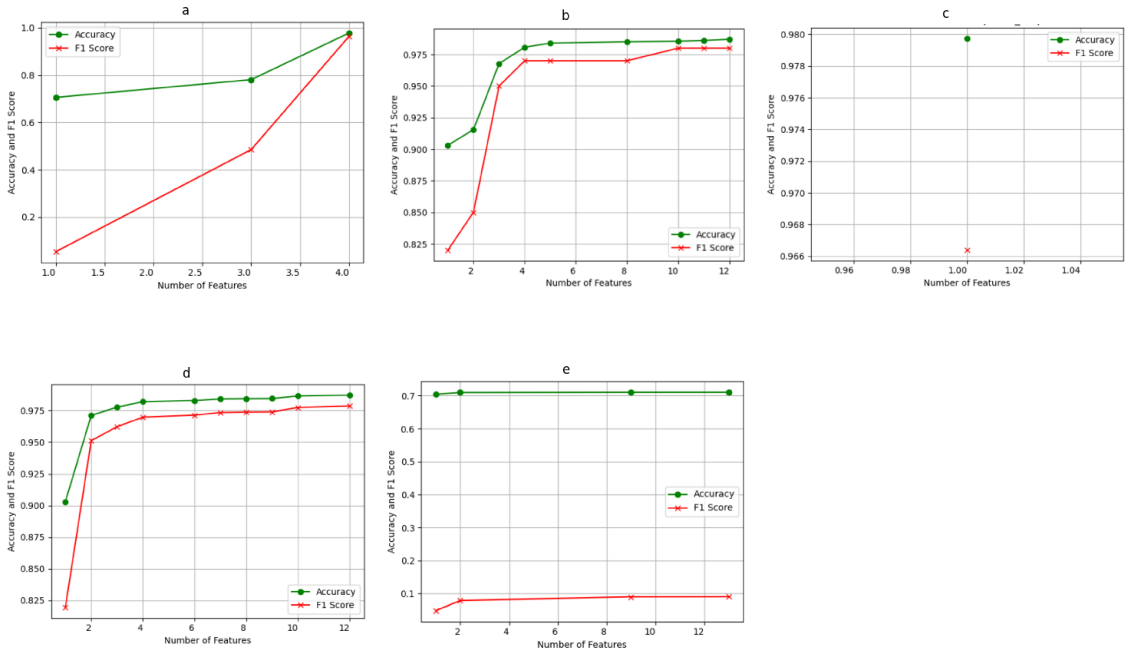


Figure 5.19: From left to right Graphs represent a.SP-FFS-RF, b.SP-FFS-XGB, c.WFI-FFS-RF, d.WFI-FFS-XGB, and e.WFI-FFS-GB plot for Acc vs F1-Score on normalized Response dataset using FFS

close to the highest-performing approach for this dataset. Graphs (b) and (d) show a spike in performance with the first four features, followed by a stabilization of performance parameters. In graph (e), although Accuracy is high, the F1-Score remains low, indicating poor performance of the classifier. Notably, WFI-FFS-RF considers only a single feature, thus it could not be included in these graphical representations.

5.6 Results for Evaluating Classifier Performance Using Cross-Validation on Selected Feature Sets

To validate the results obtained from the Feature Selection phase, we implemented the 10-fold Cross-Validation. We compared performance across multiple classifiers based on metrics including Accuracy, Number of Features used, Execution Time, F1-Score, Precision, Recall, False Positive Rate (FPR), False Negative Rate (FNR), True Positive Rate (TPR), and True Negative Rate (TNR). The following sections

present the experimental results for each dataset separately.

5.6.1 Command Dataset

Table 5.11 presents the outcomes obtained from applying Cross Validation to the feature set derived from SPFS for both the non-normalized and normalized Command datasets across various ranking mechanisms. As depicted in Table 5.11, the SP-RF and WFI-RF approach demonstrates superior performance over the others on the Command dataset. The SP-RF approach utilizes 13 features to achieve an accuracy of 98.86% with a training time of 208.92 seconds whereas the WFI-RF approach uses 11 features in 150.67 seconds to attain an accuracy of 98.78%. However, SP-RF has better precision, recall, and f1-score in comparison to WFI-RF. Therefore, it can be inferred that for the non-normalized Command dataset, SP-RF performs better. In contrast, on the normalized Command dataset, WFI-RF exhibits superior performance with an accuracy of 99.27%, a training time of 233.87 seconds, and employing 12 features. Notably, SP-RF achieves a comparable accuracy of 99.24% with significantly reduced training time and fewer features. Given the emphasis on time-critical infrastructure in our study, SP-RF emerges as the superior performer. Table 5.12 presents detailed results following the application of cross-validation to

Performance Metrics	**Command**				**Command_Preprocessed**			
	SP-RF	**SP-XGB**	**WFI-RF**	**WFI-XGB**	**SP-RF**	**SP-XGB**	**WFI-RF**	**WFI-XGB**
# of Features	**13**	12	**11**	12	**10**	13	**12**	11
Training time (s)	**208.92**	14.61	**150.67**	10.66	**179.13**	13.53	**233.87**	11.48
Accuracy (%)	**98.86**	98.18	**98.78**	98.18	**99.24**	98.37	**99.27**	97.91
Mean Precision	0.994	0.991	0.991	0.991	0.997	0.994	0.998	0.99
Mean Recall	0.958	0.932	0.957	0.932	0.971	0.937	0.972	0.922
Mean F1 Score	0.976	0.961	0.974	0.961	0.984	0.965	0.985	0.954
FPR	0.002	0.003	0.003	0.003	0.001	0.002	0.001	0.003
FNR	0.042	0.068	0.043	0.068	0.029	0.063	0.028	0.078
TPR	0.958	0.932	0.957	0.932	0.971	0.937	0.972	0.922
TNR	0.998	0.997	0.997	0.997	0.999	0.998	0.999	0.997
Feature Selection Time (s)	4.4768	0.9952	3.9931	0.9354	4.7174	0.9757	6.3348	0.8806

Table 5.11: Performance Metrics for Command and Command_Preprocessed using SPFS

the feature set derived from FFS for both the non-normalized and normalized Command datasets across different ranking mechanisms. Analysis reveals that for the non-normalized Command dataset, SP-RF and WFI-RF exhibit closely competitive performance. SP-RF achieves 98.86% accuracy in 220.91 seconds using 13 features, while WFI-RF achieves 98.88% accuracy in 237.66 seconds, also with 13 features.

The differences in other parameters are minimal. Hence, SP-RF is concluded to outperform WFI-RF in terms of execution time in this scenario. For the normalized Command dataset, SP-RF achieves an accuracy of 99.32% in 253.98 seconds using 13 features and WFI-RF achieves an accuracy of 99.31% in 252.24 seconds using 13 features as well. The computation time does not differ significantly as well as the total features used are constant across both the approaches hence, it can be deduced that SP-RF outperforms the other approaches.

Performance Metrics	**Command**				**Command_Preprocessed**			
	SP-RF	**SP-XGB**	**WFI-RF**	**WFI-XGB**	**SP-RF**	**SP-XGB**	**WFI-RF**	**WFI-XGB**
# of Features	**13**	13	**13**	12	**13**	13	**13**	13
Training time (s)	**220.91**	13.26	**237.66**	4.92	**253.98**	12.66	**252.24**	9.15
Accuracy(%)	**98.86**	98.36	**98.88**	98.1	**99.32**	98.37	**99.31**	98.67
Mean Precision	0.994	0.993	0.994	0.992	0.998	0.994	0.998	0.991
Mean Recall	0.958	0.937	0.959	0.928	0.973	0.937	0.973	0.965
Mean F1 Score	0.976	0.965	0.976	0.959	0.986	0.965	0.985	0.978
TPR	0.958	0.937	0.959	0.928	0.973	0.937	0.973	0.965
TNR	0.998	0.998	0.998	0.998	0.999	0.998	0.999	0.996
FPR	0.002	0.002	0.002	0.002	0.001	0.002	0.001	0.004
FNR	0.042	0.063	0.041	0.072	0.027	0.063	0.027	0.035
Feature Selection Time (s)	4.505289	0.967567	4.712730	0.924249	5.396931	1.444020	4.955189	0.975510

Table 5.12: Performance Metrics for Command and Command_Preprocessed using FFS

5.6.2 Function Dataset

Table 5.13 summarizes Cross Validation results using SPFS feature sets for both non-normalized and normalized Function datasets across different ranking mechanisms. In the non-normalized dataset, WFI-RF and SP-RF achieve similar accuracies of 99.37% and 99.36%, respectively, with comparable training times and feature counts. However, in the normalized dataset, WFI-RF and SP-RF stand out with 99.61% and 99.49% accuracy respectively using 14 and 11 features with a training time of 236.79 s and 390.17 s. Table 5.14 presents Cross Validation outcomes using FFS feature sets for both non-normalized and Normalized Function datasets across various ranking mechanisms. In the non-normalized dataset, SP-RF and WFI-RF perform equally well, achieving an accuracy of 99.37%. However, WFI-RF demonstrates lower training time compared to SP-RF, indicating superior performance in this scenario. In the normalized dataset, both classifiers achieve similar results except for training time, where WFI-RF again outperforms SP-RF.

Performance Metrics	Function				Function_Preprocessed			
	SP-RF	SP-XGB	WFI-RF	WFI-XGB	SP-RF	SP-XGB	WFI-RF	WFI-XGB
# of Features	**15**	14	**15**	13	**11**	12	**14**	13
Training time(s)	**294.34**	23.71	**290.27**	24.63	**390.17**	20.43	**236.79**	23.92
Accuracy(%)	**99.36**	94.08	**99.37**	98.74	**99.49**	98.16	**99.61**	98.74
Mean Precision	0.991	0.936	0.991	0.987	0.99	0.979	0.992	0.989
Mean Recall	0.971	0.699	0.972	0.938	0.98	0.912	0.985	0.937
Mean F1 Score	0.981	0.8	0.981	0.962	0.985	0.944	0.988	0.962
FPR	0.981	0.01	0.002	0.002	0.002	0.004	0.002	0.002
FNR	0.029	0.301	0.028	0.062	0.02	0.088	0.015	0.063
TPR	0.971	0.699	0.972	0.938	0.98	0.912	0.985	0.937
TNR	0.998	0.997	0.998	0.998	0.998	0.996	0.998	0.998
Feature Selection Time (s)	7.284817	2.031957	7.244934	1.901097	9.539596	4.524569	6.706321	1.849505

Table 5.13: Performance Metrics for Function and Function_Preprocessed using SPFS

Performance Metrics	Function				Function_Preprocessed			
	SP-RF	SP-XGB	WFI-RF	WFI-XGB	SP-RF	SP-XGB	WFI-RF	WFI-XGB
# of Features	**17**	17	**17**	17	**16**	17	**15**	16
Training time(s)	**294.05**	30.76	**287.4**	23.04	**402.39**	26.501096	**392.06**	17.32
Accuracy(%)	**99.37**	98.91	**99.37**	98.91	**99.65**	98.92	**99.66**	98.94
Mean Precision	0.991	0.991	0.991	0.991	0.994	0.992	0.995	0.992
Mean Recall	0.972	0.945	0.972	0.945	0.985	0.944	0.985	0.945
Mean F1 Score	0.981	0.967	0.981	0.967	0.990	0.967	0.990	0.968
TPR	0.972	0.945	0.972	0.945	0.985	0.944	0.985	0.945
TNR	0.998	0.998	0.998	0.998	0.999	0.998	0.999	0.998
FPR	0.002	0.002	0.002	0.002	0.001	0.002	0.001	0.002
FNR	0.028	0.055	0.028	0.055	0.015	0.056	0.015	0.055
Feature Selection Time (s)	8.442460	2.116954	9.470311	2.209312	9.761231422	2.140021	7.701041	1.993298

Table 5.14: Performance Metrics for Function and Function_Preprocessed using FFS

5.6.3 Response Dataset

Table 5.15 presents the results derived from the application of cross-validation on the feature set obtained from SPFS for both non-normalized and normalized Response datasets, evaluated across different ranking mechanisms. Analyzing the data in Table 5.15, the WFI-RF method achieves an accuracy of 98.98% with a training time of 149.43 seconds, utilizing 5 features. Conversely, the SP-RF method attains a comparable accuracy of 98.97% within a reduced training time of 117.8 seconds, using the same number of features. This indicates that SP-RF delivers nearly identical accuracy to WFI-RF but in a shorter duration. For the normalized Response dataset, the maximum accuracy achieved is 98.67% in 12.48 seconds, employing 10 features. An additional consideration is that the SP-XGB method achieves a similar accuracy with a marginal difference of 0.05%, utilizing fewer features, nearly equivalent training time, and significantly reduced feature selection time. Therefore, SP-XGB is deemed the superior performer when considering various factors such as feature space, number of features used, feature selection time, training time, and accuracy.

Table 5.16 presents the results obtained from the application of cross-validation on

Performance Metrics	Response				Response_Preprocessed			
	SP-RF	SP-XGB	WFI-RF	WFI-XGB	SP-RF	SP-XGB	WFI-RF	WFI-XGB
# of Features	**5**	10	**5**	9	8	**7**	2	**10**
Training time (s)	**117.8**	14.84	**149.43**	12.35	126.4	**12.4**	396.17	**12.48**
Accuracy (%)	**98.97**	98.61	**98.98**	98.67	97.64	**98.62**	97.88	**98.67**
Mean Precision	0.991	0.991	0.991	0.991	0.964	0.99	0.965	0.991
Mean Recall	0.975	0.963	0.975	0.965	0.958	0.964	0.965	0.965
Mean F1 Score	0.983	0.977	0.983	0.978	0.961	0.977	0.965	0.978
FPR	0.004	0.004	0.004	0.004	0.016	0.004	0.015	0.004
FNR	0.025	0.037	0.025	0.035	0.042	0.036	0.035	0.035
TPR	0.975	0.963	0.975	0.965	0.958	0.964	0.965	0.965
TNR	0.996	0.996	0.996	0.996	0.984	0.996	0.985	0.996
Feature Selection Time (s)	3.3507	1.0822	3.3958	1.0036	3.9804	0.9457	9.7439	3.8750

Table 5.15: Performance Metrics for Response and Response_Preprocessed using SPFS

the feature set derived from FFS for both non-normalized and normalized Response datasets, evaluated across various ranking mechanisms. Analyzing the data in Table 5.16, it is observed that SP-RF and WFI-RF are closely matched in performance for the non-normalized Response dataset. WFI-RF achieves an accuracy of 98.89% with a training time of 217.23 seconds using 5 features, whereas SP-RF attains an accuracy of 98.78% with a significantly shorter training time of 140.57 seconds using 3 features. The performance across other parameters is very similar for both methods. Hence, SP-RF is considered superior due to its use of fewer features, shorter training time, and reduced feature selection time while maintaining comparable accuracy. For the normalized Response dataset, WFI-XGB outperforms other approaches, achieving an accuracy of 98.71% in 8.99 seconds using 12 features. Additionally, WFI-XGB demonstrates superiority across other evaluated parameters.

Performance Metrics	Response				Response_Preprocessed			
	SP- RF	SP- XGB	WFI- RF	WFI- XGB	SP- RF	SP- XGB	WFI-RF	WFI-XGB
# of Features	**3**	11	**5**	**12**	4	**12**	1	**12**
Training time(s)	**140.58**	13.7	**217.23**	8.99	442.57	**14.13**	1167.44	**8.99**
Accuracy(%)	**98.78**	98.55	**98.89**	98.71	97.71	**98.63**	97.90	**98.71**
Mean Precision	0.989	0.990	0.990	0.991	0.964	0.991	0.966	0.991
Mean Recall	0.971	0.961	0.973	0.966	0.960	0.963	0.965	0.966
Mean F1 Score	0.980	0.976	0.981	0.978	0.962	0.977	0.965	0.978
TPR	0.971	0.961	0.973	0.966	0.960	0.963	0.965	0.966
TNR	0.995	0.996	0.996	0.996	0.984	0.996	0.985	0.996
FPR	0.005	0.004	0.004	0.004	0.016	0.004	0.015	0.004
FNR	0.029	0.039	0.027	0.034	0.040	0.037	0.035	0.034
Feature Selection Time (s)	4.826	1.079	5.602	1.038	5.806	1.174	12.534	1.059

Table 5.16: Performance Metrics for Response and Response_Preprocessed using FFS

5.7 Final Results after comparative analysis

Based on the data collected and the inferences drawn, this section emphasizes the most appropriate approach for all datasets, considering ranking mechanisms and feature selection algorithms. The selection is based on parameters such as total execution time, the number of features selected, and F1-Score. Fig 5.20 and Fig 5.21 show the accuracy attained through cross-validation across all datasets and ranking mechanisms using FFS and SPFS respectively. The intensity of the shading correlates with accuracy, darker regions indicate higher accuracy. Similarly, Fig 5.22 and Fig 5.23 illustrate the total execution time required for each approach measured in seconds using FFS and SPFS respectively, with darker areas signifying shorter execution times. The total execution time is defined as:

Total Execution Time = Pre-Processing Time + Feature Ranking Time + Feature Selection Time + 10-fold Cross-Validation time

As observed from Fig 5.20, Fig 5.21, Fig 5.22 and Fig 5.23 it is evident that the normalization of the Command and Function datasets enhances performance. Conversely, for the Response dataset, superior results were obtained without normalization. Furthermore, the highest-performing methods within the FFS framework marginally outperform those of SPFS in terms of accuracy. However, considering total execution time, SPFS demonstrates a clear superiority over FFS. Given our objective to identify methods suitable for time-critical systems, it can be concluded that SPFS is the preferable approach in this context.

Now that we have shortlisted the top two performing approaches and the optimal version of the dataset(Normalized or Non-Normalized) we proceed towards selecting the best approaches for each dataset. Fig 5.24 illustrates the Number of features used, Total Execution Time, and Accuracy attained for the top two performing methods for both FFS and SPFS. As observed from Fig 5.24, using the SP and WFI ranking mechanism, the RF classifier with the FFS Feature Selection algorithm achieves the highest accuracy for the Normalized Command dataset. However, its drawback is the extensive execution time, necessitating 13 features, which is the highest among the methods considered, rendering it unsuitable for time-critical applications. Notably, with the SP ranking mechanism, the RF classifier using the SPFS algorithm as the feature selection technique achieves the shortest execution time while maintaining

comparable accuracy with only 10 features. Therefore, for the normalized Command dataset, the combination of the SP ranking mechanism, RF classifier, and SPFS feature selection algorithm can be deemed the most effective approach.

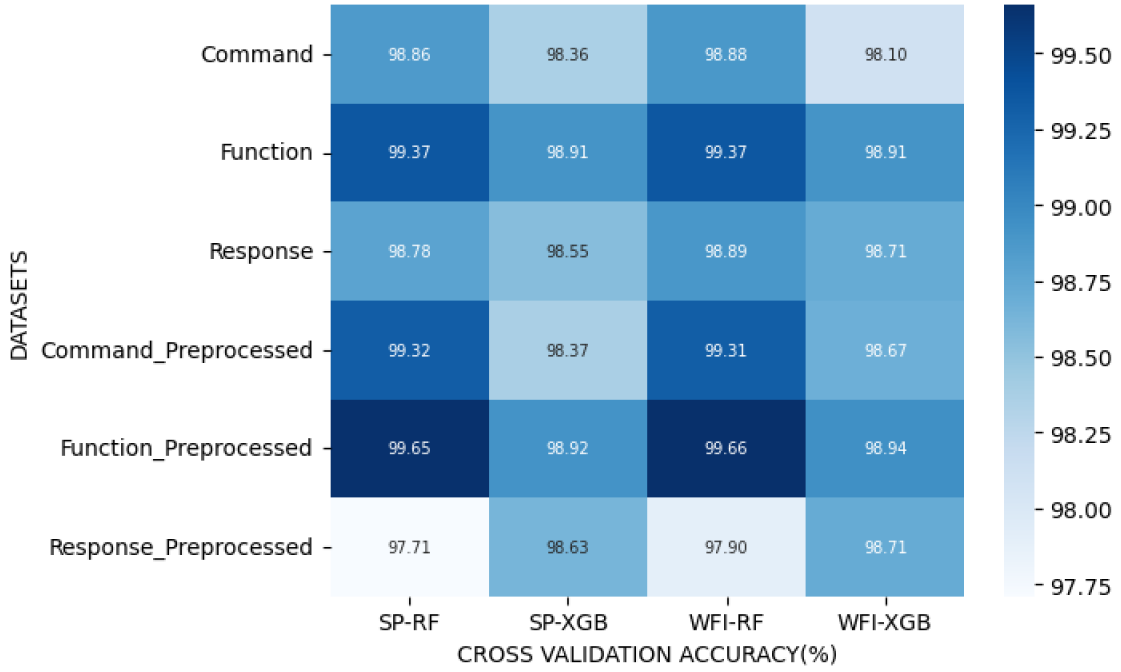


Figure 5.20: Cross Validation accuracy across different datasets, ranking mechanism using FFS

Fig 5.25 illustrates the Number of features used, Total Execution Time, and Accuracy attained for the top two performing methods using FFS and SPFS for the Function dataset. As observed from Fig 5.25, the combination of the FFS algorithm with the RF classifier, utilizing SP and WFI ranking mechanisms, provides superior results in terms of accuracy while employing all 16 features. However, the total execution time is significantly high, rendering it impractical for time-critical systems. Therefore, it can be concluded that the combination of the WFI and SP ranking mechanism, RF classifier, and SPFS feature selection algorithm achieves nearly equivalent accuracy to that of FFS while using fewer features and requiring less training time. The WFI-SPFS-RF approach yields a comparable accuracy with FFS using far less execution time and lesser number of features. Therefore, for the normalized Function dataset, WFI-SPFS-RF is an optimal approach.

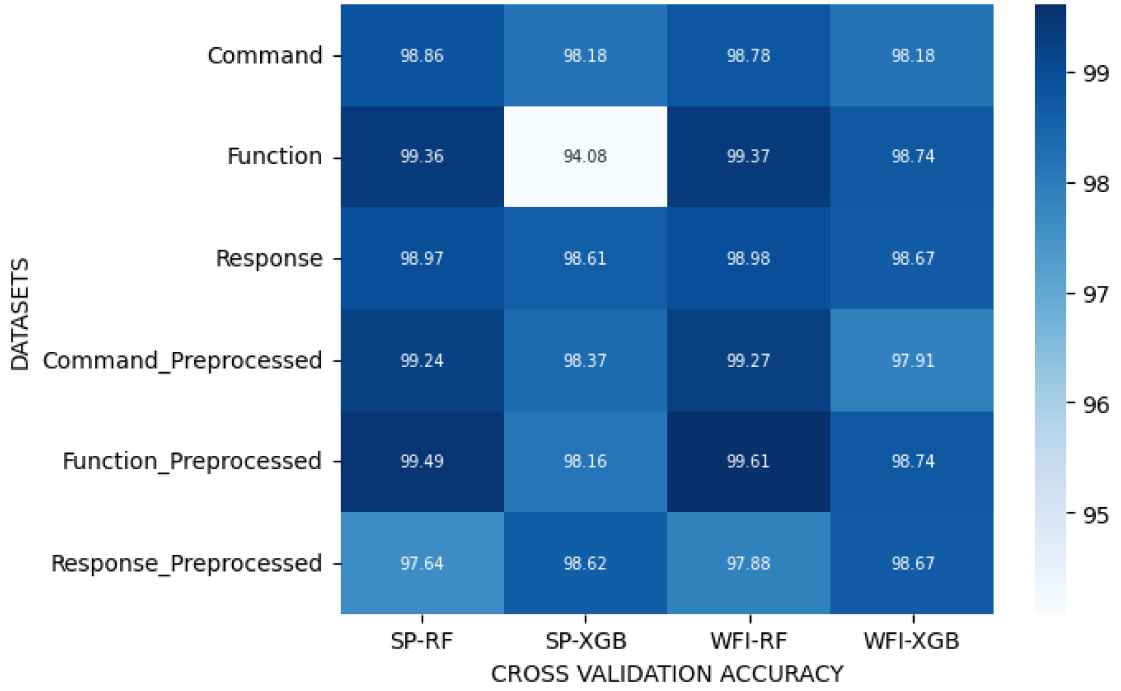


Figure 5.21: Cross Validation accuracy across different datasets, ranking mechanism using SPFS

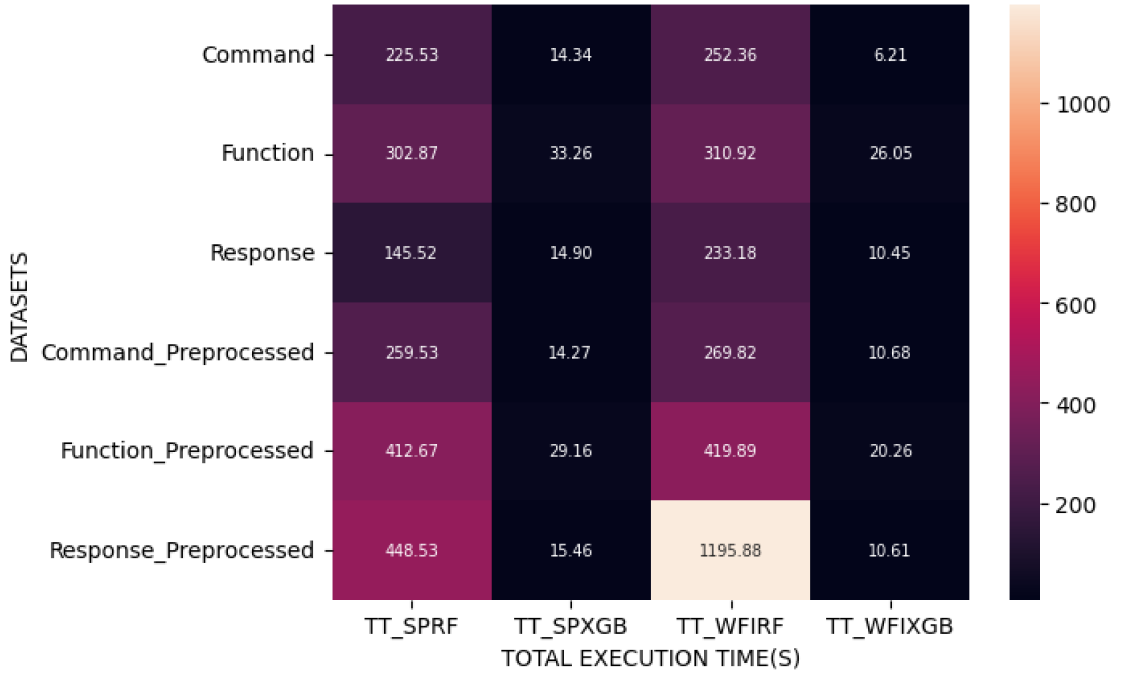


Figure 5.22: Total Execution Time across different datasets, ranking mechanism using FFS

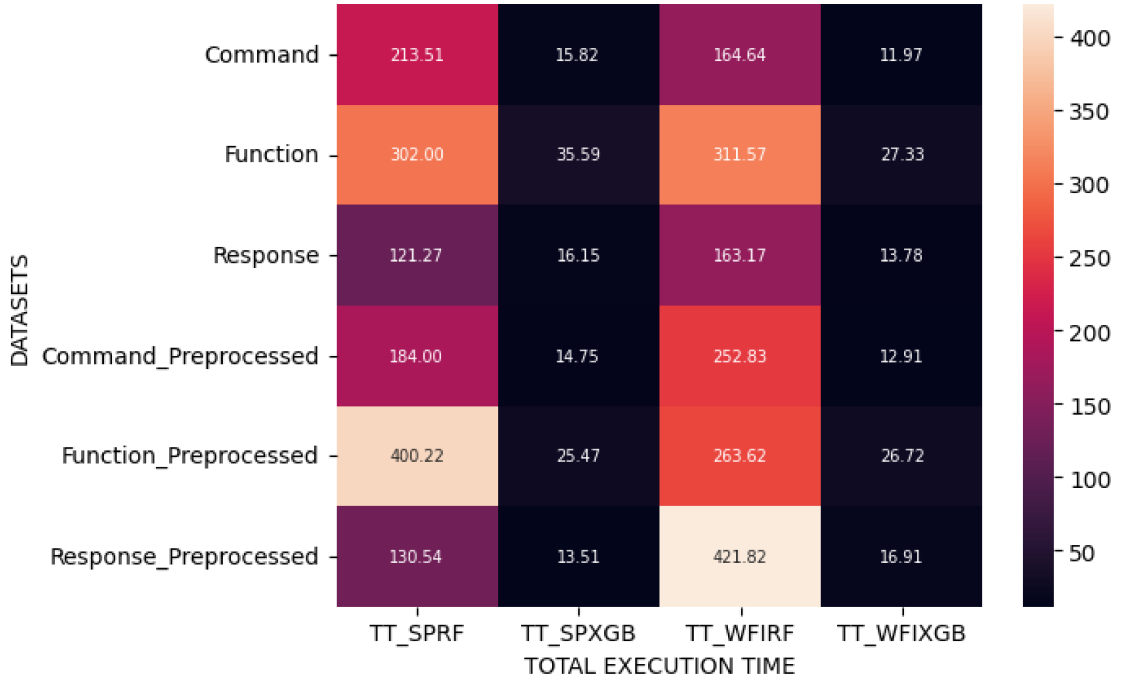


Figure 5.23: Total Execution Time across different datasets, ranking mechanism using SPFS

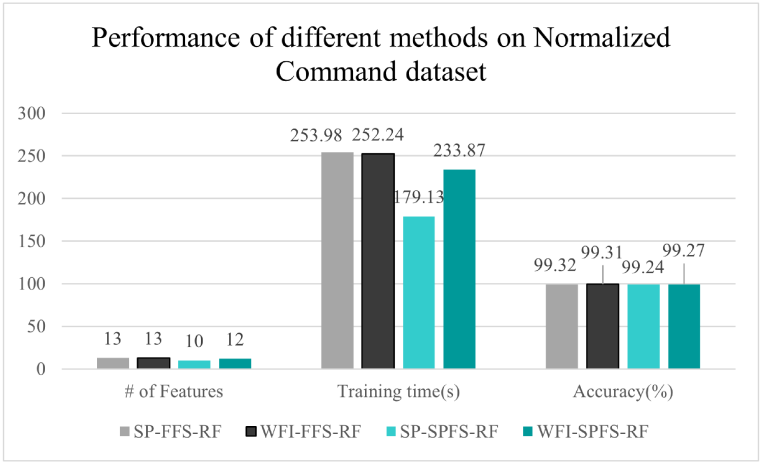


Figure 5.24: Final Classifiers and Ranking Mechanisms for Normalized Command dataset

Fig 5.26 illustrates the Number of features used, Total Execution Time, and Accuracy attained for the top two performing methods using FFS and SPFS for the Response dataset. As observed from Fig 5.26, using SP and WFI ranking mechanism, RF classifier, and SPFS feature selection algorithm yields the most promising performance. Both approaches acquire a similar kind of accuracy of nearly 99% using 5

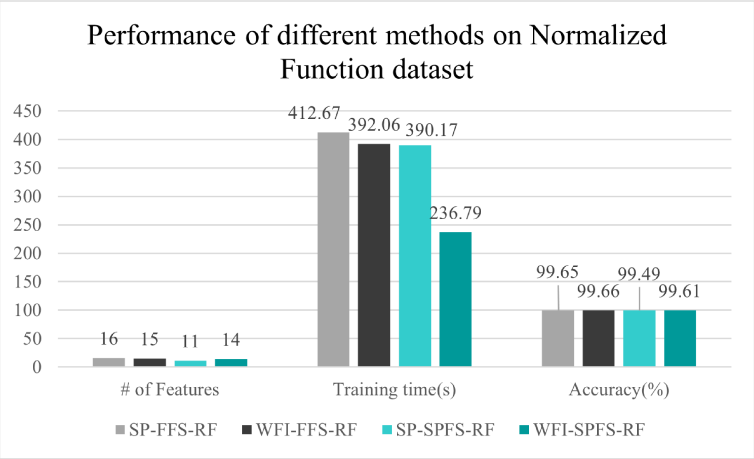


Figure 5.25: Final Classifiers and Ranking Mechanisms for Normalized Function dataset

features. However, the SP-SPFS-RF aaproach requires less execution time compared to WFI-SPFS-RF. Hence, the SP ranking mechanism, coupled with the RF classifier and SPFS feature selection algorithm, emerges as the optimal approach. Table 5.17

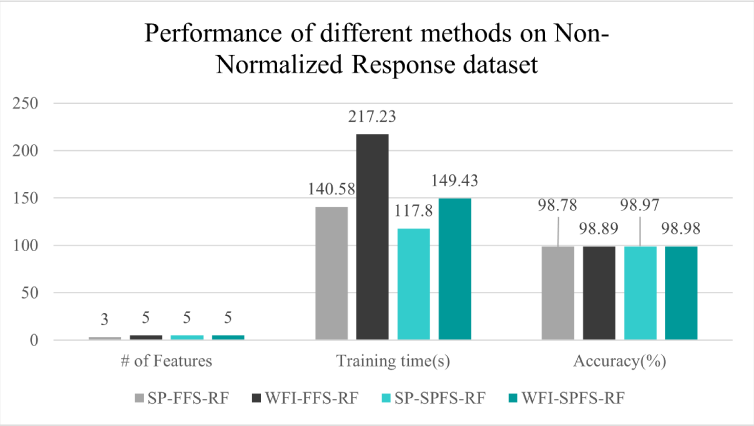


Figure 5.26: Final Classifiers and Ranking Mechanisms for Non-Normalized Response datasets

provides a more detailed view of best configurations and acquired performance parameters w.r.t each dataset. Based on the results acquired, we can conclude that for a system operating in a time-critical manner features should be ranked using the SP ranking mechanism, feature selection using the SPFS algorithm, and classification using the XGB classifier. Therefore, for time-critical systems SP-SPFS-XGB has consistently performed well with a very minimal execution time. However, if the aim is to get the best possible performance for sensitive systems where data protection

is critical; ranking using the SP ranking mechanism, feature selection using SPFS, and classification using RF should be performed.

CRITICAL INFRASTRUCTURE APPLICATIONS								
FFS	**Without-Preprocessing**				**With Preprocessing**			
	Best Method	**Features**	**F1-Score**	**Execution Time**	**Best Method**	**Features**	**F1-Score**	**Total Time**
Command	SP, RF	13	0.98	225.53	SP, RF	13	0.99	259.53
Function	SP, RF	17	0.98	302.87	SP, RF	16	0.99	412.67
Response	WFI, RF	5	0.98	233.18	WFI-XGB	12	0.98	10.61
SPFS	**Without-Preprocessing**				**With Preprocessing**			
	Best Method	**Features**	**F1-Score**	**Execution Time**	**Best Method**	**Features**	**F1-Score**	**Total Time**
Command	SP, RF	13	0.98	213.51	SP, RF	10	0.98	184
Function	SP, RF	15	0.98	302	WFI, RF	14	0.99	263.62
Response	SP, RF	5	0.98	121.27	SP, XGB	7	0.98	13.51
Best approach for highly available systems (response time is critical): With Pre-processing-SP-SPFS-XGB								
Best approach for sensitive systems (data is critical): With Pre-processing-SP-SPFS-RF								

Table 5.17: Comparison of different methods for Critical Infrastructure Applications on Command, Function, and Response dataset

Table 5.18 shows the comparison of our approach with the state-of-the-art research conducted. Our approach yields better performance for clusters Command and Response in contrast to the work done by [55] and attains almost similar accuracy for the Function cluster. Also, for the gas pipeline dataset we have compared our results with [55], [59], [61], [60], [56], and [21] who have worked on Binary classification for IDS.

Table 5.18: Comparison with the state-of-the-art approaches

Ref	**Preprocessing (Y/N)**	**FS (Y/N)**	**Feature Count**	**Best Model, Classification**	**Accuracy**	**F1-Score**	**Precision**	**Recall**	**Exec Time (Y/N)**
[55]	No	No	All (17)	Random Forest, Clusters and Multiclass	C: 97.73% F: 99.97% R: 97.73%	C: 0.977 F: 1 R: 0.977	C: 0.977 F: 1 R: 0.977	NA	No
[61]	No	Yes, k-means algorithm (k=2)	Not Mentioned	Ensemble Models, Binary and Multiclass	98.39%	0.983	0.984	0.983	No
[4]	Yes, Standardization	Yes, PCA, CCA, and ICA	Not Mentioned	Proposed Approach, Multiclass	97%	0.95	0.98	0.92	No
[59]	Yes, Min-Max Normalization, RBF kernel	No	All (17)	One-Class Support Vector Machines (OCSVM), Binary	99.13%	NA	NA	NA	No
[60]	No	No	All (17)	Decision Tree, Binary, Multiclass, and Specific	97.50%	0.975	0.975	0.975	No
[56]	Yes	No	All (17)	OCC-eSNN, Binary	98.82%	0.988	0.988	0.988	No
[21]	No	Yes,FDA and Cost matrix	17	Random Forest, Binary	97.80%	0.949	NA	NA	No
Our Work	**Yes, Min-Max**	**No**	**17**	**Random Forest**	**99.1%**	**0.986**	**0.991**	**0.981**	**4110.03 s**
		Yes	**14**	**SP-SPFS-DT**	**99.22%**	**0.982**	**0.986**	**0.978**	**24.43 s**
			10	**SP-SPFS-RF**	**C:99.24%**	**0.984**	**0.997**	**0.971**	**179.13 s**
			11	**WFI-SPFS-RF**	**F: 99.61%**	**0.988**	**0.99**	**0.98**	**239.79 s**
			5	**SP-SPFS-RF**	**R: 98.97%**	**0.983**	**0.991**	**0.975**	**117.8 s**

Chapter 6

Future Work

While significant strides have been made in securing gas pipeline systems through the application of ML techniques, challenges related to dataset imbalance persist, warranting further investigation. Addressing this imbalance could reveal whether the performance of the selected approaches can be enhanced further. It would be prudent to explore whether balancing the dataset leads to measurable improvements in accuracy and detection efficiency. Additionally, investigating the potential of alternative ML methodologies, including Deep Learning techniques, within the distinct clusters of the gas pipeline dataset could offer valuable insights for attack detection in ICSs.

Our novel approach also presents an opportunity to extend its application beyond binary classification to multi-class classification within the Gas Pipeline dataset, potentially broadening its utility in complex scenarios. Moreover, the integration of other feature selection techniques with our proposed feature ranking mechanisms could further evaluate the effectiveness of different approaches in detecting cyber-attacks within Critical Infrastructures (CIs).

To enhance the scalability and robustness of our methodology, it is crucial to test it against more complex datasets derived from other Industrial Control Systems (ICSs). Such testing would assess the approach's adaptability and efficacy across diverse operational contexts. Furthermore, evaluating the applicability of our approach in real-time scenarios is essential to ensure its practical relevance and reliability in dynamic environments.

Our approach also holds promise for global applications. By extending its use to other gas pipeline systems worldwide, facilitated through a cloud interface or an advanced data pipeline, we could enable continuous monitoring and timely updates of the classifiers. This would ensure that the system remains agile and responsive, consistently adapting to new data and evolving threats in real-time.

Chapter 7

Conclusion

Our innovative methodology, SP-SPFS, significantly diminishes the overall response time, rendering it exceptionally well-suited for real-time applications within Critical Infrastructures (CIs) including but not limited to gas pipeline systems. This reduction in response time is instrumental in enabling the prompt identification of cyber-attacks, thereby facilitating a more immediate and effective response. Consequently, this ensures the continuous and secure operation of CIs, safeguarding their functionality against potential threats. The robustness of our approach lies not only in its speed but also in its adaptability, making it a critical tool in the ongoing effort to protect vital infrastructure.

Building on this foundation, our work specifically focuses on identifying and evaluating various approaches that effectively reduce response time within an IDS tailored for the gas pipeline SCADA dataset and its three distinct clusters namely Command, Function, and Response. By conducting a comparative analysis of multiple ML classifiers, feature ranking mechanisms, and feature selection algorithms, we determined the optimal combinations for different systems—those that are time-sensitive and those that are performance-sensitive.

We evaluated the performance of five ML classifiers—DT, RF, GB, XGB, and NB on the Gas Pipeline dataset with and without feature selection. Our findings indicate that RF, GB, and XGB outperformed the others and are suitable for further evaluation based on the analysis done using the Gas Pipeline dataset. Our findings indicate that:

- Using the SP-SPFS-DT approach on a non-normalized gas pipeline dataset we attained an accuracy of 99.22% in 24.43s using 14 features.
- On a non-normalized gas pipeline dataset with no feature ranking and feature selection we observed that the RF classifier attained an accuracy of 99.1% in 4110.03s using 17 features.

The feature ranking and feature selection significantly reduce the total execution time of anomaly detection as well as increase the overall accuracy. For the three clusters: Command, Function, and Response we used our two feature ranking mechanisms, SP and WFI to establish the hierarchical arrangement of features. We then compared the performance across feature sets generated with these ranking mechanisms using feature selection algorithms, specifically FFS and our novel algorithm SPFS. Our results show that the SP and WFI approach with RF offers the best accuracy, while the SP approach with XGB provides a good balance of accuracy and reduced response time. To validate our results, we employed 10-fold cross-validation.
From the 10-fold cross-validation results, we observed that pre-processing the Command and Function datasets yields better performance, whereas the Response dataset performs better without pre-processing. The following presents the results deduced:

- In reference to the Command dataset, the SP-SPFS-RF method demonstrated superior performance, achieving an accuracy of 99.24% within 179.13 seconds, utilizing 10 features.

- In reference to the Function dataset, the WFI-SPFS-RF approach yielded the best results acquiring an accuracy of 99.61% in 236.79s using 14 features.

- In reference to the Response dataset, the SP-SPFS-XGB provides the best performance in terms of execution time taking 12.4s to attain an accuracy of 98.62% using 7 features. The SP-SPFS-RF approach provided the best results attaining an accuracy of 98.97% in 117.8s using 5 features. The difference in the accuracy attained is not significant in comparison to execution time. Hence, for the Response dataset, SP-SPFS-XGB is the optimal approach.

Our study highlights that for CIs where rapid response time is paramount, the SP-SPFS-XGB approach with a pre-processed dataset offers superior performance. Conversely, for performance-sensitive systems where maintaining data integrity is critical, the SP-SPFS-RF approach, also with a pre-processed dataset, proves to be more effective. These insights reinforce the adaptability and efficacy of our methodology across various operational scenarios within CIs.

Bibliography

[1] Darshana Upadhyay. *Security Solutions for Supervisory Control And Data Acquisition (Scada) Networks In Industrial Control Systems*. Phd thesis, Dalhousie University, April 2023. Available at http://hdl.handle.net/10222/82409.

[2] Ian Turnipseed. *A New SCADA Dataset for Intrusion Detection System Research A New SCADA Dataset for Intrusion Detection System Research*.

[3] Geeta Yadav and Kolin Paul. Architecture and security of scada systems: A review. *International Journal of Critical Infrastructure Protection*, 34:100433, 2021.

[4] Izhar Ahmed Khan, Dechang Pi, Zaheer Ullah Khan, Yasir Hussain, and Asif Nawaz. Hml-ids: A hybrid-multilevel anomaly prediction approach for intrusion detection in scada systems. *IEEE Access*, 7:89507–89521, 2019.

[5] Darshana Upadhyay and Srinivas Sampalli. Scada (supervisory control and data acquisition) systems: Vulnerability assessment and security recommendations. *Computers Security*, 89:101666, 2020.

[6] Oyeniyi Akeem Alimi, Khmaies Ouahada, Adnan M. Abu-Mahfouz, Suvendi Rimer, and Kuburat Oyeranti Adefemi Alimi. A review of research works on supervised learning algorithms for scada intrusion detection and classification. *Sustainability*, 13(17), 2021.

[7] Dimitrios Pliatsios, Panagiotis Sarigiannidis, Thomas Lagkas, and Antonios G. Sarigiannidis. A survey on scada systems: Secure protocols, incidents, threats and tactics. *IEEE Communications Surveys Tutorials*, 22(3):1942–1976, 2020.

[8] Jakapan Suaboot, Adil Fahad, Zahir Tari, John Grundy, Abdun Naser Mahmood, Abdulmohsen Almalawi, Albert Y. Zomaya, and Khalil Drira. A taxonomy of supervised learning for idss in scada environments. *ACM Comput. Surv.*, 53(2), apr 2020.

[9] Manar Alanazi, Abdun Mahmood, and Mohammad Jabed Morshed Chowdhury. Scada vulnerabilities and attacks: A review of the state-of-the-art and open issues. *Computers Security*, 125:103028, 2023.

[10] Slavica V. Boštjančič Rakas, Mirjana D. Stojanović, and Jasna D. Marković-Petrović. A review of research work on network-based scada intrusion detection systems. *IEEE Access*, 8:93083–93108, 2020.

[11] Sagarika Ghosh and Srinivas Sampalli. A survey of security in scada networks: Current issues and future challenges. *IEEE Access*, 7:135812–135831, 2019.

[12] Mohiuddin Ahmed, Adnan Anwar, Abdun Mahmood, Zubair Shah, and Michael Maher. An investigation of performance analysis of anomaly detection techniques for big data in scada systems. *EAI Endorsed Transactions on Industrial Networks And Intelligent Systems*, Volume 15:16, 05 2015.

[13] UCI Machine Learning Repository — archive.ics.uci.edu. https://archive.ics.uci.edu/dataset/129/kdd+cup+1998+data. [Accessed 01-08-2024].

[14] KDD Cup 1999 Data — kaggle.com. https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data. [Accessed 01-08-2024].

[15] Nsl-kdd — kaggle.com. https://www.kaggle.com/datasets/hassan06/nslkdd. [Accessed 01-08-2024].

[16] itrust labs_dataset info - itrust — itrust.sutd.edu.sg. https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/. [Accessed 01-08-2024].

[17] Tommy morris - industrial control system (ics) cyber attack datasets — sites.google.com. https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets. [Accessed 01-08-2024].

[18] Maia B. Cook. Additional Tennessee Eastman Process Simulation Data for Anomaly Detection Evaluation — dataverse.harvard.edu. https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/6C3JR1. [Accessed 01-08-2024].

[19] Abigail, Ryan, Hinne Hettema, and Kenneth Radke. Machine learning in industrial control system (ics) security: current landscape, opportunities and challenges. *Journal of Intelligent Information Systems*, 60(2):377–405, Oct 2022.

[20] Zsolt János Viharos, Krisztián Balázs Kis, Ádám Fodor, and Máté István Büki. Adaptive, hybrid feature selection (ahfs). *Pattern Recognition*, 116:107932, 2021.

[21] Abouzar Choubineh, David A. Wood, and Zahak Choubineh. Applying separately cost-sensitive learning and fisher's discriminant analysis to address the class imbalance problem: A case study involving a virtual gas pipeline scada system. *International Journal of Critical Infrastructure Protection*, 29:100357, 2020.

[22] Thomas Morris and Wei Gao. Industrial control system traffic data sets for intrusion detection research. volume 441, pages 65–78, 03 2014.

[23] Abdu Gumaei, Mohammad Mehedi Hassan, Shamsul Huda, Md. Rafiul Hassan, David Camacho, Javier Del Ser, and Giancarlo Fortino. A robust cyberattack detection approach using optimal features of scada power systems in smart grids. *Applied Soft Computing*, 96:106658, 2020.

[24] Deval Bhamare, Maede Zolanvari, Aiman Erbad, Raj Jain, Khaled Khan, and Nader Meskin. Cybersecurity for industrial control systems: A survey. *Computers Security*, 89:101677, 2020.

[25] A Daneels, Switzerland Geneva, Geneva Salter, and Switzerland. *WHAT IS SCADA?*

[26] What is a feature engineering? — IBM — ibm.com. https://www.ibm.com/topics/feature-engineering, 2024. [Accessed 01-08-2024].

[27] Meritshot. Standardization v/s Normalization — meritshot. https://medium.com/@meritshot/standardization-v-s-normalization-6f93225fbd84, 2023. [Accessed 01-08-2024].

[28] piyushagni5. Feature selection techniques in machine learning, Sep 2020.

[29] Summarizing quantitative data — Statistics and probability — Khan Academy — khanacademy.org. https://www.khanacademy.org/math/statistics-probability/summarizing-quantitative-data, 2023. [Accessed 01-08-2024].

[30] Statistics for Machine Learning: A Complete Guide — Simplilearn — simplilearn.com. https://www.simplilearn.com/tutorials/machine-learning-tutorial/statistics-for-machine-learning, 2021. [Accessed 01-08-2024].

[31] What Is Machine Learning (ML)? — IBM — ibm.com. https://www.ibm.com/topics/machine-learning, 2021. [Accessed 01-08-2024].

[32] 1.10. Decision Trees — scikit-learn.org. https://scikit-learn.org/stable/modules/tree.html#decision-trees, 2024. [Accessed 01-08-2024].

[33] RandomForestClassifier — scikit-learn.org. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. [Accessed 01-08-2024].

[34] 1.9. Naive Bayes — scikit-learn.org. https://scikit-learn.org/stable/modules/naive_bayes.html, 2024. [Accessed 01-08-2024].

[35] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232, 2001.

[36] What is XGBoost? — nvidia.com. https://www.nvidia.com/en-us/glossary/xgboost/, 2019. [Accessed 01-08-2024].

[37] Abhishek Jain. A Comprehensive Guide to Performance Metrics in Machine Learning — abhishekjainindore24. https://medium.com/@abhishekjainindore24/a-comprehensive-guide-to-performance-metrics-in-machine-learning, 2024. [Accessed 01-08-2024].

[38] Understanding the Confusion Matrix in Machine Learning - GeeksforGeeks — geeksforgeeks.org. https://www.geeksforgeeks.org/confusion-matrix-machine-learning/, 2017. [Accessed 01-08-2024].

[39] exida.com LLC. RISI - The Repository of Industrial Security Incidents — risidata.com. https://www.risidata.com/Database/event_date/asc/P210, 2015. [Accessed 01-08-2024].

[40] Major SCADA. 14 major scada hacks, Feb 2021.

[41] Evolution of ICS Attacks & Future Disruptive Events — Dragos — dragos.com. https://www.dragos.com/resources/whitepaper/evolution-of-ics-attacks-and-the-prospects-for-future-disruptive-events/, 2019. [Accessed 01-08-2024].

[42] The Hacker News. Elfin Hacking Group Targets Multiple U.S. and Saudi Arabian Firms — thehackernews.com. https://thehackernews.com/2019/03/apt33-cyber-espionage-hacking.html, 2019. [Accessed 01-08-2024].

[43] Hackers hit Norsk Hydro with ransomware. The company responded with transparency - Source — news.microsoft.com. https://news.microsoft.com/source/features/digital-transformation/hackers-hit-norsk-hydro-ransomware-company-responded-transparency/, 2023. [Accessed 01-08-2024].

[44] Hacker tries to poison water supply of Florida city — bbc.com. https://www.bbc.com/news/world-us-canada-55989843, 2021. [Accessed 01-08-2024].

[45] Reuters. Colonial pipeline halts all pipeline operations after cybersecurity attack. *Reuters*, 2021. Accessed: 2021-05-08.

[46] Kalyeena Makortoff. World's biggest meat producer jbs pays $11m cybercrime ransom, Jun 2021. The Guardian.

[47] Reuters. After 'godzilla attack!' u.s. warns about traffic-sign hackers. https://www.reuters.com/article/usa-hacking-traffic-idUKL1N0ON1WE20140606/, 2014. [Accessed 01-08-2024].

[48] Oyeniyi Akeem Alimi, Khmaies Ouahada, Adnan M Abu-Mahfouz, and Kuburat Oyeranti Adefemi Alimi. Empirical comparison of machine learning algorithms for mitigating power systems intrusion attacks. In *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–5. IEEE, 2020.

[49] Darshana Upadhyay, Jaume Manero, Marzia Zaman, and Srinivas Sampalli. Gradient boosting feature selection with machine learning classifiers for intrusion detection on power grids. *IEEE Transactions on Network and Service Management*, 18(1):1104–1116, 2020.

[50] Pallavi Arora, Baljeet Kaur, and Marcio Andrey Teixeira. Evaluation of machine learning algorithms used on attacks detection in industrial control systems. *Journal of The Institution of Engineers (India): Series B*, 102(3):605–616, 2021.

[51] Mehmet Turkoz, Sangahn Kim, Youngdoo Son, Myong K Jeong, and Elsayed A Elsayed. Generalized support vector data description for anomaly detection. *Pattern Recognition*, 100:107119, 2020.

[52] Fahd A Alhaidari and Ezaz Mohammed Al-Dahasi. New approach to determine ddos attack patterns on scada system using machine learning. In *2019 International conference on computer and information sciences (ICCIS)*, pages 1–6. IEEE, 2019.

[53] Andres Robles-Durazno, Naghmeh Moradpoor, James McWhinnie, and Gordon Russell. Real-time anomaly intrusion detection for a clean water supply system, utilising machine learning with novel energy-based features. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[54] S Shitharth et al. An enhanced optimization based algorithm for intrusion detection in scada network. *Computers & Security*, 70:16–26, 2017.

[55] Ahsan Al Zaki Khan and Gursel Serpen. Misuse intrusion detection using machine learning for gas pipeline scada networks, 05 2019.

[56] Konstantinos Demertzis, Lazaros Iliadis, and Stefanos Spartalis. A spiking one-class anomaly detection framework for cyber-security on industrial control systems. pages 122–134, 08 2017.

[57] Sara Tamy, Hicham Belhadaoui, Mahmoud Almostafa Rabbah, Nabila Rabbah, and Mounir Rifi. An evaluation of machine learning algorithms to detect attacks in scada network. In *2019 7th Mediterranean Congress of Telecommunications (CMT)*, pages 1–5, 2019.

[58] Rocio Lopez Perez, Florian Adamsky, Ridha Soua, and Thomas Engel. Machine learning for reliable network attack detection in scada systems. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 633–638, 2018.

[59] Andrés Felipe Sánchez Prisco and M. John Freddy Duitama. Intrusion detection system for scada platforms through machine learning algorithms. In *2017 IEEE Colombian Conference on Communications and Computing (COLCOM)*, pages 1–6, 2017.

[60] Majed Al-Asiri and El-Sayed M. El-Alfy. On using physical based intrusion detection in scada systems. *Procedia Computer Science*, 170:34–42, 2020. The 11th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 3rd International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops.

[61] Krishna Madhuri Paramkusem and Ramazan Aygun. Classifying categories of scada attacks in a big data framework. *Annals of Data Science*, 5:28, 09 2018.

[62] *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier BV, Jan 2011.

[63] XGBoost Documentation &x2014; xgboost 2.1.1 documentation — xgboost.readthedocs.io. https://xgboost.readthedocs.io/en/stable/, 2022. [Accessed 01-08-2024].