Mälardalen University
School of Innovation Design and Engineering
Västerås, Sweden

Thesis for the Degree of Master of Science (60 credits) in Computer Science with Specialization in Software Engineering 30.0 hp DVA424

# INTRUSION DETECTION USING MACHINE LEARNING FOR INDUSTRIAL CONTROL SYSTEMS

Roland Plaka
rpa20001@student.mdh.se

Examiner: Radu Dobrin
Mälardalen University, Västerås, Sweden

Supervisors: Sasikumar Punnekkat
Mälardalen University, Västerås, Sweden

Company supervisor: Ali Balador
RISE - Research Institutes of Sweden, Västerås,Sweden

June 11, 2021

## Abstract

*An intrusion detection system (IDS) is a software application that monitors a network for unauthorized and malicious activities or security policy violations related to confidentiality, integrity, and availability of a system. In this thesis, we performed detailed literature reviews on the different types of IDS, anomaly detection methods, and machine learning algorithms that can be used for detection and classification. We propose a hybrid intrusion detection software architecture for IDS using machine learning algorithms. By placing appropriate machine learning algorithms in the existing detection systems, improvements in attack detection and classification can be obtained. We have also attempted to compare the machine learning algorithms by testing them in a simulated environment to make performance evaluations. Our approach provides indicators in selecting machine learning algorithms that can be used for a generic intrusion detection system in the context of industrial control applications.*

***Keywords: Intrusion detection, machine learning, security***

## Acknowledgements

Many people had supporting roles during this unique journey full of memorable moments. Foremost, I would like to express my sincere gratitude to the great team of my supervisors. I want to thank my university supervisor Professor Sasikumar Punnekat for his kind support, excellence, patience, and encouragement. I deeply appreciate and thank the collaboration with Dr. Ali Balador for his thoughtful advice, caring, and continuous energizing support as a company supervisor. Last but not least, I would like to thank Dr. Jan Carlson and Prof. Assoc. Radu Dobrin very much for their helpful comments and consultancy to formulate the thesis scope at the beginning.

I have been fortunate to be part of such an inspiring leading research organization as RISE Research Institutes of Sweden and work with scientists and knowledgeable colleagues.

Special thanks also to my manager Larisa Rizvanovic, and all my great researcher colleagues, who gave me valuable hints to do professional work. I want to express my profound appreciation to my home university, the European University of Tirana that made it possible to attend my studies at Mälardalen university via the Erasmus+ program. My study experience at Mälardalen university has allowed me to grow the academic background, meet new friends, and work with great people.

Above all, I thank God for helping me in my entire life, and then I would like to express my deepest gratitude to my family and my close friends for all their supportive presence, understanding, and being patient with me. Without their support, I would not have reached here.

**Roland Plaka, Västerås, May 2021**

**The man who moves a mountain begins by carrying away small stones.**
**Confucius**

# Table of Contents

# 1 Introduction

Industrial control systems(ICS) are widely used in managing water and sewage, communications, electricity supply, transportation, public safety. It is necessary to protect these manufacturing facilities from cyber attacks. Security is the primary concern for any industrial control system network due to the generation of many attacks every day. Software is a critical component in industrial control systems. A large part of the effort is spent, and well-known software engineering practices are used to ensure that the software function as intended.

Software engineering is a necessary discipline that provides control over software functionalities, quality, and resources by contributing to software development. Software engineering concerns all processes of software production. These processes form the so-called system development life cycle or SDLC. System development life cycle consists of: :

1. Planning and requirement
2. Design
3. Coding
4. Testing
5. Implementation
6. Maintenance

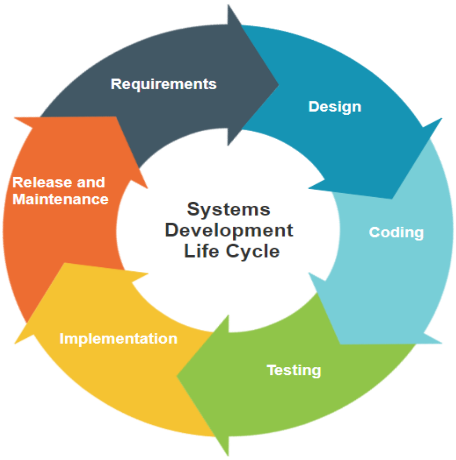


Figure 1.1: System development life cycle

**Why is intrusion detection important in software development life cycle?**
Complex software systems face security risks that may lead to system failures during their operational phase. For such reasons, it is vital to adopt a set of practices that support security assurance. Traditional security-related activities are performed only during the testing phase. In general, developers do not focus on the security vulnerabilities but give importance to developing the system's functionality. As a result, many bugs, flaws, and similar vulnerabilities or errors in a developed system may exist. A secure SDLC would help developers maintain software security by reducing the number and severity of vulnerabilities while reducing maintenance costs. Intrusion detection activities are performed heavily to meet the challenge of developing and maintaining large, complex software systems in a changing environment. Intrusion detection would make developers available to identify security vulnerabilities resulting from coding errors, system configuration faults, or other operational weaknesses from outside and inside, determine risk and identify mitigation strategies. Intrusion detection would help to define security requirements which are helpful to reflect changes in system functionality. Besides, intrusion detection would help identify the minimum security quality levels and how engineering teams will be held accountable.

## 1.1 Problem Formulation

Providing security to the industrial networks using IT solutions may not be a reasonable approach because of the different functionalities that these networks have. Hence, to effectively protect the ICS network from the increasing number of intrusions and reduce their impact, an efficient Intrusion Detection Systems(IDS) which can minimize the effects of the attacks is vital. However, existing IDSs have shown inefficiency in detecting zero-day attacks. They also suffer from false positives (unnecessary alarm) and false negatives (which impact the security), which affect the performance and accuracy of the ICS. When designing an efficient IDS framework, the problem that struggles developers is to intertwine various components to reduce these drawbacks.

## 1.2 Main objectives and contributions

This master thesis aims to do such an investigation following the case study methodology by doing empirical research on intrusion detection systems from the software engineering perspective, focusing on the necessity of the integration mentioned above and its benefits. The main objectives of this work are as follows:

- To identify the important security requirements that an IDS should fulfill.
- To investigate different Intrusion Detection Systems (IDS).
- To Investigate AI and machine learning models and algorithms that can be run for intrusion detection in industrial control systems.
- To design a hybrid anomaly detection software architecture using machine learning, which can detect and classify the anomalies by analyzing the saved logs from an industrial control system.

The main contributions of this thesis are as below:

- We conducted a study of different intrusion detection systems and machine learning algorithms to understand the state-of-the-art. As a result, we provided a proper categorization of the intrusion detection systems according to their installment and detection method. As well, we did a related categorization for the machine learning algorithms. We selected the ones that performed better in anomaly detection and classification based on the performance metrics values of accuracy, detection rate, true positive rate, false positive rate, precision and recall.
- We evaluate the performance of the selected algorithms testing them on a data set that contains several anomalies.
- We propose an alternative improvement on the generic IDS architecture by designing a hybrid software framework for an intrusion detection system using machine learning. Our experimental tests will demonstrate the positive impact that machine learning provides on finding and categorizing attacks.
- We propose our novel architecture by combining the human factor intuition with the machine learning concept to build a secure artificially intelligent framework. Section eight describes the experimental process and the setup.

In sections two, three, and four, we overview the main terms necessary to understand the thesis, such as ICS,IDS, and machine learning. In section five, we show the steps for our research methodology. Section six shows the literature review on machine learning in IDS.
In section seven, we describe our detailed research questions (based on the objectives specified above) and provide answers to these research questions. We explained the result in section nine and made a validity evaluation. Section ten focuses on the ethical principles that we followed during the research process. Lastly, we give the conclusions and discussion for future work.

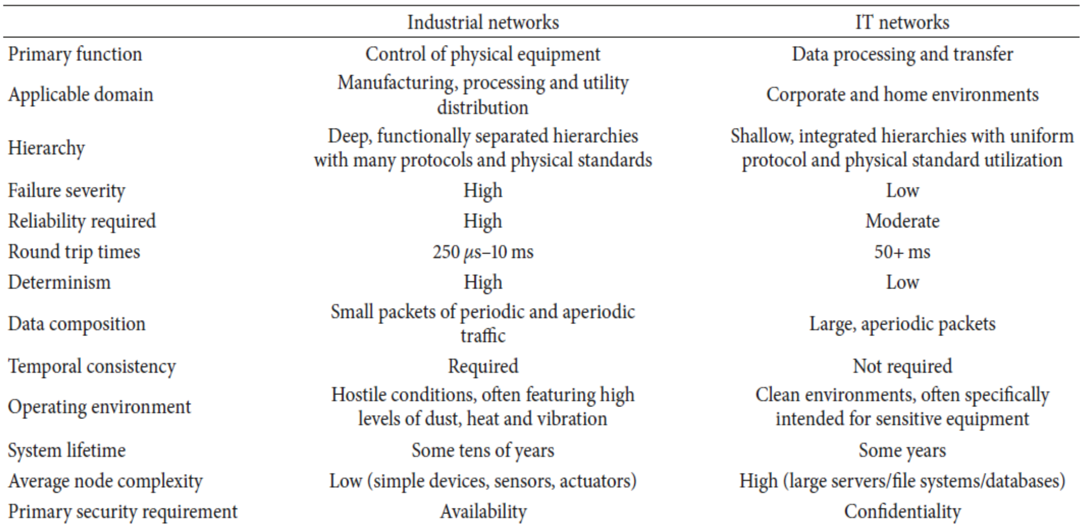
# 2 Background

The master thesis is related to cybersecurity for industrial control systems from a software engineering perspective. Our work is part of the EU project called INSECTT, including 52 partners from 12 different countries. The main goal in the Intelligent Secure Trustable Things (InSecTT) project is to provide intelligent, secure, and trustworthy systems for industrial applications, in particular for the manufacturing industry. Both RISE and Westermo are involved in and are interested to this EU project. The researchers have explored using ML techniques to fulfill the security requirements of an effective IDS. Machine learning is powerful in learning valuable features from the ongoing network traffic and can help predict normal and abnormal activities based on the learned models. We will focus on detection accuracy and performance when using machine learning(ML) detection methods in IDS. Evaluation of detection accuracy and performance are essential aspects of the intrusion detection system development's life cycle. The primary goal of the detection is to detect an ongoing attack in the shortest time reliably. The zero-day attacks are a game-changer and are harder to be noticed. For such reason, using ML algorithms is to see and classify intrusions by providing more accurate predictions. As a result, the false alarms will be minimized, and better data analysis will be given.

## 2.1 Industrial Control Systems

IDS is a broad term covering everything to control, monitoring, and production industries [19], and encompasses specific systems, such as supervisory control and data acquisition (SCADA), distributed control systems (DCS), and programmable logic controllers (PLC) [40]. Industrial control systems are interconnected to critical infrastructures. The system's security may be at risk at any time. New vulnerabilities from external sources, such as terrorists, hackers, industrial espionage [40], and internal sources such as social engineering and equipment failure, threaten industry security. During the last decade, several attacks happened. Some known cases are; the STUXNET worm attacks on Iran's nuclear installation [31]in 2010, Black Energy malware attack on the Ukrainian power grid in 2015 [3], and the German steel factory attack in 2014. Authors at [43], highlight the risk of attacks toward these systems. As a result of these attacks' complex nature and origin, the authors emphasize the need for developed security mechanisms. We cannot respond to these attacks just by providing traditional IT measures, such as firewalls, but it is necessary to have an intrusion detection system as well.

The increasing number of devices spurred the need for the expansion of industrial networks. The merge of industrial networks with IT networks was made to standardize several communication protocols (TCP/IP) necessary to exchange information between devices. As a result, both the complexity and the number of security issues are raised. Industrial networks already face similar attacks as those against IT networks. Still, applying similar methods to operational technology(OT) for mitigating these attacks does not have the same effect. This is due to the entirely different function of the components, which are part of embedded systems and face unique characteristics. It is necessary to distinguish the main issues that characterize each of these two networks. The study of Iturbe et al. [22] presents the main differences between IT networks and industrial networks. To have a better idea, it is worth highlighting the specific criteria for the two types of networks in the given table.



| | Industrial networks | IT networks |
|---|---|---|
| Primary function | Control of physical equipment | Data processing and transfer |
| Applicable domain | Manufacturing, processing and utility distribution | Corporate and home environments |
| Hierarchy | Deep, functionally separated hierarchies with many protocols and physical standards | Shallow, integrated hierarchies with uniform protocol and physical standard utilization |
| Failure severity | High | Low |
| Reliability required | High | Moderate |
| Round trip times | 250 $\mu$s–10 ms | 50+ ms |
| Determinism | High | Low |
| Data composition | Small packets of periodic and aperiodic traffic | Large, aperiodic packets |
| Temporal consistency | Required | Not required |
| Operating environment | Hostile conditions, often featuring high levels of dust, heat and vibration | Clean environments, often specifically intended for sensitive equipment |
| System lifetime | Some tens of years | Some years |
| Average node complexity | Low (simple devices, sensors, actuators) | High (large servers/file systems/databases) |
| Primary security requirement | Availability | Confidentiality |

Figure 2.1: Industrial Networks compared to IT Networks(adapted from [22])

Security requirements of industrial networks differ from IT networks. The essential security requirements for the information networks are confidentiality, integrity, and availability.
**Confidentiality**: It is the prevention of sensitive data from the disclosure of unauthorized persons or systems. The development of industrial control systems brings the need for integration with the global Internet of Things. The ICS assets interact more with the user; for such reason, private data confidentiality becomes a critical objective but not a primary one. It is the primary security requirement for IT networks. A threat example could be the information theft by injecting codes using SQLInjection, to corrupt the security credentials.
**Integrity**: Addresses the unauthorized modification of data by unauthorized persons or systems, so the violation of it may lead to safety issues. Industrial networks must assure data integrity using mechanisms and protocols which can detect manipulation by prohibited users. For instance, impersonation or spoofing is a case of a Man-in-the-Middle attack.
**Availability**: It refers to ensuring that unauthorized persons or systems cannot deny access or use to authorized users. Availability violation in critical infrastructures is known as Denial of Service (DoS). It may cause economic damage and safety issues since operators may lose the ability to monitor and control the process. It is the primary security requirement for industrial networks.
External and internal attacks allow attackers to penetrate a system, access a control center, and modify load conditions to destabilize a critical infrastructure in unpredictable ways. These actions lead to severe results or disasters; even catastrophic blackout [27]. For such reason, it is vital to detect the intrusions as quickly as possible to minimize the risk. An IDS helps to detect such intrusions. IDSs are significant network security components that examine system or network activity. They find possible intrusions or attacks that trigger security alerts when they happen.
The typical infrastructure of an industrial network setup, including control network, central office, firewall, both wired and wireless communication technologies, is presented as follows:

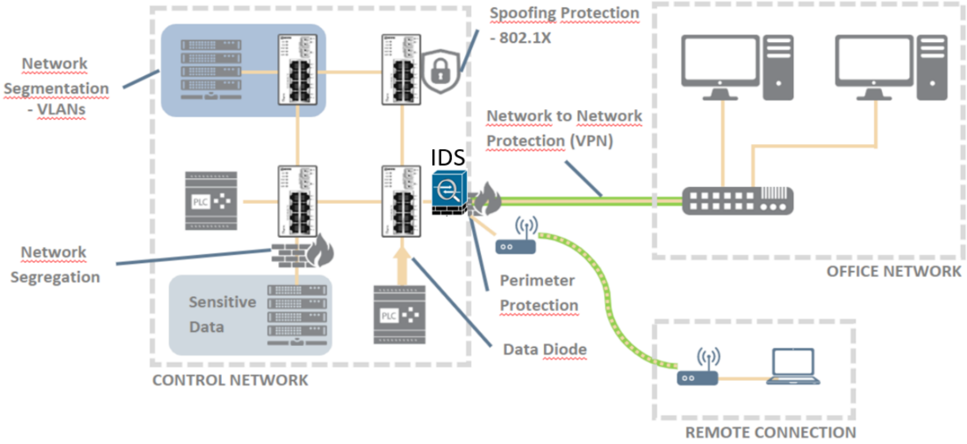


Figure 2.2: Industrial Networks Setup

Figure 3.3 summarizes the structure of a simple industrial network. At the bottom level, also called the physical layer, the sensors transmit their data to the field controllers. Asghar et al. [6] define Programmable Logic Controllers (PLCs) as a logic interface between the SCADA systems and sensors. PLCs work with the supervisory system by reading the control commands or returning the status of sensors based on their control algorithms. Direct connections through PLCs form the field network in the industrial networks. These devices commonly run on IT-based hardware and software. Field controllers serve as a connecting bridge between the two layers, exchanging data between the field and control networks. Later the data is transmitted to human-machine interfaces(HMI), engineering stations, and control servers.

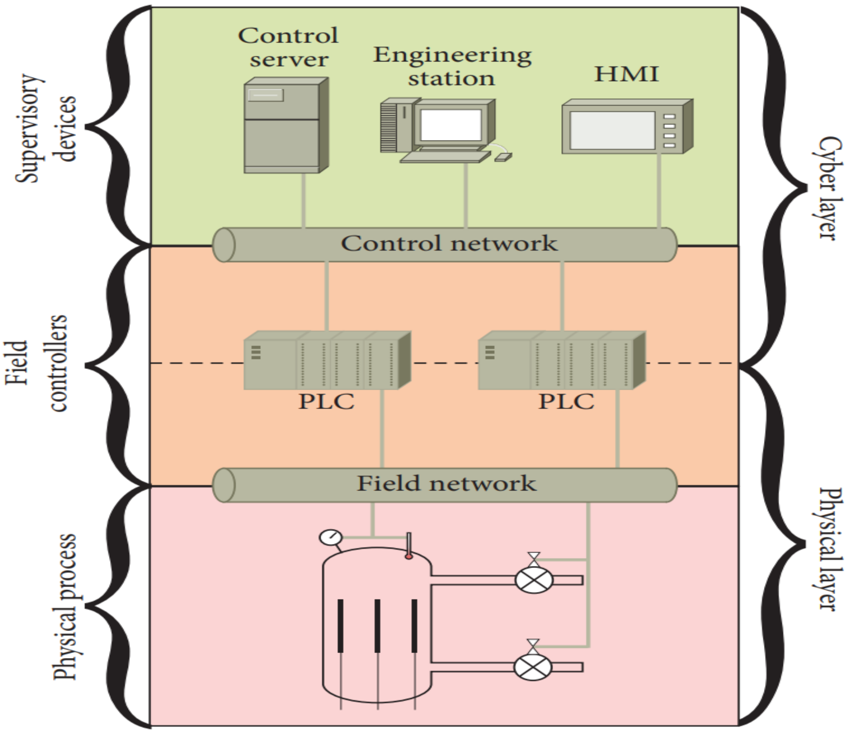


Figure 2.3: Simple industrial network

# 3 Intrusion Detection Systems-An overview

Intrusion detection systems observe activities in the network traffic or analyze data generated by a host for any intrusion attempt that violates the security. These systems are the first defense solution against the cybersecurity threats that can occur to the industry. They raise the alarm when an attack is detected. As cited at [28] "Monitor-Detect-Respond to any unauthorized activity are the adages of IDS." Based on their implementation, there are two types of IDS categories, Host-based IDS and Network-based IDS. Intrusion is an unauthorized use, misuse, and abuse of computer systems by both inside and outside intruders. The IDS's main task is to defend a computer system or computer network by detecting hostile attacks on a network system or host devices.

This paragraph will take a closer look at the intrusion detection system types and detection methods. There are two types of IDS based on their installed placement, host-based IDS and network-based IDS.

## 3.1 Host-based IDS

A host-based intrusion detection system (HIDS) generally operates within a computer, node, or device. Its primary function is internal monitoring, although many variants of HIDS have been developed that can be used to monitor network [47],[40]. A HIDS determines if a system has been compromised by inspecting the full communication stream and warning administrators accordingly, i.e., it can detect a rogue program that suspiciously accesses a system's resources or discovers that a program has modified the registry in a harmful way [40].

## 3.2 Network-based IDS

A network-based intrusion detection system (NIDS) functionality is to monitor and analyze the network traffic. Its prominent role is to protect the system from network-based threats by discovering unauthorized malicious access to a LAN and exploring traffic that traverses the wire, multiple hosts. Detection algorithms read inbound and outgoing packets and searches for any suspicious patterns, so an alert generated by NIDS notifies the administrator [40]. This technology has three different network topologies like directly connecting it to a switch spanning port [12], using a network tap, and connected inline. IDS technologies deliver both traditional IT security measures and specific measures to the unique features of ICS.
We considered it essential to have a clear comparison of these systems, summarizing some advantages and disadvantages of both types in table 3.1

| IDS Type | Advantages | Disadvantages |
|---|---|---|
| Host-based IDS | Looks at the entire OS and is able to inspect the full communication stream.<br>Can detect insider attacks that do not involve network traffic and can check end-to-end encrypted communications.<br>It does not requires extra hardware to be installed.<br>Detects intrusions by checking system calls, system directories, application logs and user activities. | More management efforts when installing configuration for every host.<br>Can be disabled when certain types of denial of service attacks occur and as a result can lead to functionality loss in HIDS.<br>Consumes host resources since OS audit logs occupy a large amount of space.<br>Monitors only local attacks in the machine that is installed.<br>Can happen delays in reporting attacks. |
| Network-based IDS | Detects attacks by scanning network traffic.<br>Monitors multiple hosts on the network to identify any suspicious activity.<br>Not suspectible to direct attacks and capable to not be detected by attackers.<br>Checks the broadest ranges of network protocols such as (TCP/UDP/ICMP/SNMP) and router Netflow records. | Challenge to recognize attacks from high speed encrypted traffic when network volume is overwhelming.<br>Some networks are not able to provide all the data analysis because switches have limited monitoring port capability.<br>Specific hardware is needed.<br>Identification only of network attacks. |

Table 3.1

We will investigate a network-based IDS as a case study and will equip it with ML methods to detect the anomalies happening in the network.

The architecture of a generic NIDS contains these components:

- **Data gathering sensors**
  They are used to monitor the infrastructure where the data gathering is performed and observe specific processes or protocols. They perform a primary classification of the data received from their location.
- **Detector engine**
  This module performs a comparison between the gathered data and the defined rules set. When IDS finds a deviation of the normal status, it raises the alarm.
- **Storage Module**
  It contains the rule sets of the IDS, which the detector uses when comparing the received data.
- **Response**
  When an alarm raises, it responds with a precisely defined action. Depending on the type of the alarm, sometimes it could be an action where the IDS can perform a predefined action, such as dropping down the malicious packets. Sometimes it can be a passive response, such as logging the activity and let the human factor decide for the action.

A typical IDS architecture is illustrated below:

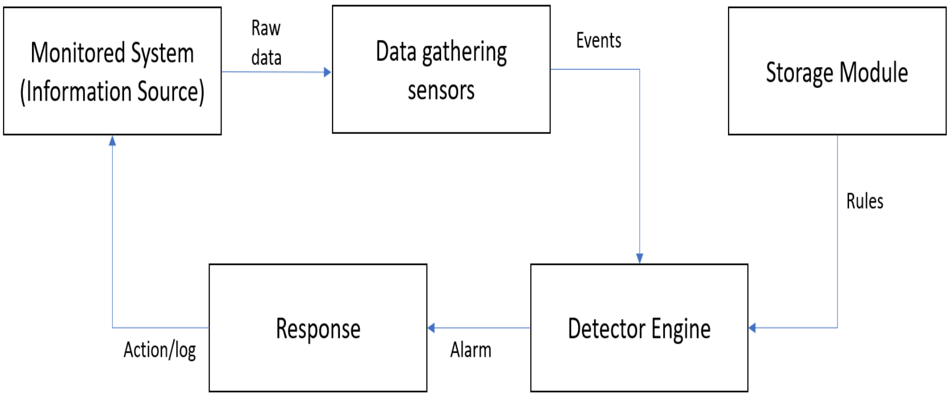


Figure 3.1: Intrusion Detection System architecture (adapted from [45])

Developers are looking for real-time network monitoring solutions since they want the reactions to be as soon as possible on time. Intrusions or attacks monitored by both NIDS and HIDS are detected in three manners:

- Signature-based detection
- Behavior-based detection
- Specification-based detection or also called the hybrid detection

## 3.3 Signature-based detection

Misuse detection or signature-based detection relies on an extensive database that contains previous well-known vulnerabilities registered in it. The system looks for a unique pattern called signature and matches the existing ones in the database in the occurring attacks. Pranggono et al. [40], in their book, provide a better understanding for signature-based IDSs, saying that every signature corresponds to an entry in the database. An example of this is Cisco systems large attack databases used in industrial environments. The primary benefit is high accuracy in detecting well-known attacks and the simple implementation. Their focus is on communication protocols like Modbus and DNP3. Some open-source or commercial tools like SNORT or BRO may increase the detection performance. However, they are still less effective in catching zero-day attacks. Shortly the adequacy of these systems depends on the upgradeability of signature databases and rulesets done by vendors a couple of times per year. This technology is not so much preferred to provide cybersecurity for an ICS. However, it is still implemented in water treatment systems, electrical power grids because they integrate with more sophisticated and modern models..

### 3.4 Behavior-based detection

Kaouk et al. [25] mention that behavior-based detection models the expected behaviors of the system and network. It alerts the admin whenever the behavior deviates from the predefined threshold. Unlike signature-based detection, these systems can detect zero-day attacks since no rules are needed to be written. This detection technology makes it harder for attackers to learn the IDSs capabilities. However, they have a high false alarm rate and difficulties detecting the attack type. Several researchers have combined several techniques and assets to improve these drawbacks. However, when used alone, they still show a low detection rate. Another disadvantage is difficulties in defining the ruleset.

### 3.5 Specification-based detection

Machine learning methods make it possible to combine both behavior-based and signature-based detection to form a new type of detection called hybrid detection or specification-based detection. Soon, this type of detection may dominate the market of Industrial Control Systems. This combination reduces the detection gaps mentioned earlier by reducing both false negative and false positive rate alerts. Surveys have shown that using machine learning algorithms effectively impacts cyber-attack detection and is seen as a promising approach to enhance cybersecurity. The following picture illustrates intrusion detection systems categorization based on their installment and detection technique.

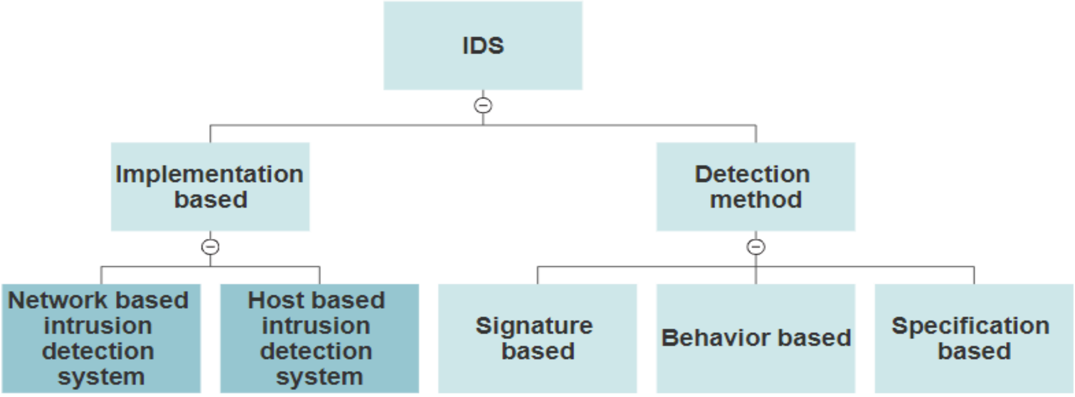


Figure 3.2: IDS classification

## 4 Machine learning algorithms

Machine learning algorithms can improve network security by making necessary calculations and decisions such as recognizing the type of packets in the traffic. Machine learning algorithms are categorized as follows:

## 4.1 Supervised algorithms

Supervised algorithms need fully labeled class data. Network intrusion detection based on supervised machine learning algorithms requires a dataset divided into two parts: training data and testing data. The main objective is to train algorithms with labeled data to create a model. The next step is using this learned model to predict the 'unknown' in test data. In their survey of IDS [26], authors explain the training process, saying that supervised algorithms identify the relevant features and classes. Inputs are records from network or host sources combined with an output value specified as a label. In general, label categories are called attack(intrusion) or normal data. Feature selection is applied to get rid of unneeded features. The algorithms are tested after the training data learns the relationship between the input data and labels utilizing either classification or regression. In the testing stage, algorithms predict the anomalies and the relevant class from the new incoming data. Rosa et al. [44] additionally advocate that supervised ML models are typically tuned for a solitary situation, either for a particular process either for a single communication protocol. There is a wide assortment of supervised machine learning algorithms. However, this thesis presents the most commonly used under the classification category. These algorithms are Support Vector Machine(SVM), K-Nearest Neighbor(KNN), Artificial Neural Networks(ANN), Naïve Bayes, Decision Tree(DT), and Random Forest(RF). Supervised learning is better for the detection of known attacks, yet not for new attacks.

## 4.2 Unsupervised algorithms

Unsupervised learning use clustering algorithms to generate a model from unlabeled data. In this way, they can distinguish malicious inputs from network traffic or host logs. According to their statistical properties, unsupervised methods analyze the data characteristics randomly, without any prior knowledge. Nisioti et al. [36] present arguments to emphasize that unsupervised methods do not require the time-consuming data training stage. A significant property of these algorithms is feature selection. Clustering techniques aim to separate the input data into clusters that achieve the highest relevance and the lowest redundancy by inspecting the data structure. During the analysis, they define a threshold. Nisioti et al. further assert that the number of regular instances in a dataset is more significant than the number of anomalies. After the clusters are created based on their relations, each of them is assigned a unique score. When the score of a cluster exceeds the threshold, then it is treated as malicious. Consequently, regular traffic will group into large clusters, and attacks will be in smaller clusters. This process is much more efficient compared to the training stage of supervised learning. Moreover, unsupervised learning methods can detect unknown attacks, called zero-day attacks, with no need for labeled data. However, their disadvantage is that they show a high false-positive rate. Zaman et al. [54] mention the most commonly used algorithms: K-mean clustering, Fuzzy clustering, and Hierarchical clustering.

## 4.3 Hybrid algorithms

Erbad et al. [16] say that actual solutions used for intrusion detection systems deal with different issues such as detection in real-time, low detection accuracy, and irregular detection rates. Thus, the need arose to develop hybrid machine learning algorithms for dealing with these problems. The main objectives of hybrid machine learning algorithms are reducing false-negative and false-positive alarms. Hybrid methods used for intrusion detection combine supervised and unsupervised machine learning approaches to improve the system´s performance. Buczak et al. [11] at his work describe the method of combining the accuracy of supervised algorithms for known attacks with the detection ability to catch unknown attacks of unsupervised algorithms. Since the attacks are changing their operational behavior, it is essential to think beyond classic IT-based solutions. Components in hybrid classifiers perform different tasks to maximize intrusion detection in networks, i.e., pre-processing, classification, clustering based on the combination. Even there is a great need for exploration in this field, some researchers have already published their works. Anton et al. [4] introduced a hybrid match of SVM and Bayes algorithms. SVM algorithm was used to distinguish data into normal and anomalies. A decision tree was used to spot the known types of anomalies, and a Naive Bayes algorithm to detect the unknown anomalies. This technique showed an 80.68 % accuracy and 1.57% detection rate, overcoming the limitations of each algorithm and improving the general performance of the intrusion detection system. Pranggono et al.[40] introduce some applications of hybrid algorithms. Their work presents a network intrusion detection model proposed by Fovino et al., which relates both unsupervised and supervised methods to detect complex attacks to SCADA systems.

# 5 Research Methodology

In this chapter, we introduce the methodology that we use for our work. We will use the case study methodology to investigate in-depth intrusion detection and the machine learning field. This empirical research leads us to explore some possible integration between these two fields. Another reason for selecting this methodology is because we are handling both qualitative and quantitative data.
The methodology that we followed is presented in these stages:

## 5.1 Literature Review

This stage involves literature study for a deep understanding of the state-of-the-art of Intrusion Detection Systems software (IDS) and investigation over machine learning solutions. Essentially reading papers for understanding the related work, reading articles over the differences that may threaten the industrial control systems, and understanding the different ML-based approaches for designing an IDS.

## 5.2 Design

We elaborate on the concept of designing a machine learning framework with intrusion detection capabilities. We integrate multiple components that could help to improve the performance of IDS on the basis of accuracy, detection rate, precision, false negative alarms and true positive alarms.

## 5.3 Implementation

In this phase, we apply different ML algorithms for intrusion detection and classification on an open-source dataset called BATADAL. The main motivation for choosing this dataset is that it fits our thesis scope since modern water distribution systems is an excellent example of an ICS and rely on computers, sensors, and actuators for both monitoring and operational purposes. Moreover, we decided to use BATADAL for the reasons listed below:

- Has a sufficient number of data to perform training and testing for our proposed framework.
- It is a dataset of recent years.
- Has a more accurate and detailed description of the attacks/anomalies it contains.

Unable to use a simulator to build attacks, we decided to use the security logs of the BATADAL water distribution system dataset that were collected after a passive listening of the traffic in the network. This dataset contains a Denial of Service attack.

## 5.4 Experimentation and evaluation

The focus will be on experimentation and evaluations on the effectiveness of our proposed algorithms in Weka machine learning software. Weka is a workbench, which provides machine learning algorithms for regression, classification, clustering, association rule, and attribute selection. Besides, it posses tools for data preprocessing which allowed us to compare different machine learning methods on the dataset we selected.

# 6 Literature review on machine learning algorithms in IDS

In this section are introduced papers that discuss and provide background for the problem mentioned above. At the same time, we look for inspirational examples that could help us design a hybrid intrusion detection system framework. We will study the literature surveys and previous works to propose the framework we are searching for. We reviewed different approaches during our research process and identified the most relevant articles that we thought could provide us a method to solve our problem.
The work by Krotofil and Gollmann [19],[29] focuses on different types of attacks on existing systems but also concludes that most efforts so far have been devoted to intrusion detection system (IDS). A comprehensive systematic overview of security in cyber-physical systems is given in by Humayed et al. In [19],[21] authors identify that significant security challenges in ICS which are change management as well as the ability to handle legacy systems.
Duque et al.[15] proposed a high-efficiency rate, low false positives, and false negatives model for an Intrusion Detection System, using the unsupervised machine learning algorithm called k-means. The algorithm was applied to the NSL-KDD dataset using several clusters. The best results were generated when the number of clusters matches the data types in the dataset.
Gaikwad et al.[18] in their paper, make use of the Bagging Ensemble method to implement an IDS. They used Partial Decision Tree as a base classifier due to its simplicity. Next, they used a Genetic algorithm for feature selection. The proposed intrusion detection system is evaluated in terms of classification accuracy, true positives, false-positive, and model building time. The outcomes revealed that the introduced system scored the highest classification accuracy of 99.71% using cross-validation. This model exhibits higher classification accuracy than all classifiers except the C4.5 classifier on the test dataset.
In [32] the authors proposed a novel feature representation approach, called the cluster center and nearest neighbor(CANN) approach. This methodology estimated and added two distances, the first dependent on the distance between each data sample and its cluster center. The subsequent distance is between the data and its nearest neighbor in the same cluster. This new and one-dimensional distance is utilized to address each data sample for intrusion detection by a K-NN classifier. The outcomes show that the CANN classifier performs better than k-NN and SVM in classification accuracy, detection rates, and false alarms. It provides high computational efficiency for the time of classifier training and testing.
Parvat et al.[5] illustrate a multi-layer hybrid algorithm for intrusion detection combining the Naïve Bayes and C4.5 algorithm. Initially, a misuse detection model is constructed dependent on the C4.5 algorithm. Afterward, the training data is parsed into smaller subsets using this model. Then, multiple one-class Naïve Bayes algorithm models are made for the parsed subsets. The hybrid classifier is used as a preprocessor to shorten the training time and improve the IDS's performance. After evaluating various machine learning algorithms such as Naïve Bayes, C4.5 or J48, PART, and Random Tree algorithms, the results showed that the Decision Tree, C4.5 and Naïve Bayes give the higher accuracy and low time complexity.
Authors at [35] present a hybrid approach to minimize the malicious traffic in the network. This combination of Artificial Immune Network with Radial Basis Function (AIN+RBF) with ANN performs the best accuracy compared to other approaches.
Farnaaz et al.[17] built an intrusion detection system model using an RF ensemble classifier. RF performs well compared to the J48 algorithm in terms of accuracy, detection rate, and false alarm rate. The RF detected four types of attack, such as DoS, Probe, U2R, and R2L. It proved it is efficient with a low false alarm rate and high detection rate.
In [33] authors compare machine learning algorithms SVM, Naïve Bayes, J.48, and Decision table for anomaly detection. The results show that each algorithm has a different performance. No single algorithm has a high true positive rate, but J.48 decision tree has high accuracy and low misclassification rate.
Aburomman et al.[2] implemented an ensemble algorithm merging Linear Discriminant Analysis(LDA), Principle Component Analysis(PCA) with SVM. This method produced better accuracy in comparison to a single feature extraction method.
A hybrid model consisted of SVM and Genetic Algorithm is presented by Atefi et al.[8]. The results showed that when the hybrid model was used, the precision-fitted in around 98% and performed

higher accuracy of detecting intrusions. Related to the True Positive rate, the hybrid model showed that the predicted attacks were much more than 99%, and the False positive rate was with a rate of 1.78.

Belavagi et al.[10] in his paper, assembled an exact model using Logistic Regression, Gaussian Naive Bayes, RF, and SVM. The outcomes show that the RF algorithm outperforms the other algorithms on classifying whether the data is normal or an attack, based on precision, recall, and accuracy. It has 99% accuracy.

In [48] authors propose using the Average One Dependence Estimators(AODE) algorithm to detect several types of attacks in a network intrusion detection system. The proposed model seems efficient with a low false alarm rate and detection rate compared with the Naïve Bayes algorithm.

According to Ashfaq et al.[7] their proposed fuzziness-based semi-supervised learning approach assisted with supervised learning algorithm improves the classifier´s performance for the IDSs. Their work is designed with a single hidden layer feed-forward neural network(FFNN). They have trained to output a fuzzy membership vector, using the neural network with random weights as a base classifier. Next, the fuzzy quantity was used on the unlabeled samples. Then the classifier is retrained after incorporating each category separately into the original training set. The results show that unlabeled samples belonging to low and high fuzziness groups make significant contributions to improve the classifier´s performance compared to Naïve Bayes, SVM, and RF. Besides this, this study proves that samples belonging to the mid fuzziness group have a higher risk of misclassification for IDSs.

Authors at [37] show a study of the performance of hybrid machine learning techniques such as ANN and SVM with radial kernel algorithms for supervised learning and K-means for unsupervised learning. Also, they use Principal Component Analysis(PCA) and Gradual Feature Reduction(GFR) as feature selection techniques. The outcomes reveal that the composite model has the strength of supervised learning to detect the known attacks and the power of unsupervised learning to detect the unknown attacks. ANN and K-Means are an effective combination for intrusion detection. The most reliable model for intrusion detection is ANN and Gradual Feature Reduction(GFR) hybrid model since it performs the best.

Karamudin et al.[24] introduce a hybrid intrusion detection system employing Logitboost with RF algorithm. The results show that their methodology exhibits predominance regarding accuracy and detection rate over the conventional approaches while saving low false rejection rates. Among the other algorithms, this ensemble approach has successfully detected unknown attacks.

Jabbar et al.[23] in their paper deal with a novel hybrid algorithm called RF and Average One-Dependence Estimator(RFAODE). They use RF to improve accuracy and to reduce the error rate. They tested this merge on the Kyoto dataset. The algorithm achieved 90.51% accuracy and 0.14% false alarm rate, obviously outperforming AODE, Naive Bayes, and RF algorithm. It classified network traffic efficiently as normal or malicious.

In [1] a comparison between Naive Bayes, Decision Trees-Naive Bayes Tree, Best First Tree, J48, RF, and Multi-Layer Perceptron neural network(MLPNN) was presented to give more information about known and predict the class of unknown attacks. The study realized that multiple algorithms should be involved to increase accuracy.

Swathi et al.[42] observe that their Indexed Partial Distance Search k-Nearest Neighbor(IKPDS) and Partial Distance Search K-nearest Neighbor(PDSK) classification algorithms compared to traddítional kNN gives 99.6% accuracy. The outcomes determined that IKPDS gave a better classification of attacks within a short time sequence.

Authors [9] in their work suggested an intrusion detection system based on the fuzzy min-max neural network(FMNN) and the particle swarm optimization(PSO). Performance metrics taken into consideration to test the effectiveness of the system were classification accuracy and classification error. This design required less time to train the data and delivered more excellent performance compared to other classifiers.

Authors[34] enhanced the existed machine learning algorithms in suggesting a better solution to detect a different kind of malicious traffic. They recommended to use three methods $(SVM^2, k-means^2, SVM+k)$. From the different performance metrics discussed, such as accuracy, detection rate, false alarm rate, the authors conclude that their approach has better performance metric results than the traditional existing algorithms. These techniques influence the multi-level tweak to help intrusion detection.

Chowdhury et al. [13] proposed a combination of simulated annealing and SVM for intrusion detection to improve the accuracy and lower the false positive rate.
Machine learning algorithms introduced in [4] SVM and RF are fit for boosting the detection capacities of IDSs. The given algorithms can detect 90-95% of the attacks in the data. In any case, RF somewhat outperforms SVM since it gives intends to ascertain the meaning of individual features with satisfactory results.
Hasan et al.[20] in their paper, use Logistic Regression (LR), SVM, DT, RF, and ANN machine learning algorithms. The utilized assessment metrics are accuracy, precision, recall, f-1 score, and Receiver Operating Characteristic Curve(ROCC). The authors are more focused on classifying multiple classes. With the used dataset, the RF technique is the appropriate one for solving cyberattacks on Internet of Things networks because of its precise forecast.
Saranya et al.[46] carry out LDA, Classification, Regression Tree(CART), and RF in the classification phase. A metric like an accuracy and kappa was used to measure the algorithms' work in the evaluation phase. Therefore, RF shows the highest rate of accuracy with 99.65%. The author's reason that algorithms such as RF, ANN, and decision trees yield better-classifying attacks.
Rajagopal et al.[41] introduce a combination made up with RF, LR, K-NN, and SVM. The authors used a real-time dataset. They consolidated that the proposed approach fills in as a practical viewpoint for real-time network intrusion detection since the model predicted accurately. Their work shows that the recommended machine learning algorithms SVM, decision trees, K-NN, and k-means clustering perform well in detecting anomalous traffic in SCADA systems [39]. All these methods that we presented demonstrated their positive impact in improving detection performance. Most of these were tested using open-source datasets such as KDD99, and the results may be different when applied in different environments. As a conclusion from this literature review, we learned that the accuracy is increased when merging more than two ML algorithms for identifying the anomalies in the network traffic. The following table summarizes the papers used in related work and is relevant to ML in IDS.

Table 6.1: Overview of publications relevant to ML in IDS

| Ref. | Detection | Classification | Algorithm | Attack | Metrics | Pros-Cons |
|---|---|---|---|---|---|---|
| [15] | | | K-means | Scanning attacks<br>DoS<br>Penetration attacks | Detection<br>False positive<br>False negative<br>Clusters | A higher efficiency rate is achieved when the correct number of clusters is applied. |
| [18] | | | Bagging<br>Ensemble<br>Partial<br>Decision Tree<br>GA | Normal<br>Attack | Accuracy<br>True Positives<br>False Positives<br>Time | PDT showed high accuracy.<br>True positive rate and false positive rate better than all classifiers. IDS quite simple and accurate.<br>It requires more time to build the model. |

**Table 6.1 continued from previous page**

| Ref. | Detection | Classification | Algorithm | Attack | Metrics | Pros-Cons |
|---|---|---|---|---|---|---|
| [32] | Normal<br>or<br>attack | Binary<br>classification | Cluster center<br>K-NN | DoS<br>R2L<br>U2R<br>Probe<br>Normal | Accuracy<br>Detection<br>False alarms | Less time effort for training and testing.<br>Higher accuracy than K-NN and SVM.<br>Can not detect U2L and R2L attacks.<br>Centers and nearest neighbors performs the best in terms of detection rate and false alarm rate. |
| [5] | normal<br>or<br>attack. | | Naive Bayes<br>C 4.5 decision tree | DoS<br>R2L<br>U2R<br>Probe<br>Normal | Accuracy<br>Time | Naive Bayes and C 4.5 decision tree give higher accuracy.<br>Low time complexity.<br>Can be installed against real-time attacks. |
| [35] | normal<br>or<br>attack. | | AINRBF+NN<br>LSSVM. | DoS<br>R2L<br>U2R<br>Probe<br>Normal | Accuracy<br>Detection<br>False positive | Best accuracy is shown by AIN+RBFNN.<br>Best detection rate is noticed by LSSVM. |
| [17] | Detects four types of attacks. | 10 cross validation. | RF<br>J.48 | DoS<br>R2L<br>U2R<br>Probe | Accuracy<br>DR<br>MCC | RF<br>has shown:<br>good accuracy.<br>high detection rate.<br>low false alarm rate<br>-Mathew's correlation coefficient(MCC) is higher than J48 |
| [33] | | | SVM<br>Naive bayes<br>J.48<br>Decision table | DoS<br>R2L<br>U2R<br>Probe<br>Normal | True positive<br>False positive<br>Precision | Accuracy of J.48 is very much high among all other algorithms.<br>Misclassification rate is also low for J.48.<br>Naive Bayes has high misclassification rate as well as low accuracy rate. No single algorithm has high TPR for all different classes. |
| [2] | | | LDA<br>PCA<br><br>SVM | | Detection<br>Accuracy | Ensemble feature extraction methods showed a particularly excellent rate in detection rate. |

**Table 6.1 continued from previous page**

| Ref. | Detection | Classification | Algorithm | Attack | Metrics | Pros-Cons |
|---|---|---|---|---|---|---|
| [8] | | Support vectors do the classification task. | SVM<br>GA | | Accuracy<br>Precision<br>True positive<br>False positive | Higher accuracy of detecting intrusion.<br>The highest True positive rate. The lowest False positive rate. Minimum error rate.<br>Highest precision rate. |
| [10] | Gaussian Naive Bayes | Logistic regression | LR<br>Naive Bayes<br>SVM<br>RF | DoS<br>R2L<br>U2R<br>Probe | Precision<br>Recall<br>F1-score<br>Accuracy | RF classifier outperforms the other methods in identifying whether the data traffic is normal or an attack.<br>RF has the accuracy of 99%. |
| [48] | Attack or normal | | Average one dependence estimators | DoS<br>R2L<br>U2R<br>Probe | Accuracy<br>Detection<br>False alarm | AODE is efficient with low false alarm rate and high detection rate. Accuracy is improved. |
| [7] | | | FFNN | Normal<br>Anomaly | Accuracy | Classification accuracy is increased.<br>The samples belonging to the the mid fuzziness group have a higher risk of misclassification for IDS. |
| [37] | | | NN and SVM<br>SVM with radial kernel<br>K-means with PCA and GFR | DoS<br>R2L<br>U2R<br>Probe | Accuracy<br>Precision | NN+GFR2 supervised learning combination is detecting known attacks.<br>NN and K-means are good in detecting unknown attacks.<br>Hybrid models show better accuracy and precise detection of both known and unknown attacks. |
| [24] | Logit boost to detect known and unknown traffic | | Logit Boost<br>RF | Web attacks | Accuracy<br>Detection | This model shows superiority, regarding accuracy and detection rate over the traditional approaches.<br>It preserves low false rejection rates.<br>Reduced detection time. |

**Table 6.1 continued from previous page**

| Ref. | Detection | Classification | Algorithm | Attack | Metrics | Pros-Cons |
|---|---|---|---|---|---|---|
| [23] | | | RFAODE | Normal<br>Malicious traffic | Accuracy<br>False alarm | The proposed approach recorded high accuracy, with high detection rate and low false alarm rate. This approach can be used to detect network intrusion and classify traffic data as normal or malicious. |
| [1] | | | Naive Bayes<br>Best-First<br>J.48 | | Precision<br>Recall | NB tree and BF tree classifiers have the highest precision values, while Naive Bayes and BF tree classifiers have the highest recall values. The highest F-score values belong to Naive Bayes. Naive Bayes and Naive Bayes Tree had better results when all the training dataset was used. Normal data was highly detected and classified by all classifiers except NB classifier. |
| [42] | Attack or malicious | | IPDSKNN<br>PDSKNN | DoS<br>R2L<br>U2R<br>Probe | Accuracy<br>Time<br>Error rate | The proposed methods show remarkably high accuracy rate when used for classification. IKPDS has a less computational time compared to traditional K-NN and PKDS. |
| [9] | | | FMM<br>NN<br>and the PSO | DoS<br>R2L<br>U2R<br>Probe<br>Normal | Classification accuracy<br>Classification rate | Online adaption facility. Less time required for training. |
| [34] | | | SVM<br>K-NN<br>K-means | DoS<br>R2L<br>U2R<br>Probe | Accuracy<br>Detection<br>False alarm<br>Recall<br>Precision<br>F1-score<br>Time training | SVM and K-means have the best performance metrics. K-means is better in accuracy, false alarm rate and time training. SVM2, K-means2 and SVM+K means approach shows better results in all metrics. |

**Table 6.1 continued from previous page**

| Ref. | Detection | Classification | Algorithm | Attack | Metrics | Pros-Cons |
|---|---|---|---|---|---|---|
| [38] | | | SVM<br>Algorithm proposed<br>K-means<br>ANN<br>Back propagation | Fuzzers<br>analysis<br>Backdoors<br>DoS<br>Exploits<br>Reconnaissance<br>Shellcode<br>Worm | Accuracy<br>False positive | Algorithm proposed and ANN performed the best with high accuracy and low false positive rate. |
| [4] | RF detects anomalies. | | SVM<br>RF | | Accuracy<br>Precision<br>Recall<br>F1-score | Both methods enhance capabilities of industrial IDS. Both methods detect between 90-95 of attacks. |
| [20] | | SVM | LR<br>SVM<br>Decision Tree<br>RF<br>ANN | DoS<br>Probe<br>Malicious control<br>Malicious operation<br>Scan<br>Spying<br>Wrong setup | Accuracy<br>Precision<br>Recall<br>F1-score | RF performs better than the other techniques. Decision Tree, RF and ANN have high accuracy too. |
| [46] | | | LDA<br>Classification and Regression Trees<br>RF | DoS<br>R2L<br>U2R<br>Probe<br>Normal | Accuracy<br>Precision<br>Recall<br>F1-score<br>Detection<br>False positive | RF algorithm yield better accuracy than LDA and CART. The performance of the algorithms depends on the size of the dataset. |
| [41] | | | RF<br>LR<br>K-NN<br>SVM | | Accuracy | RF performs better in the given datasets |
| [39] | | SVM<br>Decision tree<br>K-NN<br>K-means | SVM<br>Decision trees<br>K-NN<br>K-means | Command and response injection attacks | Accuracy<br>F1-score | SVM, Decision trees, and K-NN had the best performance. K-means did not perform satisfactorily. |

# 7 Research questions and approach

We formed three important research questions based on the identified research problem and to meet the intended objectives. We now describe these questions and our answers as well as approach for solving them.

1. **What are the important security requirements that an intrusion detection system should fulfill?**
   Article published by ukdiss.com [52] enlists some of the desired requirements that an intrusion detection system should fulfill securely. Here we present some of those:

   - An ideal intrusion detection system must afford to detect attacks without producing false-negative and false-positive alarms (effectiveness aspect).
   - This system must report different types of attacks in the shortest possible time (efficiency aspect).
   - It should be fault-tolerant whenever an accident may happen.
   - Lastly, it should be easily deployed and configurable to implement the system's security policies.

   In this thesis we will focus to accomplish the first and the second security requirements.

2. **What architecture is suitable for meeting the requirements of an effective and efficient IDS?**
   In this study, we are focused on network intrusion detection systems. We did an investigation to understand better the significant problems and challenges that current systems have. The common challenges in intrusion detection systems are false alarm rate, low detection rate, unbalanced datasets, late response time, and testing environment. Mliki et al. [34] note that machine learning prepares software applications to become more accurate in detecting new attacks and preventing unauthorized access without being especially programmed, only by using example data. Machine learning algorithms are applied in intrusion detection to automatically identify and classify security threats due to their ability to learn and improve past experiences. These algorithms need input data to build a mathematical detection model, which can classify the network traffic.

   In [32] the authors state that machine learning algorithms build a self-learning system capable of autonomously obtaining and integrating knowledge by finding events that show unwanted deviations. The general purpose of using these methods in this thesis is to enhance the detection rate, reduce false positives, and improve prediction accuracy. Ferdiana et al. [32] in their paper, point out some of the techniques used on the IDS topic such as; clustering, classification, estimation, association, prediction, statistic, and dataset analysis. Section 3 mentioned the advantages and disadvantages of two anomaly detection methods: signature-based detection and behavior-based detection.
   The signature-based detection method provides more accuracy in finding anomalies. However, it recognizes only the signatures defined in the ruleset database. This technique requires constant updates about the new attacks. This task generally is done by the system analysts. On the other hand, the behavior-based detection method is more efficient in detecting anomalies called zero-day attacks. The drawback of this method is the high number of false-positive alarms.
   After we analyzed the shortcomings of both methods, numerous researches led us to the application of ML algorithms. The conjunction of the machine learning process with both methods made us adopt a hybrid solution. To mitigate these disadvantages, we got inspired by the work of Veeramachaneni et al.[53] to design a hybrid system. The authors have proposed an active modeling approach. They use a framework divided into three phases called training, deployment, and feedback updating. We have adopted their approach, and different from them, we have added the machine learning process in particular. In our framework, we will use clustering and classification for the selected dataset called BATADAL.

   In this way, it will see if it is possible to identify and classify anomalies in a shorter time and with greater accuracy. To further emphasize the contribution of this work, we give a more detailed explanation of the added components. We are looking for a time-effective solution, reducing the response time and having a low false-positive rate. The human factor has proved to be an inseparable part of security. Here, we ensure that the security analyst's expertise and intuition can be integrated and improve machine learning solutions.

**Proposed approach**
The network of an industrial system has a broad scope, making it more difficult to monitor its traffic. However, a module called the "Data gathering module" can facilitate data collection from many sensors installed on the network. Then through the transmission module, these unproven data are transmitted to the anomaly detection modules as log files. Dietz et al. [14], in their paper, mentions the Logstash tool to normalize data. From this point of view, we propose that logs coming from transmission modules can be analyzed in the "Data Analyzing Module" to facilitate further processing and readability. The data will go through three stages of processing. The first stage is related to the use of a behavior-based detection method. We figure out that unsupervised machine learning alone cannot achieve high classification accuracy. However, it can detect a more significant number of attacks. For us will be possible to find new anomalies that deviate from the threshold.
The novelty of our approach is adapting the machine learning process with behavior-based detection. We assume that this will make us able to identify the extreme and rare events in the data. The machine learning process starts with dividing data into two categories, training data, and testing data. This task can be done by one "Data divider module." As a start, the K-means algorithm can be applied to learn the malicious and normal data. Then a learned model will be created. This learned model is will be applied to testing data. As a result, we have the first detection and classification based on machine learning using an unsupervised detection technique. The model will pass to the third stage, which is signature-based detection. A correlation engine detects anomalies based on rules. This correlation engine dictates how much data matches the signatures it has defined in the labels. This comparison will give a more accurate classification of data. Anomalies will trigger an alarm in the visualization modules, where the graphical presentation of data occurs. Lastly, the security analyst will find it easier to define new attacks and rules at the "Rules Module." Repeating this process will increase the accuracy of finding and classifying intrusions using machine learning.

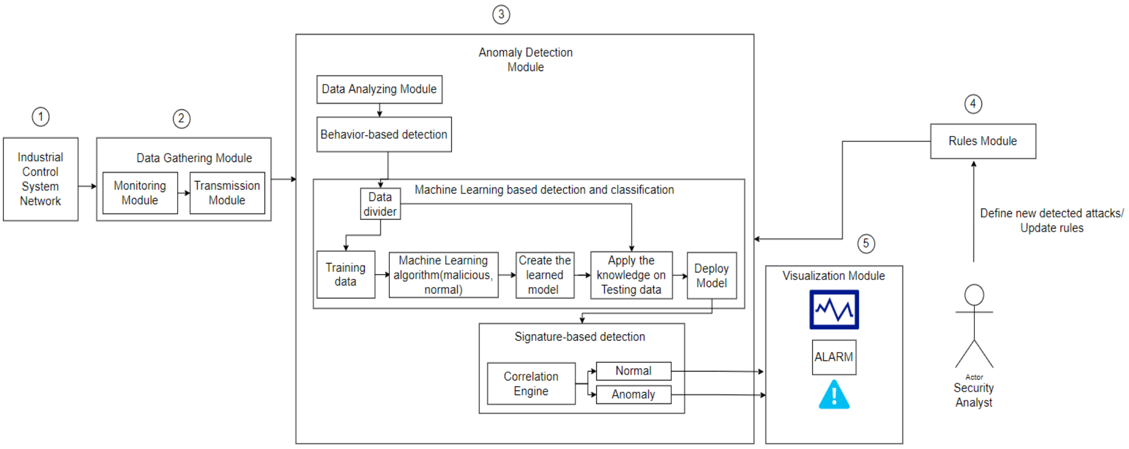


Figure 7.1: AI powered software architecture of IDS

3. **What are the machine learning algorithms used in intrusion detection systems that can detect and classify attacks?**
Based on the papers from the literature review, categorizing the used algorithms is necessary because it provides a better understanding for the reader to figure out which of these algorithms are used mostly. This classification is illustrated as follows.

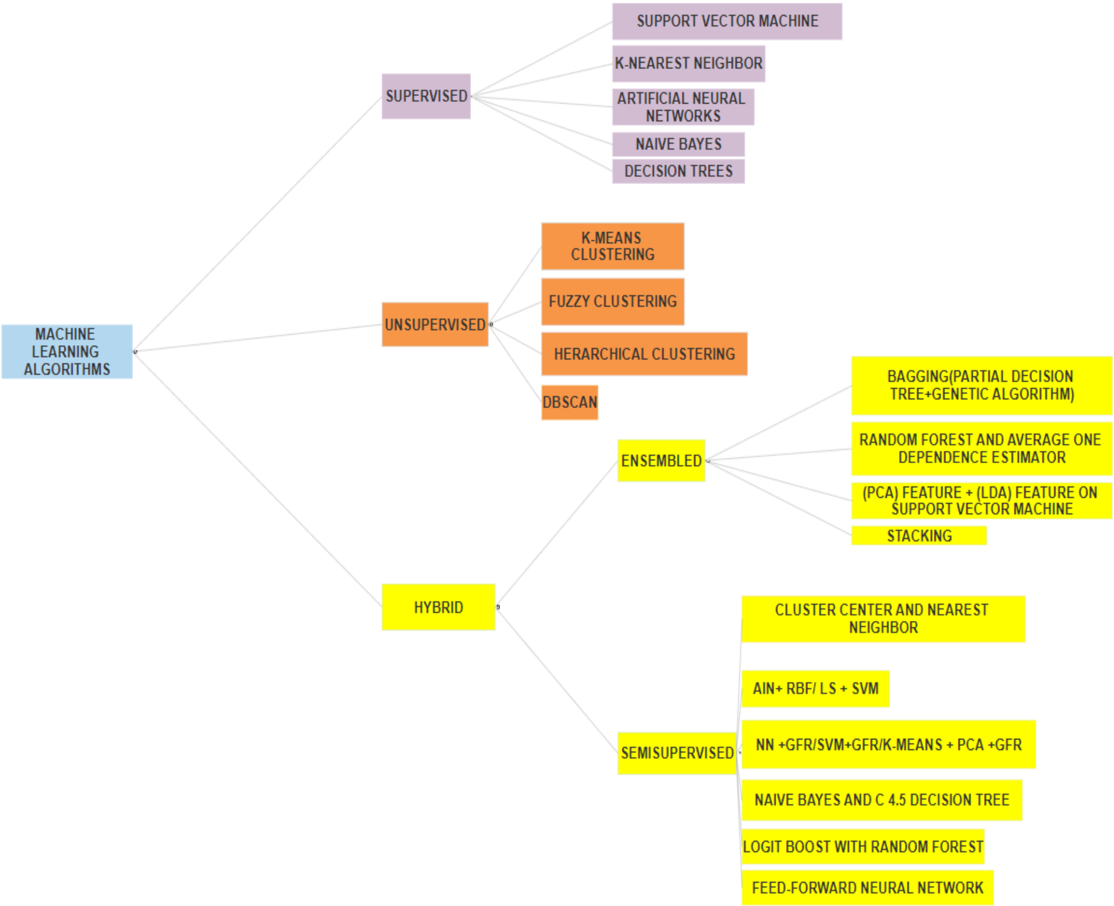

MACHINE LEARNING ALGORITHMS
SUPERVISED
SUPPORT VECTOR MACHINE
K-NEAREST NEIGHBOR
ARTIFICIAL NEURAL NETWORKS
NAIVE BAYES
DECISION TREES
UNSUPERVISED
K-MEANS CLUSTERING
FUZZY CLUSTERING
HERARCHICAL CLUSTERING
DBSCAN
HYBRID
ENSEMBLED
BAGGING(PARTIAL DECISION TREE+GENETIC ALGORITHM)
RANDOM FOREST AND AVERAGE ONE DEPENDENCE ESTIMATOR
(PCA) FEATURE + (LDA) FEATURE ON SUPPORT VECTOR MACHINE
STACKING
SEMISUPERVISED
CLUSTER CENTER AND NEAREST NEIGHBOR
AIN+ RBF/ LS + SVM
NN +GFR/SVM+GFR/K-MEANS + PCA +GFR
NAIVE BAYES AND C 4.5 DECISION TREE
LOGIT BOOST WITH RANDOM FOREST
FEED-FORWARD NEURAL NETWORK

Figure 7.2: Classification Diagram

The selected works included literature reviews, systematic mapping studies, conference papers, book series, and journals published between 2015 and 2021. At the end of the research process, we came up with 25 studies that were considered relevant and helpful for realizing our goals. We collected various algorithms which belong to different categories. Referring to the above classification tree diagram, we figured out that ten studies mention the Support Vector Machine algorithm in the supervised machine learning algorithms category. Next, we figured out that five papers have mentioned the K-means algorithm, which belongs to unsupervised machine learning algorithms. Lastly, we found ten other algorithms, which we put in the same so-called hybrid category. Our main objective is to select the most efficient algorithms in detecting and classifying attacks. Since we cannot test all of these algorithms, we decided to assess them based on the values obtained from our literature. We do this with the motive of choosing the algorithms that give us the best results. The selected algorithms will be applied and validated in our experiment later. The measurement metrics are accuracy, detection rate, false alarm rate, recall, and precision.
Looking at the pros-cons column in the literature review table, we realize that we have enough data about the Support Vector Machine algorithm. In four cases, authors at papers [8],[34], [4] and [39] present arguments to emphasize that this algorithm has shown high accuracy in intrusion detection, high True Positive rate, and low False Positive rate. Besides, it has higher precision values and a low minimum error rate compared to the other ones. Support Vector Machine outperforms the other supervised algorithms.

K-means is one of the most used algorithms in the unsupervised ML algorithms category. In three out of five papers related to the K-means algorithm, authors highlighted the following performance evaluations: The assumptions at [15]seem acceptable, saying that k-means achieves a higher efficiency rate when the correct number of clusters is applied. Perez et al.[37] conclude that the combination of neural networks with the k-means algorithm is good in detecting unknown attacks. The single most striking observation to emerge from the data comparison is that k-means combination with the SVM2 has shown better results in all metrics such as high accuracy, low false alarm rate, and low time training in intrusion detection. However, its detection rate is not very high because, unlike supervised category algorithms, unsupervised algorithms have drawbacks, which is their high false alarm rates.

As mentioned in the background section, hybrid algorithms are a combination of more than two other algorithms. This combination is done to minimize the shortcomings of algorithms in detecting and classifying normal and malicious data. Farnaaz et al.[17] further assert that Random Forest has shown good accuracy, high detection rate, low false alarm rate than the J48 tree. Belavagi et al.[10] additionally emphasize that RF outperforms the other methods in identifying whether the data traffic is regular or an attack, showing 99% accuracy. Kamarudin, in his work [24], draws our attention to the proposed Logitboost algorithm combination with Random Forest. This model shows superiority regarding accuracy and detection rate over the traditional approaches. Even more, it preserves a low false alarm rate and reduces detection time. Jabbar et al.[23] found a novel intrusion detection system utilizing Random Forest and AODE algorithms. He proved that this approach recorded high accuracy, high detection rate, and low false alarm rate. Additionally, he advocates that RFAODE can be helpful to detect network intrusion and classify traffic data as usual or malicious. The experimental setups studied at [4] bear a close resemblance of SVM and Random Forest performance. Here, both methods enhance the capabilities of industrial IDS and are capable of detecting between 90-95% of attacks. The compared results on [20] demonstrate that Random Forest performs better than the other techniques, showing that this algorithm has high accuracy above the others. Performance analysis of ML algorithms in [46] describes that RF yields better accuracy than LDA and CART. In the given datasets at [41], Random Forest performs better.

We decided to choose the Support Vector Machine algorithm from supervised algorithms, K-means from the unsupervised methods. Finally, Random Forest as an ensemble method to test in our experiment. Regardless of which algorithms are combined, we must understand that hybrid algorithms convincingly increase performance in IDS. Still, performance is closely related to the data contained in the dataset because different datasets produce different results.

# 8 Experimental setup and evaluations

In this section we describe how we conducted the experiment with the ML algorithms and show their performance results.
First we give a brief introduction to our planned experiment. Ideally we would prefer to conduct this experiment with an actual setup by applying different attacks on a target network. Next stage, we would prefer to collect the traffic data logs, and in the last step, we would like to apply our chosen machine learning algorithms. At the final stage, we would prefer to detect and classify the data traffic based on predefined rules using our selected algorithms and see their performance. However, the time was not on our side to model such attacks on a real system implemented by us, so that we could visualize the performance of the proposed algorithms on that system. Hence, we had to choose a ready-made dataset that contains one attack type similar to the one that we wished to model on the target network. The target network corresponds to the water distribution system, and our selected dataset corresponds to the traffic logs. We have followed the experiment steps that are similar for both the scenarios, which is applying a machine learning process combining both the usage of clustering and classification algorithms. We are going to give an in-depth description of the steps that we used in our experiment. In machine learning, the first step is to make the algorithms familiar with the data they will recognize. For this purpose, we divided our dataset into two parts, namely, training data, and testing data. Since we wanted the algorithm to learn a

comprehensive network traffic model, we partitioned the data as 80% in the training dataset and 20% in the testing dataset.

From the literature review, we learned that clustering algorithms perform better in detecting unknown attacks, so we considered this knowledge to apply it in seeing the unlabelled data in our dataset. We used the K-means algorithm for clustering the data. The output of this stage would be the K-means model, which contains data distributed in different clusters. These clusters would be used to train the classification models of SVM and RF algorithms. SVM and RF algorithms would classify the data. We prepare the dataset with SVM and RF to have their models. We would add the clusters as a feature(column)to train the same SVM and RF models. To measure the performance adequacy of these algorithms, we upload these models on the test dataset and compare the results to see any improvement when we apply this combined process.

The process that we followed to setup the experiment is described in figure 8.1.

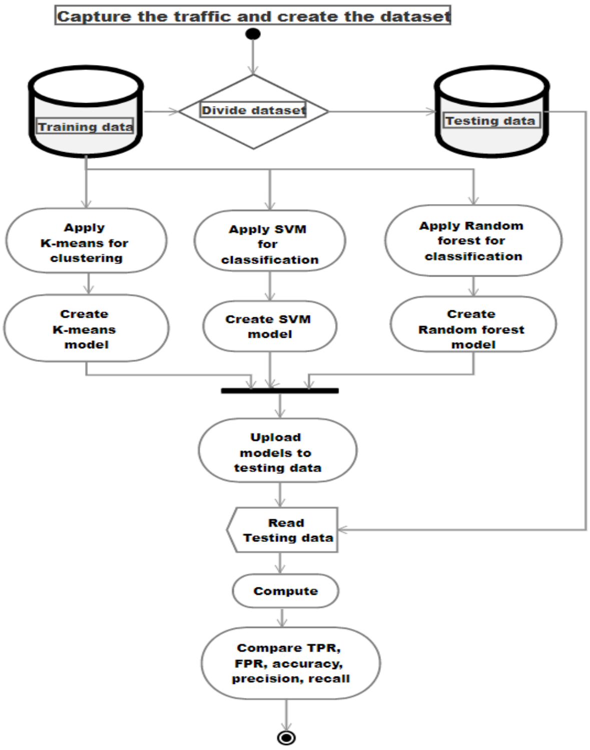

Capture the traffic and create the dataset
Training data
Divide dataset
Testing data
Apply K-means for clustering
Apply SVM for classification
Apply Random forest for classification
Create K-means model
Create SVM model
Create Random forest model
Upload models to testing data
Read Testing data
Compute
Compare TPR, FPR, accuracy, precision, recall

Figure 8.1: Process followed in experiments

## 8.1 Case and subjects selection

The case studied is improving the intrusion detection process in an industrial control system by adapting machine learning algorithms as a solution. Since we do not have real data on which we can make the experiment and the evaluations, we decided to handle data from an open-source dataset, specifically a water distribution system, provided by iTrust. iTrust is a multidisciplinary research center located at the Singapore University of Technology and Design.

Water distribution systems collect reservoirs, tanks, pumps, valves, and pipes that deliver water to our taps. All sorts of smart devices are added to these systems, including sensors that can measure water level in a tank or the pressure in a pipe and programmable logic controller that can do things like automatically turning on a pump when a tank is close to empty. While all these devices can allow the system to run more reliably and efficiently, they also expose the system to potential attacks on the software that controls it. If a hacker can remotely access the components of such a system, they could do all sorts of damage. Attacks range from simply stealing data to damaging equipment, cutting off the water supply, or even releasing excess treatment chemicals into the system. Hackers can spy on the system with eavesdropping attacks, overload the system with denial of service attacks, and send bogus data with deception attacks. Most of these attacks will have noticeable effects on the system, such as strangely low pressures. However, hackers can even cover their footprints by sending back data and pretending that everything is fine. This means that can be tricky to detect attacks on the system either with human operator or detection algorithms. Working in real-time systems can not always be possible since researchers may not have direct access to the industrial control systems. Still, the primary reason is that companies do not prefer to share their network traffic since it is a matter of security. However, it is possible to access some of the early prepared datasets.

## 8.2 Data collection

We see data collection in two points of views: Literature data collection and experimental data collection.

We collected data for the literature review from two digital databases IEEE Xplore and Google Scholar. For the literature review, we collected the most recent papers published between 2015-2021. For the experiment, we connected to iTrust, and they provided some of the datasets mentioned above. BATADAL dataset [50] contains data from a simulated water distribution system called C-Town Public Utility. C-town is based on a real-world medium-sized network. Water storage and distribution across the demand nodes is guaranteed by seven tanks, those whose water levels trigger the operations of one valve and eleven pumps distributed in 5 pumping stations. Pumps, valves, and tank water level sensors are connected to nine PLCs located near the hydraulic components. C-town has a SCADA system that collects the readings from all PLCs and coordinates the operations of the entire network.

The first training dataset was released on November 20, 2016. It contains data without attacks. This dataset was used to study the normal system operation.

The second training dataset was released on November 28, 2016. Researchers applied six different attack scenarios which targeted various components and repeated them hundreds of times under different conditions. The attacks resulted in pump malfunctions, and tanks are overflowing or emptying down to deficient levels. Exactly, a water overflow in T1 occurred unexpectedly while the water level readings were below the alarm thresholds and showing pumping operations as usual. It also contains attacks that cause anomalous low levels in Tank 5 and high levels in Tank 1. These are labeled attacks. The normal data flag is assigned to 0, and attack data flag is given to 1.

The third one is the test dataset released on February 20, 2017. It is used to compare the performance of the algorithms that we selected. All the data are provided in .csv format and can be reached by sending an email to iTrust.

The below figure represents a graphical view of a water distribution system.

Table 8.1: Water distribution system components; PLCs, sensors, pumps, tanks and valves [51]

| PLC | Sensor | Actuators(controlling sensor) |
|---|---|---|
| PLC1 | - | PU1(T1), PU2(T1) |
| PLC2 | T1 | - |
| PLC3 | T2 | V1(T2), PU4(T3), PU5(T3), PU6(T4), PU7(T7) |
| PLC4 | T3 | - |
| PLC4 | T4 | PU8(T5), PU9(-), PU10(T7), PU11(T7) |
| PLC5 | T5 | - |
| PLC6 | T6 | - |
| PLC7 | T7 | - |
| PLC8 | T8 | - |
| PLC9 | T9 | - |

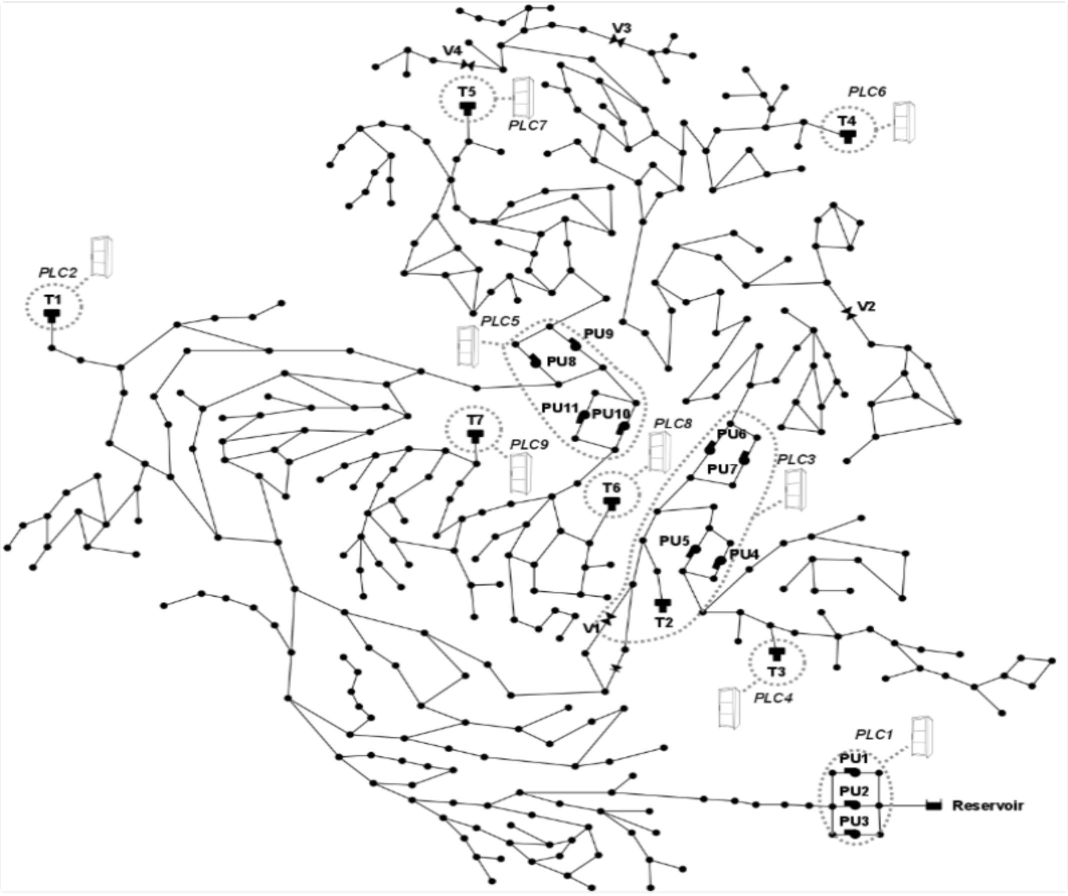


Figure 8.2: C-Town Public Utility Water Distribution system [51]

A clear presentation of the programmable logic controllers(PLCs), sensors(S), pumps(PU), tanks(T) and valves(V) distribution is given as follows: A description of the attack data is given as follows:

Table 8.2: Attacks in BATADAL dataset [51]

| ID | Attack description | SCADA concealment |
|---|---|---|
| 1 | Attacker changes LT7 thresholds(which controls PU10/PU11) by altering SCADA transmission to PLC9. Low levels in T7. | Replay attack on LT7. |
| 2 | Like attack 1. | Like Attack1 but replay attack extended to PU10/PU11 flow and status. |
| 3 | Attack alters LT1 readings sent by PLC2 to PLC1, which reads a constant low level and keeps pumps PU1/PU2 ON. Overflow in T1. | Polyline to offset LT1 increase. |
| 4 | Like Attack3. | Replay attack on LT1, PU1/PU2 flow and status, as well as pressure at pumps outlet. |
| 5 | Working speed of PU7 reduced to 0.9 of nominal speed causes lower water levels in T4. | |
| 6 | Like Attack 5, but speed reduced to 0.7. | LT4 drop concealed with replay attack. |
| 7 | Like Attack 6. | Replay attack on LT1, as well as PU1/PU2 flow and status. |

## 8.3 Performance Evaluation

Performance evaluation metrics in machine learning algorithms evaluate the machine learning framework performance with the input data and help predict how well it will work on new data. A confusion matrix is a technique of visualizing the relationship between the current outcomes and the predicted ones. It is used to evaluate the prediction accuracy of an algorithm or classifier. We defined the performance metrics definition referring to the Kumar et al.[30] work.

1. True Positive (TP): Attack is correctly classified as an attack.
2. False Positive(FP): Normal is incorrectly classified as an attack.
3. True Negative(TN):Normal is correctly classified as normal
4. False Negative(FN):Attack is incorrectly classified as normal.

The machine learning algorithm's performance is evaluated based on the below-mentioned metrics.

### 8.3.1 Accuracy

Accuracy is one of the elementary metrics for the performance evaluation of any algorithm. Mliki et al. [34] define accuracy as the ratio of the total number of true positive and true negative that are correctly detected and divided by the total amount of the dataset positive and negative amount.

The given formula calculates it as follows:

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

### 8.3.2 Detection Rate

Detection Rate addresses the accurately recognized anomalous traffic occurrences. Bose et al. [49] define as the ratio of the number of correctly classified records in a particular class to the total number of records of that class. The best outcome is equivalent to 0.0, while the worst one is equivalent to 1.0. The given formula calculates it:

$$DetectionRate = \frac{TP}{TP + FP}$$

### 8.3.3 False Alarm Rate

False Alarm Rate is also known as false-positive rate. It represents the proportion of incorrectly produced alerts for normal records to the total number of normal records. Its formula is:

$$FalseAlarmRate = \frac{FP}{FP + TN}$$

### 8.3.4 Recall

Ït is about the number of relevant instances detected. It addresses the proportion of positive prediction to the total number of positive prediction. In other words, it decides how much valuable data is available from any machine learning method. The recall formula is defined as:

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

### 8.3.5 Precision

It is defined as the number of relevant instances among the detected instances. The given formula calculates it:

$$Precision = \frac{TF}{TP + FP}$$

In this dataset, we decided to test the performance of SVM, K-means, and Random Forest algorithms because, from the above analysis in section 5, these algorithms turned out to be the most efficient. We conducted our experiment using the selected algorithms on an open-source tool called Weka to provide persuading proof. Weka is a machine learning software. This tool has its own GUI or can be accessed by standard terminal applications or a Java API and is widely used for teaching, research, and industrial applications. The experimental process is described as follows:

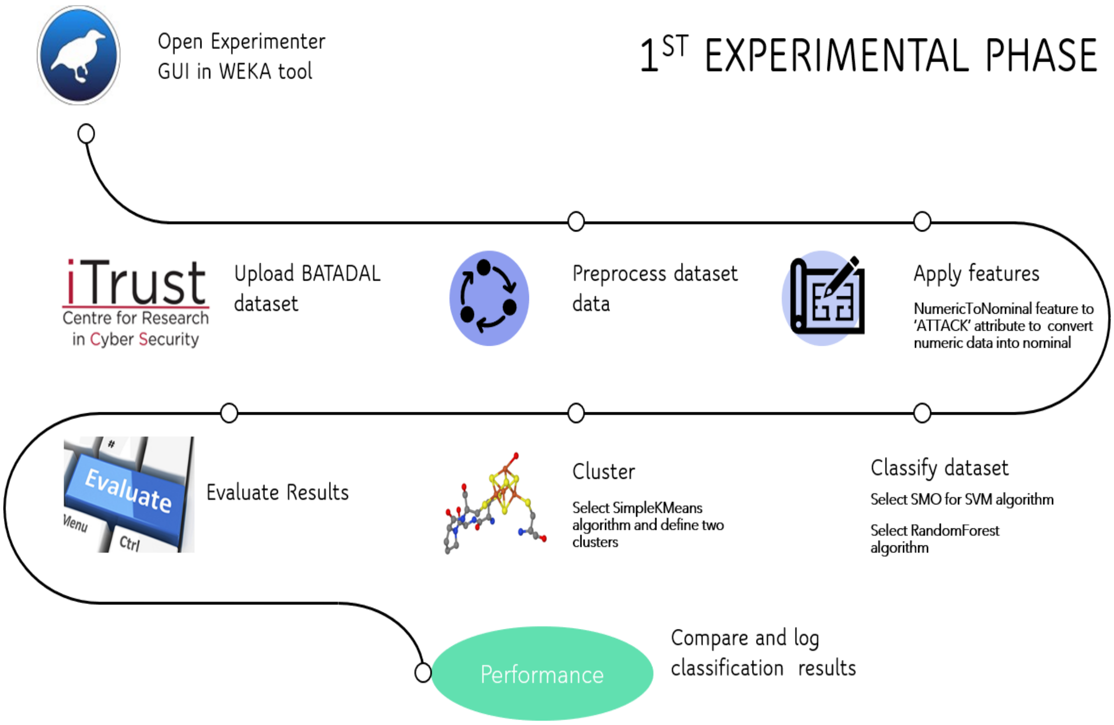
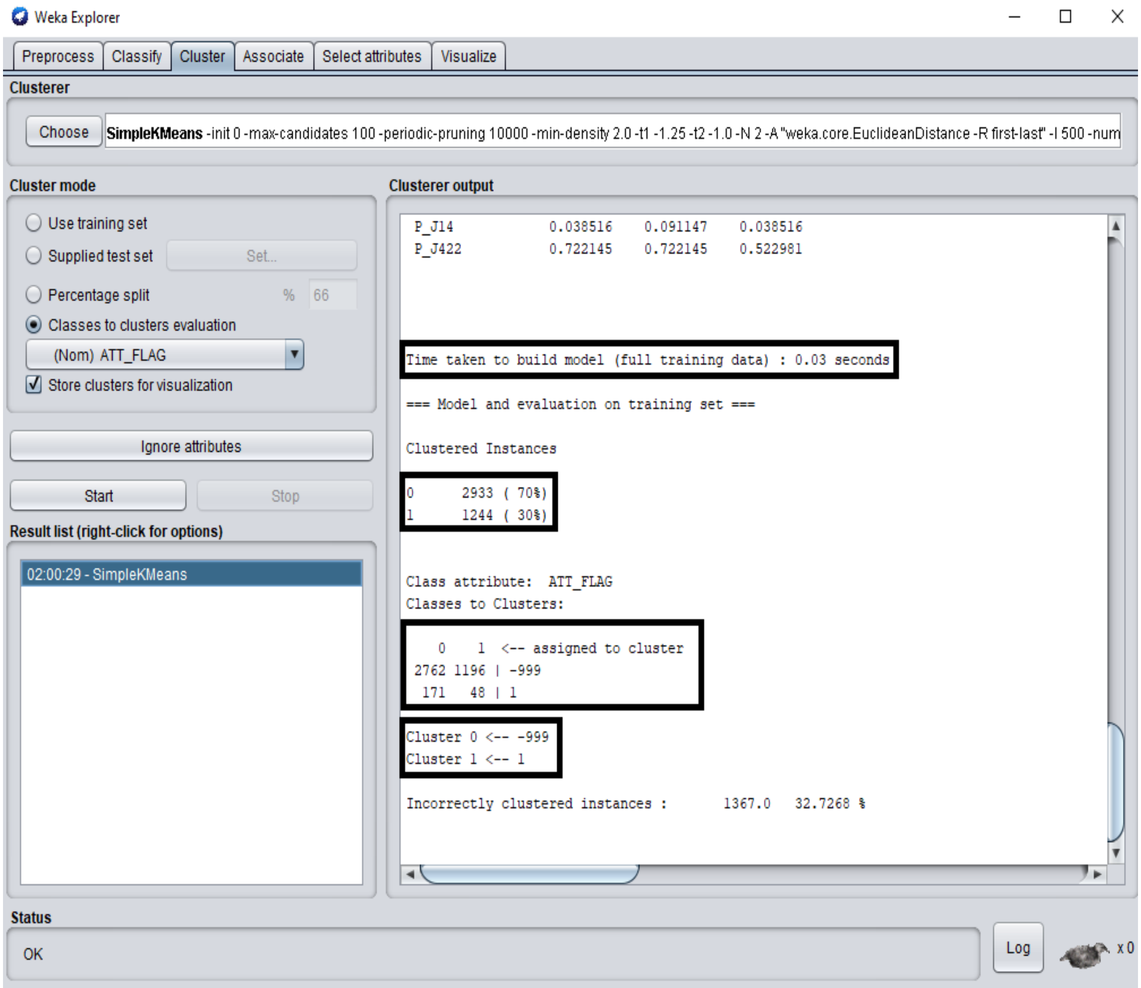

Open Experimenter
GUI in WEKA tool
1ST EXPERIMENTAL PHASE
iTrust
Centre for Research in Cyber Security
Upload BATADAL dataset
Preprocess dataset data
Apply features
NumericToNominal feature to 'ATTACK' attribute to convert numeric data into nominal
Evaluate
Menu
Ctrl
Evaluate Results
Cluster
Select SimpleKMeans algorithm and define two clusters
Classify dataset
Select SMO for SVM algorithm
Select RandomForest algorithm
Performance
Compare and log classification results

Figure 8.3: Experimental Process

Weka tool version 3.8.5 has five interfaces in the GUI chooser window. They are Explorer, Experimenter, Knowledge Flow, Workbench, and SimpleCLI(Command Line Interface). The Explorer, Knowledge Flow, and SimpleCLI handle data preprocessing, classification, regression, clustering, and association. In contrast, the Experimenter and Workbench address only classification and regression problems. Each interface has different functionalities. For our experiment, we need to use Explorer for the clustering algorithm. Initially, we open the Experimenter interface since it can handle datasets with a large amount of data. Next, we start a new experiment by uploading the BATADAL dataset that we downloaded from iTrust. Via the experimenter, we shift to the explorer interface to do the preprocessing. At preprocessing, it is essential to apply some features before selecting the algorithms. At the filter section, we choose unsupervised attributes and select the NumerictoNominal feature for the "Attack" attribute. The main reason for converting the numeric values into nominal data type is that in this dataset, it is required that the "Attack" attribute contain nominal values. The next step is to select our proposed machine learning algorithms. At the "Cluster" option, we choose the SimpleKMeans algorithm and define the number of clusters at two. We applied the default number of two clusters, cluster 0, which stands for normal data, and cluster 1, which stands for attacks. Looking at the confusion matrix, we can analyze how good this SimpleKmeans model is in terms of what it gets right and what it gets wrong. First of all, we need to see the time taken to build the model, which is a pretty short time equal to 0.03 seconds. The matrix tells us that row "0" stands for normal data and row "1" stands for anomalies.
Row "0" contains all the samples the model thinks are "0", corresponding to 2933 or 70% in total.
Row "1" has all the samples the model thinks are "1", corresponding to 1244 or 30% in total.
Knowing this interpretation, we can dig into details.

- Top left, 2762 are correctly detected normal data.
- Bottom left, 171 are correctly predicted anomaly data.
- Top right, 1196 are incorrectly predicted normal data.
- Bottom right, 48 are incorrectly predicted anomaly data.

The visualization of the results in Weka tool is given as follows:



Clusterer

SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num

Cluster mode

○ Use training set
○ Supplied test set
○ Percentage split % 66
● Classes to clusters evaluation
(Nom) ATT_FLAG
☑ Store clusters for visualization

Result list (right-click for options)

02:00:29 - SimpleKMeans

Clusterer output

```
 P_J14           0.038516    0.091147    0.038516
 P_J422          0.722145    0.722145    0.522981



Time taken to build model (full training data) : 0.03 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      2933 ( 70%)
1      1244 ( 30%)


Class attribute:  ATT_FLAG
Classes to Clusters:

    0    1  <-- assigned to cluster
 2762 1196 | -999
  171   48 | 1

Cluster 0 <-- -999
Cluster 1 <-- 1

Incorrectly clustered instances :       1367.0   32.7268 %
```

Status

OK

Figure 8.4: K-Means Performance

For this analysis, we used the training dataset and understood that K-Means seems to have a high false-positive alarms rate. The next stage of the experiment is to test which of the two selected algorithms, Support Vector Machine or Random Forests, has the best performance. To distinguish this difference, we tested these two algorithms in the BATADAL dataset.

**Experiment with Support Vector Machine** We uploaded BATADAL dataset to the WEKA tool. This dataset contains 4178 instances of data. We clicked on "Classify," and the "Choose Classifier" option appeared. Next, we selected the LibSVM package, which we previously uploaded via "Tools" to "Packet Manager." Weka allows us to choose how we want to test the dataset. Our objective is related to the ability of machine learning algorithms to predict anomalies in data traffic. For such reason, we choose the "Percentage Split" option, where we set the value to 80%. This value means that the SVM algorithm will distinguish 80% of the dataset to train the model. The remaining 20% of the data is used to test the created model of the SVM algorithm. The results given to us by the SVM algorithm are a prediction on the data defined as "Test Data." Time taken to test the model on test split is 0.07 seconds. Data is categorized in two main categories: "a," which stands for the normal data traffic, and "b" which stands for anomalies.
We will focus only on the above-defined five metrics and confusion matrix.

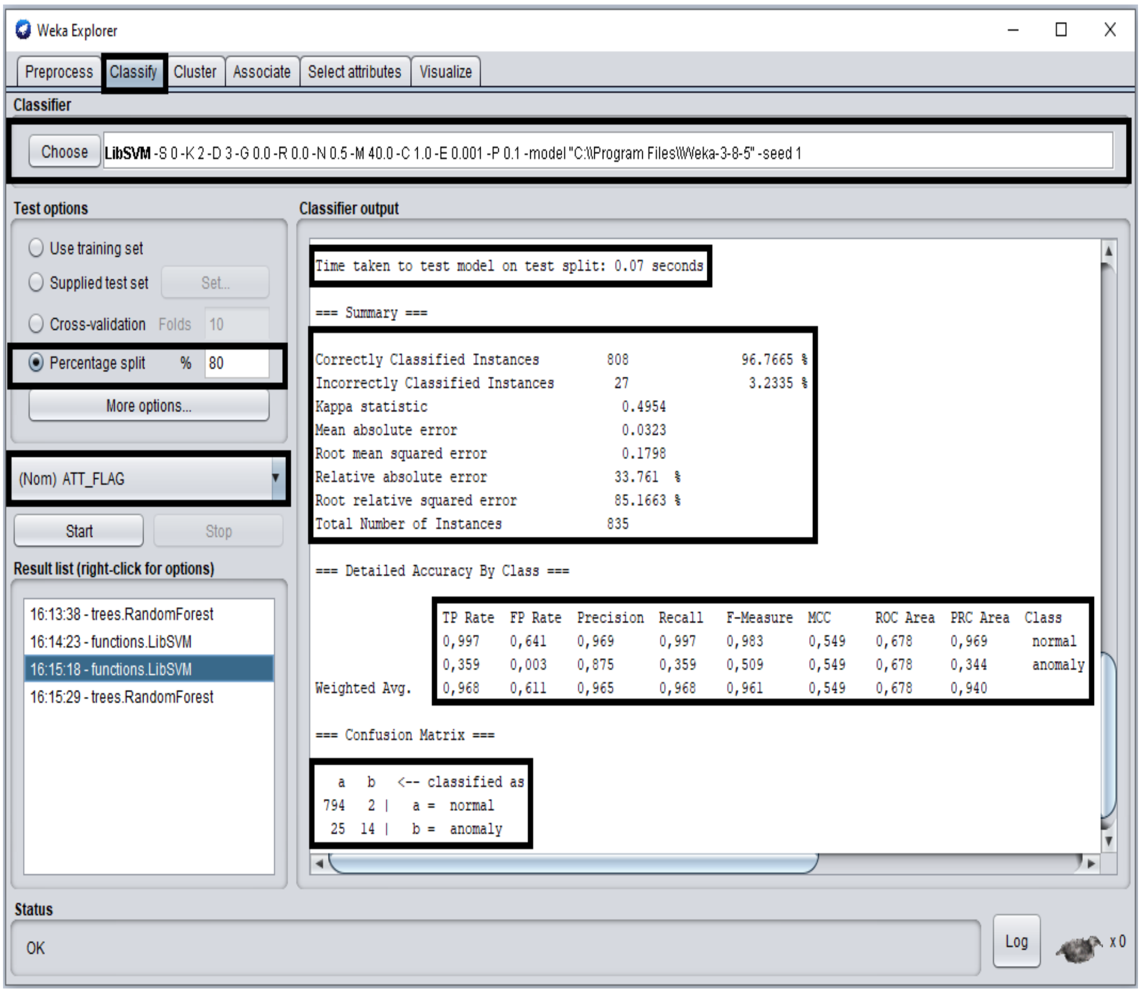
- Starting from the average True Positive rate we can say that this algorithms classified 96.8% of the data correctly.
- False positive rate seems to be 61.1%.

- Precision value is 96.5%.
- Recall value is 96.8%.

It is important to analyze the confusion matrix values as well:

- Top left, 794 data represent the True Positive rate, meaning that were correctly detected as normal data.
- Bottom left, 25 data represent the False Negative rate, meaning that are correctly predicted anomalies.
- Top right, 2 data represent the False Positive rate, meaning that are incorrectly predicted normal data.
- Bottom right, 14 data represent the True Negative rate, meaning that are incorrectly predicted anomalies.

The below figure shows the experimental results of the Support vector machine algorithm:



```
LibSVM -S 0 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model "C:\\Program Files\\Weka-3-8-5" -seed 1
```

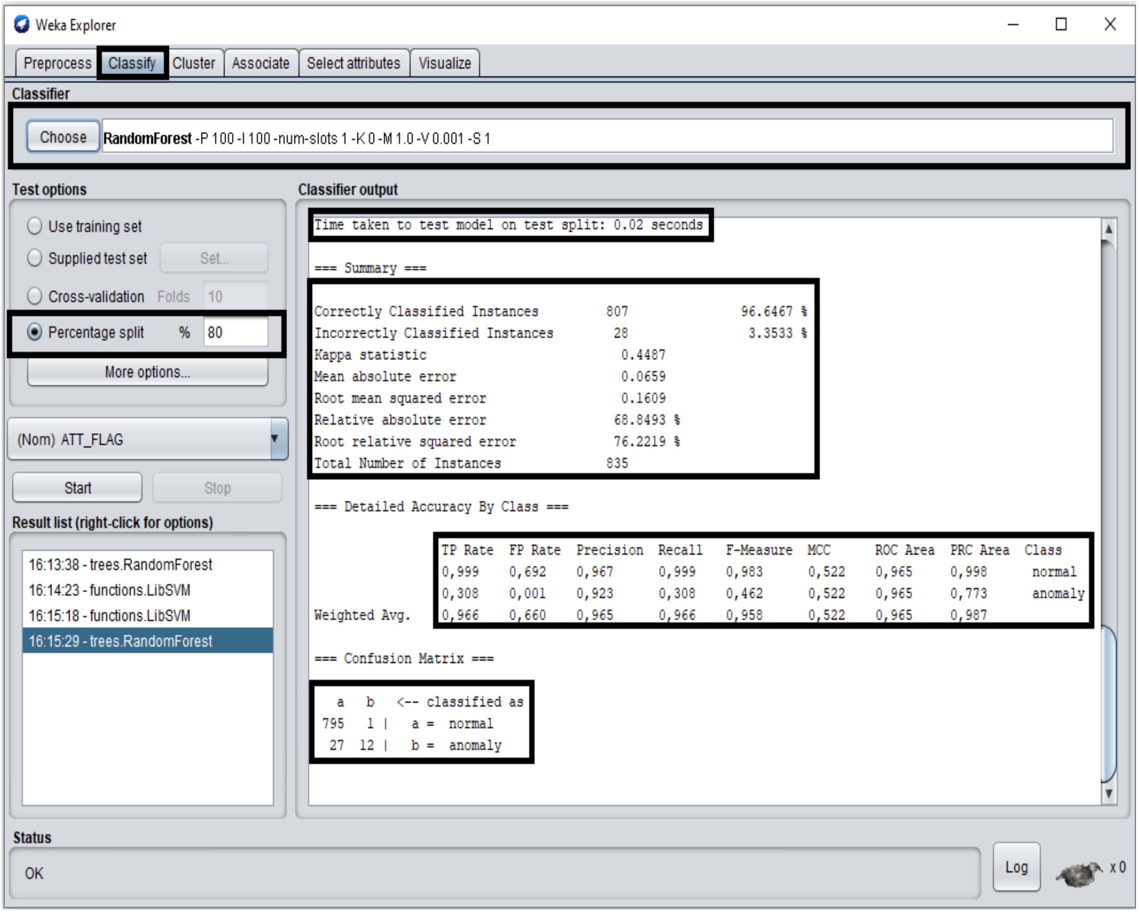
```
Time taken to test model on test split: 0.07 seconds

=== Summary ===

Correctly Classified Instances         808               96.7665 %
Incorrectly Classified Instances        27                3.2335 %
Kappa statistic                          0.4954
Mean absolute error                      0.0323
Root mean squared error                  0.1798
Relative absolute error                 33.761  %
Root relative squared error             85.1663 %
Total Number of Instances              835

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,997    0,641    0,969      0,997    0,983      0,549    0,678     0,969     normal
                 0,359    0,003    0,875      0,359    0,509      0,549    0,678     0,344     anomaly
Weighted Avg.    0,968    0,611    0,965      0,968    0,961      0,549    0,678     0,940

=== Confusion Matrix ===

   a   b   <-- classified as
 794   2 |   a = normal
  25  14 |   b = anomaly
```

Figure 8.5: Support Vector Machine Performance

To summarize the results, we can say that correctly classified instances are 808 and incorrectly classified instances are 27. Lastly, we want to see the performance of that Random forest algorithm in the same dataset.

**Experiment with Random Forest algorithm**

At the "Classifier" option, we selected the "Random Forest" ensemble algorithm. Following the same steps as in the Support Vector Machine case, we chose to test the dataset and again selected the "Percentage Split" option, where we set the value to 80%. This value means that the Random Forest algorithm will split 80% of the dataset to train the model. The remaining 20% of the data is used to test the created model using the Random Forest algorithm. The results given to us by the Random Forest algorithm are a prediction on the data defined as "Test Data." Time taken to test model on test split is 0.02 seconds. Data is categorized in two main categories:"a" which stands for the normal data traffic, and "b" which stands for anomalies.
The figure below shows the experimental results of the Random Forest algorithm:



RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

● Percentage split % 80

(Nom) ATT_FLAG

16:13:38 - trees.RandomForest
16:14:23 - functions.LibSVM
16:15:18 - functions.LibSVM
16:15:29 - trees.RandomForest

```
Time taken to test model on test split: 0.02 seconds

=== Summary ===

Correctly Classified Instances         807               96.6467 %
Incorrectly Classified Instances        28                3.3533 %
Kappa statistic                          0.4487
Mean absolute error                      0.0659
Root mean squared error                  0.1609
Relative absolute error                 68.8493 %
Root relative squared error             76.2219 %
Total Number of Instances              835

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,999    0,692    0,967      0,999    0,983      0,522    0,965     0,998     normal
                 0,308    0,001    0,923      0,308    0,462      0,522    0,965     0,773     anomaly
Weighted Avg.    0,966    0,660    0,965      0,966    0,958      0,522    0,965     0,987

=== Confusion Matrix ===

   a   b   <-- classified as
 795   1 |   a = normal
  27  12 |   b = anomaly
```

OK

Figure 8.6: Random Forest Performance

The Random Forest algorithm showed these values for each metric.

- The precise accuracy by class shows that the True Positive rate is 96.6%.
- False positive rate seems to be 66%.
- Precision value is 96.5%.
- Recall value is 96.6%.

Confusion matrix analysis shows that:

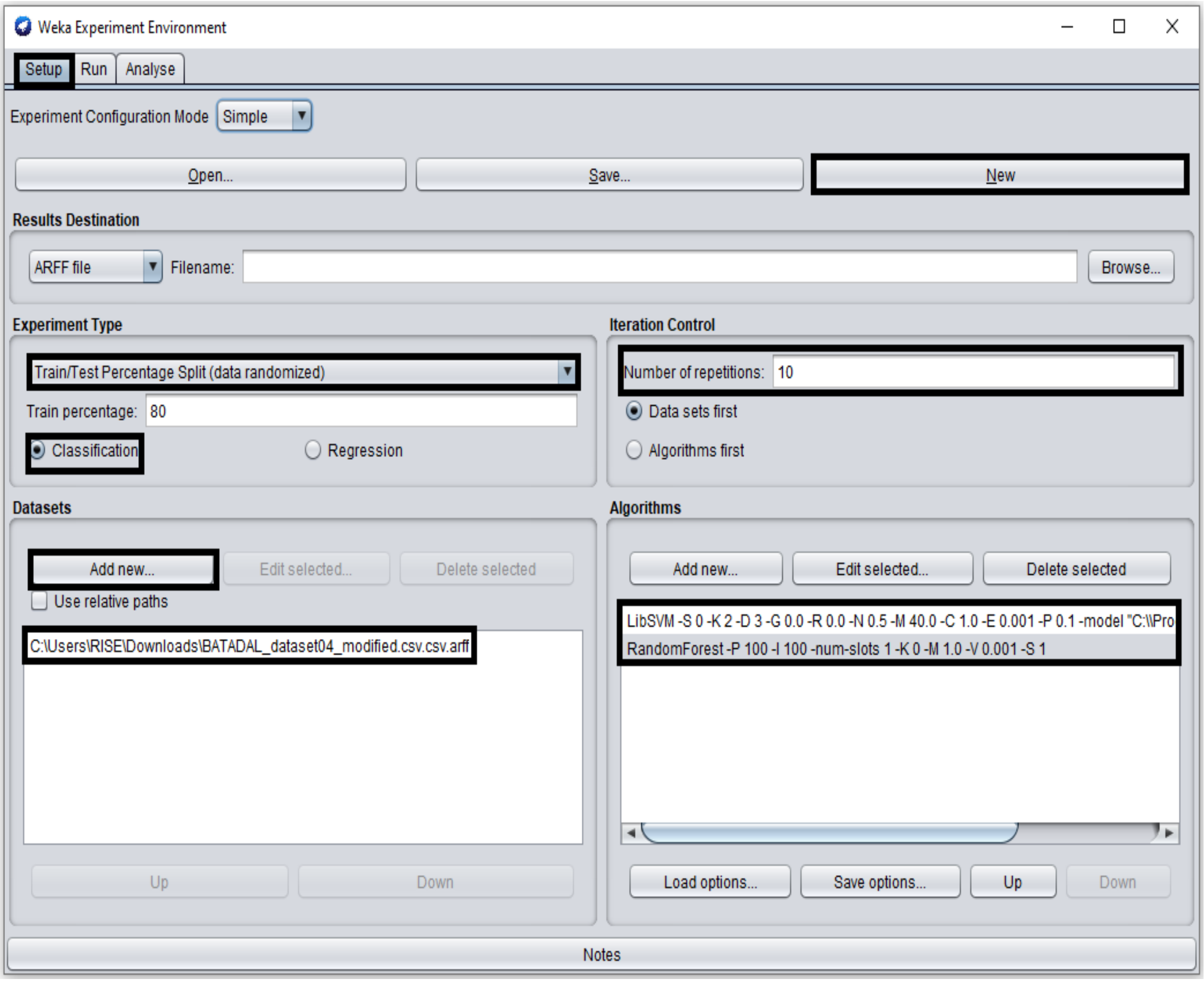
- Top left, 795 data represent the True Positive rate, meaning that they were correctly detected as normal data.
- Bottom left, 27 data represent the False Negative rate, meaning that are correctly predicted anomalies.

- Top right, 1 data represent the False Positive rate, meaning that is incorrectly predicted normal data.
- Bottom right, 12 data represent the True Negative rate, meaning that are incorrectly predicted anomalies.

The main principle of ML algorithms is to create models from the available input data, and through these models, to be able to predict new ones. For such reason, we evaluated the selected machine learning algorithms based on the ability to detect and classify the data as normal or attack.

# 9 Results

In this section, we will compare the performance metrics values between Support Vector Machine and Random Forest. To find the accuracy of the two algorithms, we used the "Experimenter" interface of the Weka tool. By selecting the "New" option automatically, the other features are enabled. Since we are interested in classification, we let the default experiment type to classification. In the datasets section, we uploaded the BATADAL dataset. Next to it, we added our selected algorithms, LibSVM and Random Forest.



Experiment Configuration Mode: Simple

ARFF file Filename:

Train/Test Percentage Split (data randomized)

Train percentage: 80

● Classification ○ Regression

Number of repetitions: 10

● Data sets first

○ Algorithms first

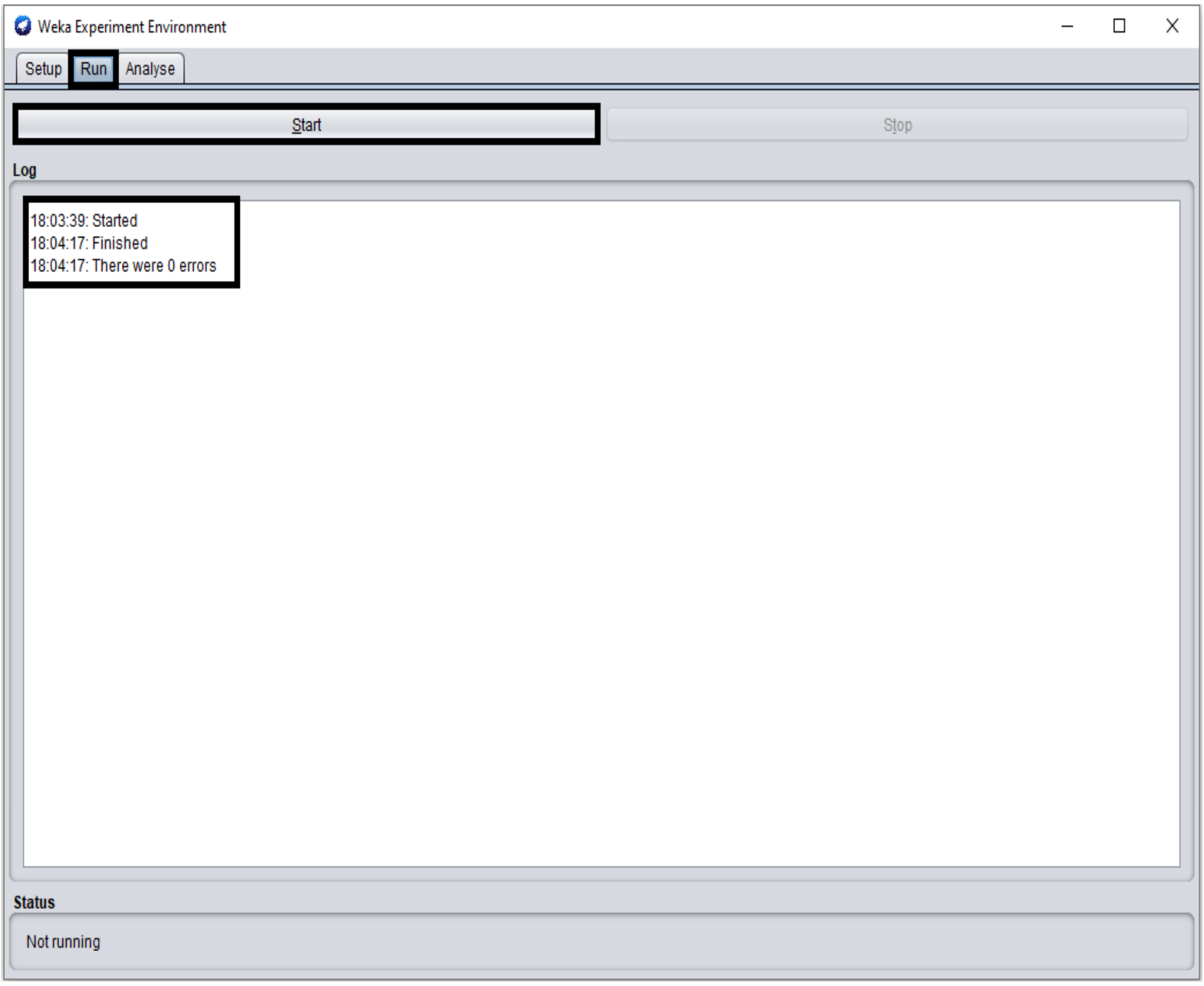
□ Use relative paths

C:\Users\RISE\Downloads\BATADAL_dataset04_modified.csv.csv.arff

```
LibSVM -S 0 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model "C:\\Pro
RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1
```

Figure 9.1: Selecting the dataset and the algorithms

Secondly, at the "Run" option, we pressed the "Start" button and let the evaluation be run. It took 38 seconds in total.



```
18:03:39: Started
18:04:17: Finished
18:04:17: There were 0 errors
```

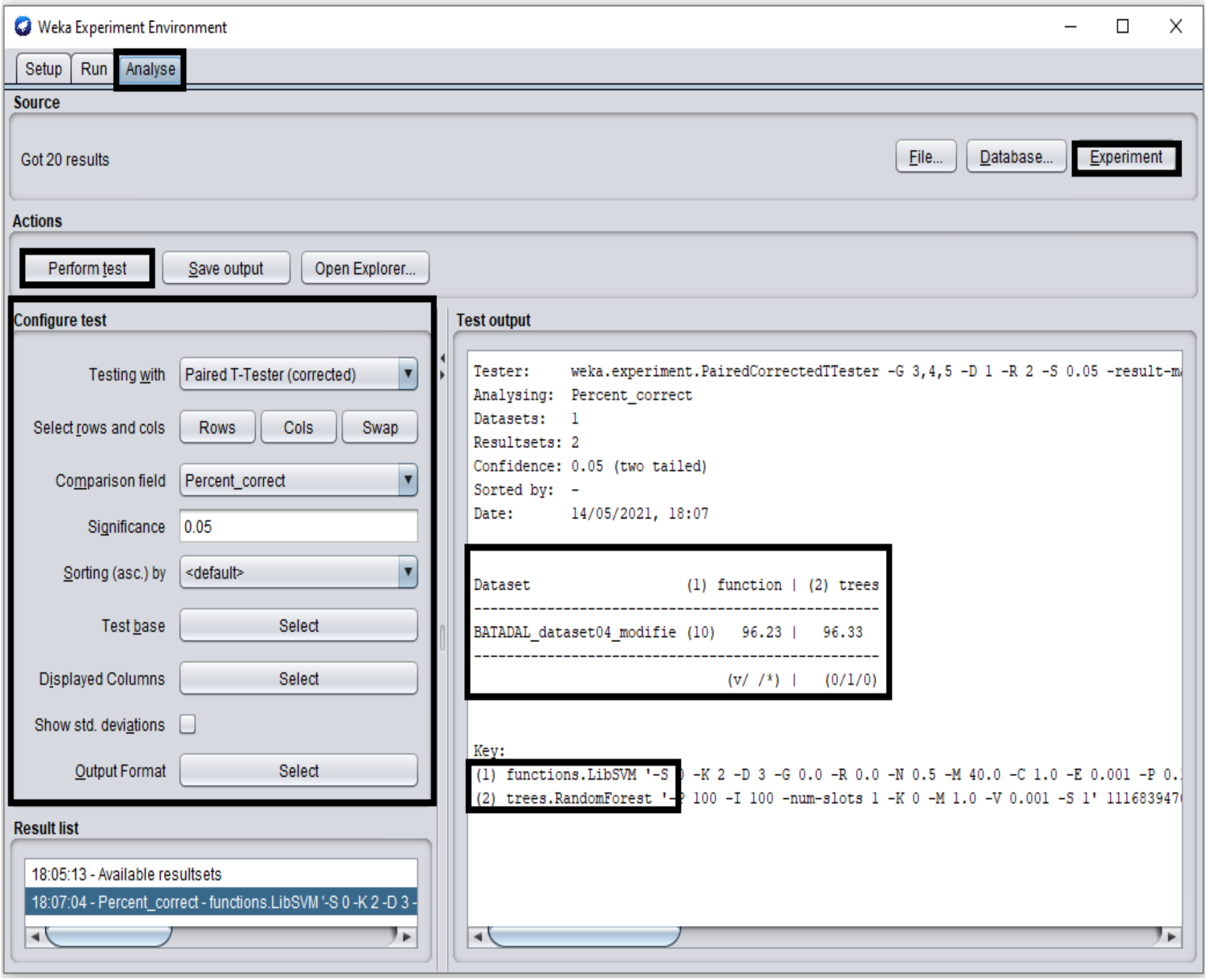
Not running

Figure 9.2: Running the experiment evaluation

Next, we selected the "Analyse" option and the "Experimenter" button. At the "Configure Test," we used the default parameters since they are the most commonly used. Finally, we pressed the "Perform Test" button. Next, we selected the "Analyse" option and the "Experiment" button. At the "Configure Test," we used the default parameters since they are the most commonly used. Finally, we pressed the "Perform Test" button. The "Test Output" gave us the accuracy of the two algorithms.

- (1) functions stay for the LibSVM algorithm.
- (2) trees stay for the Random Forest algorithm.

This analysis shows that the Random Forest accuracy value of 96.3% has outperformed the LibSVM accuracy value of 96.23% even with a small difference.



Got 20 results

Testing with: Paired T-Tester (corrected)

Select rows and cols

Comparison field: Percent_correct

Significance: 0.05

Sorting (asc.) by: <default>

Test base: Select

Displayed Columns: Select

Show std. deviations: □

Output Format: Select

18:05:13 - Available resultsets

18:07:04 - Percent_correct - functions.LibSVM '-S 0 -K 2 -D 3 -

```
Tester:     weka.experiment.PairedCorrectedTTester -G 3,4,5 -D 1 -R 2 -S 0.05 -result-m
Analysing:  Percent_correct
Datasets:   1
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       14/05/2021, 18:07


Dataset                  (1) function | (2) trees
-------------------------------------------------
BATADAL_dataset04_modifie (10)   96.23 |   96.33
-------------------------------------------------
                            (v/ /*) |   (0/1/0)


Key:
(1) functions.LibSVM '-S 0 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.
(2) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 111683947
```

Figure 9.3: Analysing the results

Here we give a comprehensive comparison between the metric values of support vector machine and random forest algorithms.

1. **Model time**: Looking at LibSVM for the "time taken to build the model," it required 0.07 seconds while Random Forest needed 0.02 seconds. LibSVM seems to need much more time, but Random Forest seems to be faster.
2. **Accuracy**: There is a slight difference between the two algorithms in terms of accuracy. Both have a plausible "accuracy," with Random Forest a bit better than LibSVM. LibSVM shows 96.23% of accuracy while Random Forest 96.33%.
3. **True Positive rate**: Both algorithms have a high "true positive rate," almost equal to each other. However, LibSVM seems to be better, with its value of 96.8%, than the value of 96.6%, corresponding to Random Forest.
4. **False Positive rate**: As mentioned before in this thesis, we expect machine learning algorithms to turn a low "false-positive rate." Random Forest seems to have a higher value than LibSVM, showing 66%, and LibSVM shows 61%.
5. **Precision**: In terms of "precision," both algorithms perform equally to each other. Their precision rate is 96.5%.
6. **Recall**: Our last metric is "recall," in which both algorithms show great rates. Support vector machine's recall value is equal to 96.8%, and Random Forest's value is equal to 96.6%.

LibSVM performs slightly better than Random Forest. We visualized this comparison in the below chart.

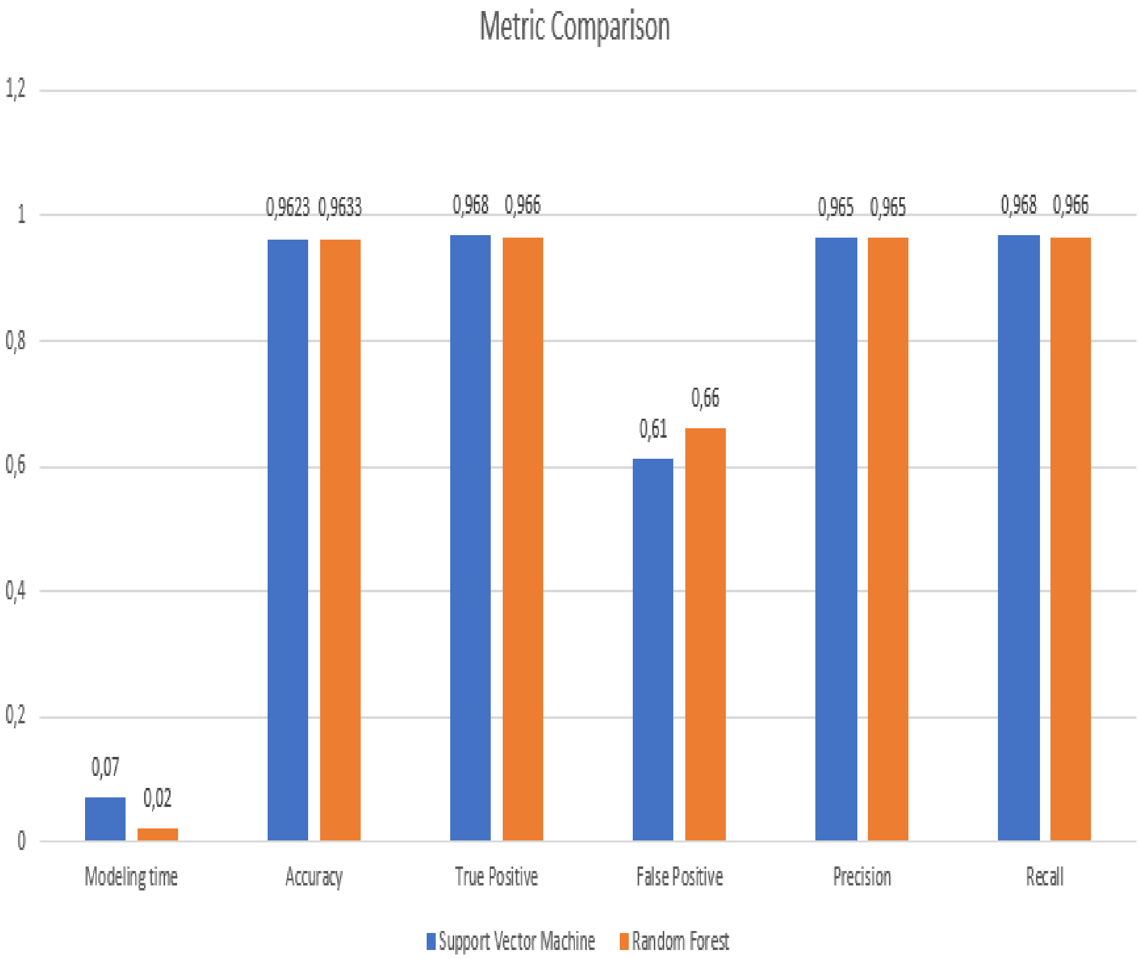


Figure 9.4: Metrics Comparison

At the first view it may look that there is not such a big difference, but when considering the size of the dataset, it will take much more time to train a bigger dataset with SVM than RF. On one side, looking at these values, we can understand that using SVM may take more time to train the model, which means that it may take more time to detect the intrusions, while RF is faster in this aspect. However, on the other side, RF looks to have a higher false-positive rate which means that it turns more false alarms. In a high-level view, both algorithms can be used in intrusion detection activities as a part of the framework we designed. They could be implemented in the correlation engine to detect normal and abnormal traffic.

## 9.1 Validity Evaluation

**Reliability evaluation**

We reached the performance results of the algorithms by repeating the experiment in 10 iterations. Our dataset consist of more than 4000 instances of data. Each iteration takes a considerable time to process all the data when specific algorithms are applied. The number of iterations is to look ahead for to find the optimal values. During the experiment we saw that after repeating it ten times, the values seem to not change too much, for such reason we decided to use ten iterations. we can say for sure that we reached reliable results in the setup that we implemented our experiment because we followed the right machine learning process to train and test the data, based on the machine learning principles. However, we could not say that these algorithms would show the same good performance when used in other datasets. Instead we would say that applying machine learning in IDS affects positively in the system's performance by increasing the prediction capacities for detection and classification.

**Internal Validity**

The design of the experiment is fundamental about internal validity. In machine learning, the algorithms must learn an anomaly and normal data on training data, and later their prediction ability is tested on new data. We followed the same logic process as well.

**External validity**

We can not say that the conclusions from the experiment would be the same if we test them in the real world since we did not apply our experiment in real system data but in simulated data. We are aware that external validity is the hardest to be achieved as well. To improve the external validity, we would require more time to conduct a field experiment. In a field experiment the system's behavior is studied outside the laboratory, in its natural setting using a concrete case study.

# 10 Ethical principles in research

Since our study does not involve any humans subjects directly or indirectly, it has no ethical aspects to report. However, I am stating below some of the ethical aspects purely from a personal point of view. Ethical problems in the computing area include the unauthorized use of hardware, the theft or misuse of the software, disputed right to products, the use of computers to commit fraud, the phenomenon of hacking and data theft, sabotage in the form of viruses, etc. In this thesis, we had to deal with open-source software and dataset. Even though these products are open-source, we still referred to their usage and their authorship to avoid misunderstandings. Concerning the experiment, as a software engineer candidate, I ensure that the results meet the highest professional standards possible when these algorithms are used. I accept full responsibility for work done and for factual and objective information presented in the report.

**Consent**

I am content that I applied voluntarily for this master thesis study in a collaboration between academia, represented by my university supervisor, and industry represented by my company supervisor. I had an excellent collaboration experience, where I learned new knowledge and practices that can be applied in real domains.

**Beneficence**

I am grateful to those who helped me with their advice and encouraged me to do the work properly.

**Confidentiality**

During these six months, Í had access to the office of RISE Västerås. Even though I did not get any confidential information relevant to my thesis, I signed a contract to protect privacy and confidentiality. Using the devices and other resources was relevant only to the research process, and I was careful to respect the data protection principles.

**Scientific value**

This research was conducted to have results for the good of society. I paid attention to avoid random and unnecessary research.

**Researcher skill**

I followed scientific principles and systematic research methods during the thesis, which I learned as part of my research methodology course.

**Respect law**

I respected the policies and rules that were existing both at MDH and RISE .

# 11 Conclusions

As part of industry 4.0, industrial control systems are connected to networks, which makes them more vulnerable to cyber-attacks. Since security is one of the main concerns that affect the system development life cycle, the need for integrating intrusion detection in the maintenance phase is apparent. In this thesis, we did an in-depth investigation on different IDSs, by researching their role, which is relevant and important to maintain security in a system. We defined some of the security requirements that an IDS should fulfill. Next, we investigated the ML algorithms, the categories they are divided into, and did a literature review. The literature review phase provided us the answers to the research questions, which are related to the main goals of our work. From the literature review, we proposed a hybrid intrusion detection framework enriched with machine learning capabilities. We integrated different components, both technical such as signature-based detection technique, behavior-based detection technique, and the human factor, to increase security. One of our main contributions is related to the combined design of the machine learning approach, which could help increase the performance of our proposed framework by identifying and classifying the intrusions and reducing the false alarms more accurately. The output logs of this detection would be very helpful for the security of a system since it would help the developers to define new security countermeasures. It would be easier for the developers to formulate defense strategies, which could prevent the system from a breakdown and expand its functionality for a longer time. In the experimental phase, we measured the performance of three types of ML algorithms, K-means, SVM, and RF. The results showed that these algorithms could be used together to minimize each other's drawbacks. Thereby, we assume that they can be applied in the framework that we designed. This combination can assist industrial control systems in identifying the threats before any damage can happen.

## 12 Discussions and future work

Intrusion detection is essential for the system development life cycle, especially in maintaining security. Intrusion detection activities can catch anomalies that are causing or may cause a system to malfunction. For such reasons, it is necessary to use efficient detection mechanisms. In this thesis, we focused on intrusion detection using machine learning in ICS. From the literature review, we found that hybrid detection systems are in the early stages, and still, there is space for improvement. We designed a theoretical framework that could be applied in the industrial control network. We propose to merge the existing detection mechanisms with machine learning and the human factor. Similar studies have discussed this approach with commonly used datasets such as KDD99, which are old and are not updated. Different from the other researchers, we selected a dataset that corresponds to an industrial control system. We discovered that we could combine a clustering algorithm with a classification algorithm in a standard process to predict the anomalies on the test data. Still, we faced some limitations as well. The first limitation is that we did not have enough time to investigate an actual industrial case study. Instead of using simulation data, we would prefer to explore an existing industrial case study because we assume that we could have done a more specific investigation that would have directly helped that industry. Another limitation was that the scope of the thesis was very wide and not so clear at the early stages, but we managed to centralize our work by focusing in one direction. However, our applied methodology of training and testing the machine learning algorithms can be generalized also in other domains. Lastly, we believe that we have provided a comprehensive work which can serve as a guide for future works. For instance, more profound research can be conducted on an actual industrial case study or to develop an open-source intrusion detection tool such as SNORT with machine learning capabilities.

# References

[1] Amira Sayed A. Aziz and Hanafi. “Comparison of classification techniques applied for network intrusion detection and classification”. In: *Journal of Applied Logic* 24 (2017). SI:SOCO14, pp. 109–118. ISSN: 1570-8683. DOI: https://doi.org/10.1016/j.jal.2016.11.018. URL: https://www.sciencedirect.com/science/article/pii/S1570868316300738.

[2] A. A. Aburomman and M. Bin Ibne Reaz. “Ensemble of binary SVM classifiers based on PCA and LDA feature extraction for intrusion detection”. In: *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*. 2016, pp. 636–640. DOI: 10.1109/IMCEC.2016.7867287.

[3] ICS-CERT Alert. “Cyber-attack against ukrainian critical infrastructure”. In: *Cybersecurity Infrastruct. Secur. Agency, Washington, DC, USA, Tech. Rep. ICS Alert (IR-ALERT-H-16-056-01)* (2016).

[4] S. D. D. Anton and S. Sinha. “Anomaly-based Intrusion Detection in Industrial Data with SVM and Random Forests”. In: *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. 2019, pp. 1–6. DOI: 10.23919/SOFTCOM.2019.8903672.

[5] Parvat Arjunwadkar Narayan. “An Intrusion Detection System,(IDS) with Machine Learning (ML) Model Combining Hybrid Classifiers”. In: *connections* 1 (2015), p. 3.

[6] Muhammad Rizwan Asghar. “Cybersecurity in industrial control systems: Issues, technologies, and challenges”. In: *Computer Networks* 165 (2019), p. 106946.

[7] Raza Ashfaq. “Fuzziness based semi-supervised learning approach for intrusion detection system”. In: *Information Sciences* 378 (2017), pp. 484–497.

[8] K. Atefi and S. Yahya. “Anomaly detection based on profile signature in network using machine learning technique”. In: *2016 IEEE Region 10 Symposium (TENSYMP)*. 2016, pp. 71–76. DOI: 10.1109/TENCONSpring.2016.7519380.

[9] Chandrashekhar Azad. “Fuzzy min–max neural network and particle swarm optimization based intrusion detection system”. In: *Microsystem Technologies* 23.4 (2017), pp. 907–918.

[10] Manjula C. Belavagi and Balachandra Muniyal. “Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection”. In: *Procedia Computer Science* 89 (2016). Twelfth International Conference on Communication Networks, ICCN 2016, August 19–21, 2016, Bangalore, India Twelfth International Conference on Data Mining and Warehousing, ICDMW 2016, August 19-21, 2016, Bangalore, India Twelfth International Conference on Image and Signal Processing, ICISP 2016, August 19-21, 2016, Bangalore, India, pp. 117–123. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2016.06.016. URL: https://www.sciencedirect.com/science/article/pii/S187705091631081X.

[11] Erhan Buczak. “A survey of data mining and machine learning methods for cyber security intrusion detection”. In: *IEEE Communications surveys & tutorials* 18.2 (2015), pp. 1153–1176.

[12] Brian Caswell and Jay Beale. *Snort 2.1 intrusion detection*. Elsevier, 2004.

[13] Md Nasimuzzaman Chowdhury. “Network intrusion detection using machine learning”. In: *Proceedings of the International Conference on Security and Management (SAM)*. The Steering Committee of The World Congress in Computer Science, Computer ... 2016, p. 30.

[14] Marietheres Dietz, Manfred Vielberth, and Günther Pernul. “Integrating digital twin security simulations in the security operations center”. In: *Proceedings of the 15th International Conference on Availability, Reliability and Security*. 2020, pp. 1–9.

[15] Solane Duque and Nizam bin Omar. “Using data mining algorithms for developing a model for intrusion detection system (IDS)”. In: *Procedia Computer Science* 61 (2015), pp. 46–51.

[16] S. Eltanbouly and M. Bashendy. “Machine Learning Techniques for Network Anomaly Detection: A Survey”. In: *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*. 2020, pp. 156–162. DOI: 10.1109/ICIoT48696.2020.9089465.

[17] Nabila Farnaaz. “Random forest modeling for network intrusion detection system”. In: *Procedia Computer Science* 89 (2016), pp. 213–217.

[18] DP Gaikwad and Ravindra C Thool. “Intrusion detection system using bagging with partial decision treebase classifier”. In: *Procedia Computer Science* 49 (2015), pp. 92–98.

[19] Christian Gehrmann and Martin Gunnarsson. “A Digital Twin Based Industrial Automation and Control System Security Architecture”. English. In: *IEEE Transactions on Industrial Informatics* 16.1 (Jan. 2020), pp. 669–680. ISSN: 1941-0050. DOI: 10.1109/TII.2019.2938885.

[20] Mahmudul Hasan and Md. Milon Islam. “Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches”. In: *Internet of Things* 7 (2019), p. 100059. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2019.100059. URL: https://www.sciencedirect.com/science/article/pii/S2542660519300241.

[21] A. Humayed et al. “Cyber-Physical Systems Security—A Survey”. In: *IEEE Internet of Things Journal* 4.6 (2017), pp. 1802–1831. DOI: 10.1109/JIOT.2017.2703172.

[22] Mikel Iturbe. “Towards large-scale, heterogeneous anomaly detection systems in industrial networks: A survey of current trends”. In: *Security and Communication Networks* 2017 (2017).

[23] MA Jabbar, Rajanikanth Aluvalu, et al. “RFAODE: A novel ensemble intrusion detection system”. In: *Procedia computer science* 115 (2017), pp. 226–234.

[24] M. H. Kamarudin and C. Maple. “A LogitBoost-Based Algorithm for Detecting Known and Unknown Web Attacks”. In: *IEEE Access* 5 (2017), pp. 26190–26200. DOI: 10.1109/ACCESS.2017.2766844.

[25] Mohamad Kaouk. “A review of intrusion detection systems for industrial control systems”. In: *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE. 2019, pp. 1699–1704.

[26] Ansam Khraisat. “Survey of intrusion detection systems: techniques, datasets and challenges”. In: *Cybersecurity* 2.1 (2019), pp. 1–22.

[27] Sim Kok et al. “A review of intrusion detection system using machine learning approach”. In: *International Journal of Engineering Research and Technology* 12 (Jan. 2019), pp. 8–15.

[28] Karthikeyan KR. “Intrusion Detection Tools and techniques–a Survey’”. In: *International Journal of Computer Theory and Engineering* 2.6 (2010), pp. 1793–8201.

[29] Maryna Krotofil and Dieter Gollmann. “Industrial control systems security: What is happening?” In: *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*. IEEE. 2013, pp. 670–675.

[30] Y.V. Kumar and K. Kamatchi. “Anomaly Based Network Intrusion Detection Using Ensemble Machine Learning Technique”. en. In: *International Journal of Research in Engineering, Science and Management* 3.4 (Apr. 2020), pp. 290–297.

[31] R. Langner. “Stuxnet: Dissecting a Cyberwarfare Weapon”. In: *IEEE Security Privacy* 9.3 (2011), pp. 49–51. DOI: 10.1109/MSP.2011.67.

[32] Wei-Chao Lin. “CANN: An intrusion detection system based on combining cluster centers and nearest neighbors”. In: *Knowledge-based systems* 78 (2015), pp. 13–21.

[33] Tahir Mehmood. “Machine learning algorithms in context of intrusion detection”. In: *2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*. IEEE. 2016, pp. 369–373.

[34] Hela Mliki. “Intrusion Detection Study and Enhancement Using Machine Learning”. In: *International Conference on Risks and Security of Internet and Systems*. Springer. 2019, pp. 263–278.

[35] Raffie ZA Mohd et al. “Anomaly-based nids: A review of machine learning methods on malware detection”. In: *2016 International Conference on Information and Communication Technology (ICICTM)*. IEEE. 2016, pp. 266–270.

[36] Antonia Nisioti. “From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods”. In: *IEEE Communications Surveys & Tutorials* 20.4 (2018), pp. 3369–3388.

[37] Deyban Perez. “Intrusion detection in computer networks using hybrid machine learning techniques”. In: *2017 XLIII Latin American Computer Conference (CLEI)*. IEEE. 2017, pp. 1–10.

[38] A. Phadke and M. Kulkarni. “A Review of Machine Learning Methodologies for Network Intrusion Detection”. In: *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*. 2019, pp. 272–275. DOI: 10.1109/ICCMC.2019.8819748.

[39] Brandon Phillips, Eric Gamess, and Sri Krishnaprasad. “An evaluation of machine learning-based anomaly detection in a SCADA system using the modbus protocol”. In: *Proceedings of the 2020 ACM Southeast Conference*. 2020, pp. 188–196.

[40] Bernardi Pranggono and Kieran McLaughlin. “Intrusion detection systems for critical infrastructure”. English. In: *The State of the Art in Intrusion Prevention and Detection*. CRC Press, 2014, pp. 150–170. ISBN: 9781482203516.

[41] Smitha Rajagopal, Poornima Panduranga Kundapur, and Katiganere Siddaramappa Hareesha. “A stacking ensemble for network intrusion detection using heterogeneous datasets”. In: *Security and Communication Networks* 2020 (2020).

[42] B Basaveswara Rao and Kailasam Swathi. “Fast kNN classifiers for network intrusion detection system”. In: *Indian Journal of Science and Technology* 10.14 (2017), pp. 1–10.

[43] Paul Roberts. *Cyberattacks inflicts massive damage on German steel factory*. URL: https://securityledger.com/2014/12/cyberattack-inflicts-massive-damage-on-german-steel-factory/. (accessed: February 2021).

[44] Luis Rosa. “Intrusion and anomaly detection for the next-generation of industrial automation and control systems”. In: *Future Generation Computer Systems* 119 (2021), pp. 50–67.

[45] Noräs Salman and Marco Bresch. “Design and implementation of an intrusion detection system (IDS) for in-vehicle networks”. In: *Dept. Comput. Sci. Eng., Chalmers Univ. Technol., Gothenburg, Sweden, Tech. Rep* 1 (2017).

[46] Sridevi Saranya. “Performance analysis of machine learning algorithms in intrusion detection system: A review”. In: *Procedia Computer Science* 171 (2020), pp. 1251–1260.

[47] Amrit Pal Singh. “Analysis of Host-Based and Network-Based Intrusion Detection System”. In: *International Journal of Computer Network and Information Security* 6 (July 2014), pp. 41–47. DOI: 10.5815/ijcnis.2014.08.06.

[48] A. Sultana. “Intelligent network intrusion detection system using data mining techniques”. In: *2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*. 2016, pp. 329–333. DOI: 10.1109/ICATCCT.2016.7912017.

[49] Bose Sundan. “A dynamic intrusion detection system based on multivariate Hotelling’s T2 statistics approach for network environments”. In: *The Scientific World Journal* 2015 (2015).

[50] Riccardo Taormina et al. “Battle of the attack detection algorithms: Disclosing cyber attacks on water distribution networks”. In: *Journal of Water Resources Planning and Management* 144.8 (2018), p. 04018048.

[51] Riccardo Taormina et al. “Characterizing cyber-physical attacks on water distribution systems”. In: *Journal of Water Resources Planning and Management* 143.5 (2017), p. 04017009.

[52] ukdiss.com. *Development of Intrusion Detection System Software*. URL: https://ukdiss.com/examples/the-internet-and-worldwide-connectivity.php?vref=1. (accessed: 13 May 2021).

[53] Kalyan Veeramachaneni et al. “$AI^2 : TrainingaBigDataMachinetoDefend$”. In: *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*. 2016, pp. 49–54. DOI: 10.1109/BigDataSecurity-HPSC-IDS.2016.79.

[54] Marzia Zaman. “Evaluation of machine learning techniques for network intrusion detection”. In: *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE. 2018, pp. 1–5.