mathematics

Article

Hydraulic Data Preprocessing for Machine Learning-Based Intrusion Detection in Cyber-Physical Systems

Ignitious V. Mboweni [1,*], Daniel T. Ramotsoela [1] and Adnan M. Abu-Mahfouz [2]

1 Department of Electrical Engineering, University of Cape Town, Cape Town 7700, South Africa
2 Council for Scientific and Industrial Research, Pretoria 0184, South Africa
* Correspondence: mbwign002@myuct.ac.za

Abstract: The protection of critical infrastructure such as water treatment and water distribution systems is crucial for a functioning economy. The use of cyber-physical systems in these systems presents numerous vulnerabilities to attackers. To enhance security, intrusion detection systems play a crucial role in limiting damage from successful attacks. Machine learning can enhance security by analysing data patterns, but several attributes of the data can negatively impact the performance of the machine learning model. Data in critical water system infrastructure can be difficult to work with due to their complexity, variability, irregularities, and sensitivity. The data involve various measurements and can vary over time due to changes in environmental conditions and operational changes. Irregular patterns and small changes can have significant impacts on analysis and decision making, requiring effective data preprocessing techniques to handle the complexities and ensure accurate analysis. This paper explores data preprocessing techniques using a water treatment system dataset as a case study and provides preprocessing techniques specific to processing data in industrial control to yield a more informative dataset. The results showed significant improvement in accuracy, F1 score, and time to detection when using the preprocessed dataset.

Keywords: critical infrastructure; critical water system infrastructure; cyber-physical systems; data preprocessing; industrial control; intrusion detection systems; machine learning; water treatment system

MSC: 68T07

check for updates

Citation: Mboweni, I.V.; Ramotsoela, D.T.; Abu-Mahfouz, A.M. Hydraulic Data Preprocessing for Machine Learning-Based Intrusion Detection in Cyber-Physical Systems. *Mathematics* **2023**, *11*, 1846. https://doi.org/10.3390/math11081846

Academic Editor: Huawen Liu

Received: 11 February 2023
Revised: 29 March 2023
Accepted: 6 April 2023
Published: 13 April 2023

1. Introduction

Water treatment systems are critical infrastructure elements that play an important role in public health, wellbeing, productivity, and functionality. By providing access to safe and clean drinking water, these systems help to ensure that communities are able to lead healthy, productive, and functional lives. Due to their importance, they need to be protected using appropriate security measures [1,2]. Cyber-physical systems (CPSs) combined with communication schemes enhance protection by allowing computerized monitoring and control of physical components. The use of these advanced technologies is crucial for the development and growth of "smart cities" in the future.

However, CPS communication schemes introduce vulnerabilities that attackers can exploit for malicious purposes. In the past, security was based on the belief that systems were isolated from one another and that security was maintained by monitoring and controlling them locally [3]. The use of off-the-shelf software and hardware in supervisory control and data acquisition (SCADA) systems opens up opportunities for threats due to the inherent vulnerabilities present in legacy control systems and their communication channels [3,4].

Attacks emanate in CPSs as intrusions, which if successful can compromise the system's integrity, confidentiality, or availability [5]. Preventative security measures do exist, but they can be defeated; this creates a need for reactive mechanisms that can assist in the recovery phase, a crucial stage in addressing the consequences of a successful attack and ensuring that the system returns to a secure state [6]. Anomaly detection (AD) is a popular behavioural intrusion detection technique that classifies system behaviour as normal or anomalous by analysing data. This technique allows for the detection of both known and unknown attacks by creating a profile of normal behaviour for the system and flagging deviations as anomalous and potentially malicious. Figure 1 provides a visual illustration of how machine learning (ML) fits together with CPSs and critical water system infrastructure.

One important part of machine learning modelling is data preprocessing, which speeds up calculations and results in more accurate models [7]. Sensor and actuator data from industrial control systems (ICS) can be complex, noisy, and nonlinear; therefore, advanced processing techniques are necessary. This study proposes the use of machine learning to detect attacks in critical water system infrastructure, with a specific focus on the processing of hydraulic component data, such as sensors that measure parameters such as pressure, flow rate, temperature, and position and actuators such as pumps, valves, and cylinders. The goal is to formulate ideal preprocessing techniques to improve machine learning model performance. This is a crucial aspect of ML modelling that is often neglected in the literature. This will enhance cybersecurity research in cyber-physical systems by offering a data preprocessing workflow and techniques that can be applied to similar data gathered from a similar environment to that used in the study of water treatment.

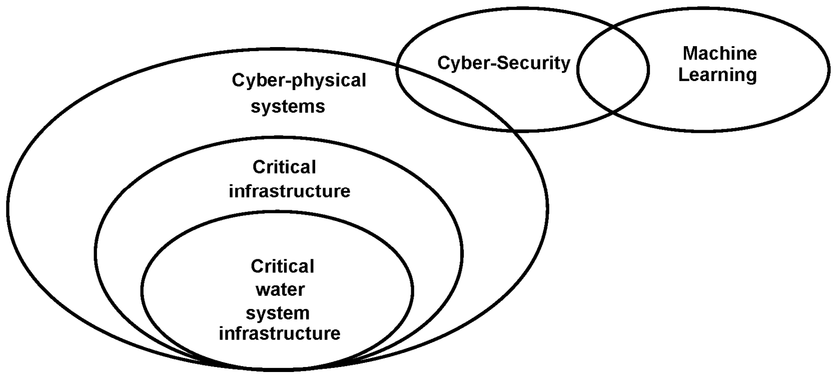


Figure 1. Illustration of connectedness between machine learning and cybersecurity in critical water system infrastructure [8].

The rest of this paper is organized as follows: The first section of the paper, titled 'Materials and Methods', describes the data used in this study and outlines the methodology applied. It includes an examination of the data preprocessing techniques that were investigated and the experiments performed. The following section, 'Experiments and Results', presents the results of the experiments conducted. The results are then discussed in depth in the next section, 'Discussion of Results', followed by a conclusion that summarizes the key findings and contributions of the study.

2. Literature Review

Data preprocessing is a crucial step in machine learning and plays a significant role in the success of intrusion detection systems in CPSs. In this literature review, we focus on the data preprocessing aspect of machine learning in intrusion detection systems for critical water system infrastructure.

Data preprocessing involves cleaning, transforming, and preparing data for analysis. One common preprocessing step is feature selection, which involves choosing the most important variables that have a significant impact on the outcome. Feature selection has proven to be a good strategy through its effectiveness in reducing overfitting. This method helps in preparing clean and comprehensible data [9]. The review provided by Fan et al. [10] outlines how feature selection techniques can be divided into three categories, which are filter, wrapper, and embedded methods. In a study by Ullah et al. [11], feature selection was performed using a genetic algorithm instead of traditional methods such as filter or wrapper methods. This optimized method was found to be less time-consuming and improved prediction accuracy compared to traditional approaches.

Another important preprocessing step is data normalization, which involves transforming the data into a standard format. This is important in intrusion detection systems as it reduces the impact of differences in measurement units and scales on the results. For example, in a study by Ashouri et al. [12], data normalization was performed using the min–max normalization method to improve building energy performance. Nawi et al. [13] investigated improving the efficacy of a multilayer perceptron (MLP) model, an artificial neural network (ANN)-based algorithm, where they evaluated three normalization techniques—min–max, z-score and decimal scaling—which improved the computational efficiency of an ANN algorithm. Halimaa A. and Sundarakantham [14] performed -based intrusion detection and highlighted the importance of preprocessing the dataset to remove or replace non-numeric or symbolic features before using naïve Bayes and SVM classifiers.

Feature extraction is also an important preprocessing step because for an ML model to learn from a signal, the raw data need to be transformed into a set of informative features. Identification of important features for a specific problem, description of those characteristics, and implementation of a method to extract those features are all necessary for manual feature extraction. In the context of water treatment systems, feature extraction can involve the extraction of relevant timestamp, statistical, and temporal and spectral features associated with physical and chemical parameters. In a study by Liu et al. [15], various techniques were used to extract time and frequency features, which were used to train a support vector machine (SVM) as a classifier to perform leakage detection; the proposed method was found to be effective based on simulation and experimental results.

Handling missing values is another important preprocessing step. Missing values can occur due to various reasons, such as sensor failures or network disruptions. In intrusion detection systems, missing values can impact the accuracy of the results and lead to false alarms. To address this, various imputation methods have been used, such as mean imputation, median imputation, and linear interpolation. For example, in a study by Bijlsma et al. [16], a large-scale human metabolomics study was performed on 600 plasma samples to detect variances in metabolic profiles when associated with a large biological variation. As part of the preprocessing, they had to deal with missing values that were caused by missing peak values during the peak picking process.

Noise reduction also plays a big role when dealing with hydraulic component data. Noise is meaningless data, which corrupt the underlying information in a dataset, and since data collected from the real world using measurement tools is hardly perfect, many times data need to be processed to remove or reduce the noise components. If they are not, there is a possibility of reduced system performance, such as classification inaccuracy and increased time and resources required to train the classifier [17]. In a study by Kang et al. [18], they applied a k-nearest neighbours (k-NN)-based filter before resampling data as a new under-sampling scheme proposed to correct class imbalance problems caused by noisy minority examples, which result in a decreased performance of a classifier. The proposed scheme exhibited an improvement in all four under-sampling methods it was implemented on, namely AdaBoost, RUSBoost, UnderBagging, and EasyEnsemble.

Outliers can have a significant impact on the accuracy of ML models, so it is important to detect and remove them. Common techniques for outlier filtering include statistical methods and machine learning algorithms. Li et al. [19] hypothesized that classification accuracy can be improved if an algorithm is trained on data where outliers have been removed. They looked at data of multispectral imaging (MSI) implemented on burn tissue. They developed an outlier detection and removal method based on Z-test, a statistical hypothesis test used to check that the means of two populations are different when the variances are known and the sample size is large. They found that the accuracy was improved from 63% to 76%, a good accuracy that is equivalent to what burn surgeons achieve when using clinical judgement.

Finally, we have dimensionality reduction, which is the process of reducing the number of features in the data. This is important because high-dimensional data can lead to overfitting, increased computation time, and decreased accuracy. Lam et al. [20] investigated the use of electricity in office buildings and its relationship with weather patterns. The study used principal component analysis (PCA) to analyse five major climatic conditions, namely dry-bulb temperature, wet-bulb temperature, global solar radiation, clearness index, and wind speed. Thereafter, they could use regression models to correlate the revealed components to electricity use, thus proving PCA to be a good dimensionality reduction method.

In conclusion, data preprocessing is a crucial step in machine learning for intrusion detection systems in critical water system infrastructure. It involves cleaning, transforming, and preparing the data for analysis. Appropriate data preprocessing can improve the accuracy of the machine learning algorithms and reduce false alarms.

3. Materials and Methods

3.1. Data

Secure Water Treatment (SWaT) is a water treatment testbed for cybersecurity research. It consists of a scaled-down six-stage water treatment process that is almost indistinguishable from a real-world treatment plant [21]. It became operational in 2015, and the dataset used in this study was collected over a 11-day period where 7 days were of normal operation and 4 days of operation had attack scenarios introduced. During the data collection process, all network traffic, sensor, and actuator data were collected. The work presented in this study looks only at sensor and actuator data, a time series dataset consisting of 51 features in the form of continuous waves sampled at a rate of 1 Hz. Stage P1 of the physical process begins by taking in raw water, followed by chemical dosing (Stage P2), filtering through an ultrafiltration (UF) system (Stage 3), dechlorination using UV lamps (Stage P4), and then feeding it to a reverse osmosis (RO) system (Stage P5). A backwash process (Stage P6) cleans the membranes in UF using the RO permeate. The six sub-processes, referred to as stages P1 to P6, are controlled by a set of dual Allen-Bradley PLCs, a primary and a redundant hot standby. The operation status of the PLCs is monitored by a SCADA system [21]. Data were collected from an empty state; it took 5 h for SWaT to stabilise while taking an extra hour to completely fill up the tanks in stages 3 and 4. This is to be accounted for in the modelling process of the ML algorithm.

A total of 36 physical attacks were introduced. The duration and impact of each attack varied, with some taking longer to manifest and affect the system dynamics. The attack types are described as follows:

1. Single stage single point (SSSP): The focus of this type of attack is a single point within a CPS. The dataset contains 26 instances of this attack.
2. Single stage multi-point (SSMP): This type of attack targets multiple points within a CPS, but only during a single stage. The set P in this case includes more than one element selected from any stage in the CPS. There are 4 instances of this attack in the dataset.
3. Multi-stage single point (MSSP): This attack type is similar to an SSSP attack, but it targets multiple stages in the CPS. There are 2 instances of this attack in the dataset.
4. Multi-stage multi-point (MSMP): This attack type is an SSMP attack implemented at two or more stages of the CPS. There are 4 of these attacks in the dataset.

3.2. Modelling Approach

When designing a machine learning model, a model workflow is followed. This creates a clear structure of the modelling process and provides preliminary insights before modelling. The model workflow is shown in Figure 2. Since the work conducted in this study focuses to a great extent on data preprocessing and not the entire ML modelling process, part 4 is the focus of the study and is discussed in great detail; in support of this, parts 1 to 3 are also discussed.

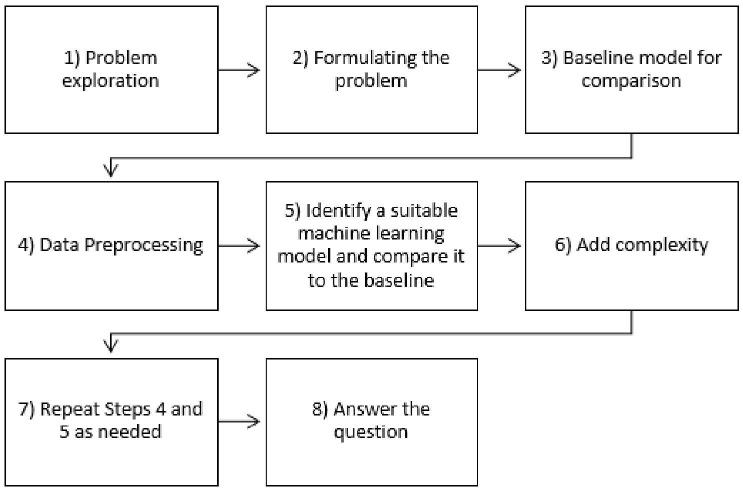

1) Problem exploration
2) Formulating the problem
3) Baseline model for comparison
4) Data Preprocessing
5) Identify a suitable machine learning model and compare it to the baseline
6) Add complexity
7) Repeat Steps 4 and 5 as needed
8) Answer the question

Figure 2. ML modelling workflow.

3.3. Problem Exploration

Problem exploration entails using numerical and visual exploration to better understand the problem and how to solve it. When performing exploration, some of the things that can be inspected are, but not limited to, data types, outliers, distributions, variance imbalances, histograms, and plots. This step is performed before formulating the problem to provide enough context to make an informed decision.

For this study, first the attacks in the sensor and actuator data were visualised. Visualisation of the attacks allows the 4 types of attacks to be viewed against the data and prepare the analyst to apply discretion in how to preprocess the variables. By doing so, visualisation provides contextual information on the types of attacks present and their impact on the data.

3.3.1. Visualising Attacks against Sensor and Actuator Data

Visualising the nature of the attacks in the data can provide valuable insight into the type of attack that is occurring, the frequency and severity of the attack, and any trends or patterns that may be present in the data. This information can be used to improve the design of the anomaly detection model and to respond to the attack more effectively.

In the SWaT dataset, one attack in particular occurred on the 28 December 2015 between times 11:22:00 and 11:28:22; it was an SSSP attack on level transmitter sensor LIT101, which measures the raw water supply tank level and storage stage (P1). The attack increased the water level by 1 mm every second to overflow the tank and also damage pump P101. The attack is visualised in Figure 3.

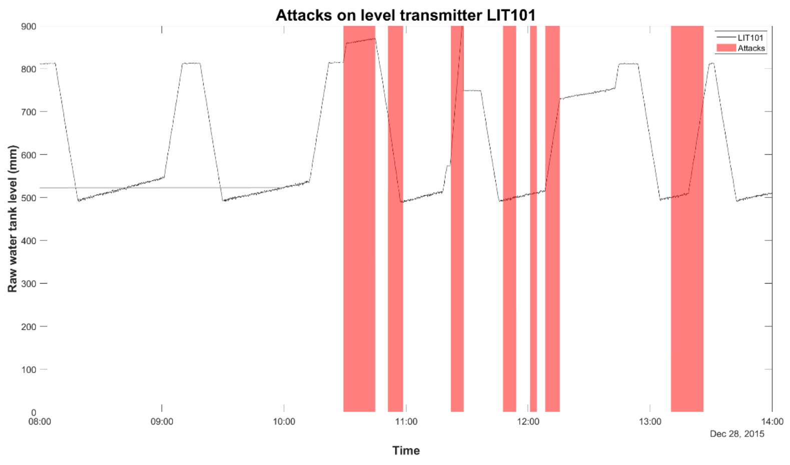


Figure 3. Attacks on level transmitter.

Observing attacks shows deviation from normal data trajectory, providing an understanding of attack impact on variables in the dataset. For example, a short-lived and unusual spike occurred before 11:30. This insight informs the selection of appropriate data preprocessing techniques.

3.3.2. Redundant Variables

Redundant variables in machine learning are features or variables that provide no additional information or predictive power to the model beyond what is already provided by other variables. These variables can be removed from the model without affecting its performance, as they are not contributing to the overall accuracy of the model. Removing redundant variables can also help reduce overfitting and improve the interpretability of the model.

The pumps used in SWaT go in pairs; one is the main pump and the other is a backup pump. The backup pump either pumps alongside the main pump or pumps when the main pump fails. This purports that they can be considered as a single pump and their values merged. Since the pump data is logical, with a value of 0 indicating that the pump is off and a value of 1 indicating that it is on, to merge data from 2 pumps an inclusive OR operation is used where the output is 1 when at least one of the values is 1 and 0 if both values are 0. The resultant variable of the merged pump data takes on the name of the pump whose name has the least numeric value, i.e., pump P101 merged with pump P102 is a new variable called P101.

3.3.3. Dataset Reduction by Correlation Analysis

Correlation analysis is a study to determine whether there is a linear relationship between variables. This method was used in the study to identify closely related variables in the dataset, further investigate the relationship, and drop variables from the dataset should they be directly correlated to others. This exercise ensures that processing techniques are only applied to a reduced dataset with non-redundant variables. Correlation matrices of plots were generated in MATLAB for this task; histograms of the variables appear along the matrix diagonal and scatter plots of variable pairs appear in the off diagonal. The slopes of the least-squares reference lines in the scatter plots are equal to the displayed correlation coefficients; high correlation is indicated by Pearson correlation coefficient values less than -0.8 and greater than 0.8.

1. Stage P1: Raw water supply and storage

   The variables explored were FIT101, LIT101, MV101, and P101. The following observations were made:

   - FIT101 measuring flow into the raw water tank was highly correlated to MV101, which controls water flow to the raw water tank at a correlation coefficient of 0.97.
   - LIT101 measuring the raw water tank level and MV101 share a negative correlation since they share an inversely proportional relationship.
   - LIT101 and P101, which pumps water from the raw water tank, share a negative correlation since they share an inversely proportional relationship.

   Consequently, MV101 is redundant to FIT101 and thus can be dropped, since it is actuator data and not many explanatory data can be extracted from it.

2. Stage P2: Chemical dosing

   The variables explored were AIT201, AIT202, AIT204, FIT201, MV201, P201, P203, and P205. The following observations were made:

   - AIT201, measuring water NaCl content level, and AIT202, measuring water HCl content level, have high correlation but were within acceptable limits at a correlation coefficient of -0.76.
   - MV201, an actuator that controls water flow to the UF feed water tank, was highly correlated to FIT201, a sensor that measures water flow, at a correlation coefficient of 0.91.

- P203, a NaCl-dosing pump, was highly correlated to MV201, at a correlation coefficient of 0.98.
- P203 and FIT201 were perfectly correlated at a correlation coefficient of 1.
- P205, a NaOCl-dosing pump, was strongly correlated to 3 features, FIT201, MV201, and P203, at correlation coefficients of 0.91, 0.9, and 0.91, respectively.

FIT201′s explanation of the events in stage P2 is tantamount to variables MV201, P203, and P205, and due to this redundancy only FIT201 was retained.

3. Stage P3: Ultrafiltration

   The variables explored were DPIT301, FIT301, LIT301, MV301, MV302, MV303, MV304, and P302. The following observation was made:

   - FIT301, a flow sensor measuring the flow of RO rejection, was highly correlated to three features at correlation coefficients of 0.96, 0.9, and 0.98 respectively, namely (1) DPIT301, a sensor that transmits differential pressure to control the backwash process, (2) MV302, a valve actuator that controls the UF backwash drain, and (3) P302, a pump actuator that pumps water from the UF feed water tank to the RO feed water tank via UF filtration.

   FIT301′s explanation of the events in stage P3 is tantamount to variables DPIT301, MV302, and P302, and due to this redundancy only FIT301 was retained; this variable can be processed to extract informative features.

4. Stage P4: Dechlorination

   The variables explored were AIT401, AIT402, FIT401, LIT401, P402, P403, and UV401. The following observations were made:

   - Four variables were closely correlated, however, considering that they form part of the dechlorination process, they were not close enough to drop any of them. They were (1) AIT402, a sensor that measures ORP to control NaHSO3 dosing by pump P204 and NaOCl dosing by pump P205, (2) FIT401, a sensor that measures the flow in order to control the UV dechlorinator, (3) P402, a pump actuator that pumps water from the RO feed tank to the UV dechlorinator, and (4) UV401, a dechlorinator actuator that removes chlorine from the water.
   - FIT401 is responsible for P402 and UV401, which are closely correlated because one controls the other.

   P402 and UV401 were dropped since they were redundant to FIT401.

5. Stage P5: Reverse osmosis

   The variables explored were AIT501, AIT502, AIT503, AIT504, FIT501, FIT502, FIT503, FIT504, P501, PIT501, PIT502, and PIT503. The following observation was made:

   - Seven variables were perfectly correlated at a correlation coefficient of 0.99 between each other and can all be explained by just one of the variables. These variables were (1) FIT501, a sensor measuring the inlet flow of the RO membrane, (2) FIT502, a sensor measuring the RO permeate flow, (3) FIT503, a sensor measuring the RO reject flow, (4) FIT504, a sensor measuring the RO re-circulation flow, (5) P501, a pump actuator that pumps dechlorinated water to RO, (6) PIT501, a sensor measuring the RO feed pressure, and PIT503, a sensor measuring the RO permeate pressure.

   Upon observation of the abovementioned variables, it can be hypothecated that the simple data of P501 can best describe the events of stage P5 in place of variables FIT501, FIT502, FIT503, FIT504, PIT501, and PIT503, which were dropped from the dataset.

6. Stage P6: RO permeate transfer, UF backwash

   In the raw water supply and storage stage (P1), only pump P602, which pumps water from the UF backwash tank to UF, was used in the data collection process. The other variable, pump P601, which pumps water from the RO permeate tank to the raw water tank, was not used and therefore could be dropped from the dataset. A correlation matrix plot was not performed since there was only one variable in this stage.

3.4. Problem Formulation

This section aims to answer the questions: What are you trying to achieve? (1) Is it a regression, clustering, or classification problem? (2) Are you seeking any specific insights? (3) Is a pattern expected? (4) Is there a specific format for the response? Using information gathered from the problem exploration stage, these questions can be answered.

In this case, we have a classification problem, as the dataset includes attacks that need to be identified. Supervised learning is a commonly used approach for anomaly detection in water treatment systems data. The use of supervised learning is driven by several factors, including the availability of labelled data, interpretability, high accuracy, fast detection, and scalability. The advantages and considerations of using supervised learning in this context are:

a. The success of anomaly detection models depends on the availability of labelled data with normal and abnormal behaviour, as this allows for the training of supervised learning models to differentiate between the two [22].
b. Supervised learning models are easy to understand, making it easy to determine the cause of anomalies. This interpretability is important in real-world applications, where understanding the source of anomalies is crucial.
c. Supervised learning models have been shown to be accurate in anomaly detection, especially when the data are well-labelled and the model is designed and optimized properly [23]. Additionally, these models can detect anomalies in real time, making them suitable for online monitoring and control applications [24].
d. Supervised learning models are capable of handling large amounts of data and can be scaled to suit large-scale water treatment systems. This scalability is necessary to ensure that the anomaly detection system can continue to function effectively as the water treatment system grows [25].

3.5. Baseline Models

A baseline model is a simple model that will act as a reference for the machine learning model. To this end, we chose two popular models for classification problems, Fine Decision Tree and Boosted Trees Ensemble. These models are commonly used for classification problems and have been seen to be effective in a variety of contexts. Moreover, we selected these models because they are well-suited for the type of data we are working with, based on their known performance in similar applications.

a. Model 1: Fine Decision Tree

Decision trees are popular in classification problems due to their simplicity and interpretability [26]. They work by recursively splitting the data into smaller subsets, with each split corresponding to a decision rule that best separates the target class [27].

b. Model 2: Boosted Trees Ensemble

This model creates an ensemble of medium decision trees using the AdaBoost algorithm [28]. Ensemble learners, which combine multiple base models, have been shown to perform relatively well as compared to a single algorithm. This is because the combination of multiple base models reduces the variance of the overall prediction, leading to better generalisation performance [29].

A second baseline model was performed to verify the experiment results. The baseline models had no added complexity in the modelling process and served as a benchmark for comparison, with the same models trained on preprocessed data. The modelling approach for training and evaluating ML models is depicted in Figure 4 and describes how the data were first split into training and testing data, followed by the application of the models and then their evaluation on test accuracy; F1 score, which is calculated from precision and recall; and the time it takes to detect an attack, referred to as the TTD.

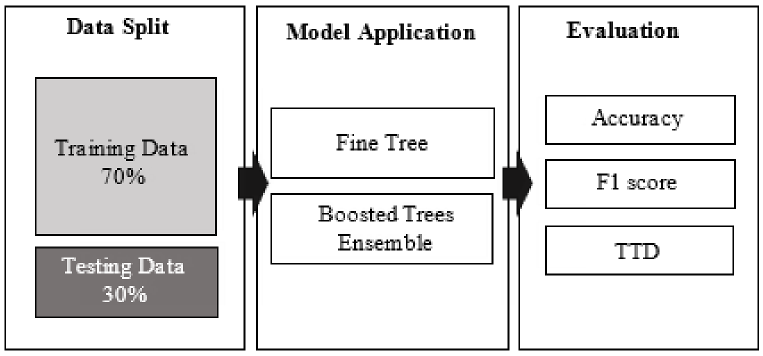


Figure 4. Modelling approach for training and evaluating ML models.

Attacks in the test data are shown in Figure 5 and the time to detection (TTD) is evaluated on the attack that happened between the 31 December 2015 at 01:17:08 and the 31 December 2015 at 11:15:27. TTD involves the calculation of the duration between the initiation of a security breach and the moment when the breach was detected. Specifically, TTD is determined by quantifying the temporal discrepancy between the onset of the attack and the point at which the attack was identified.

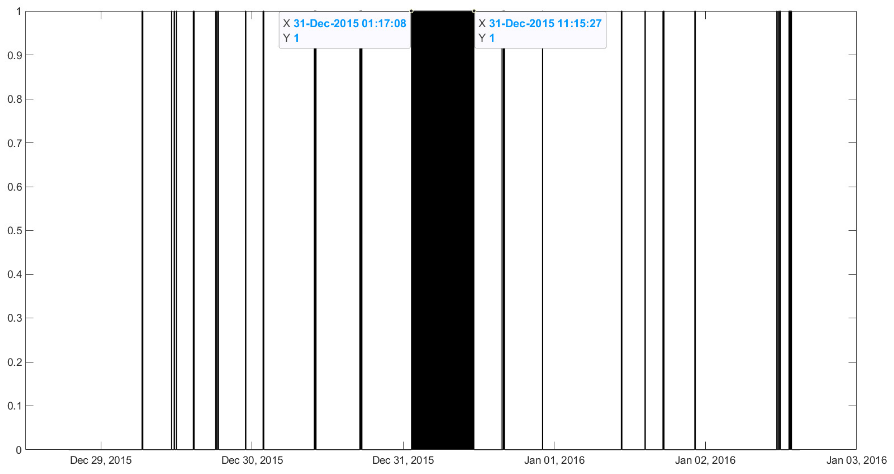


Figure 5. Test data attacks.

The parameters applied when training the models are given in Table 1. These model parameters were carried over to the experiments to maintain consistency and have fair results. The choice of the parameters was based on the scholarship from the literature review.

Table 1. Model parameters.

Training data	70% of dataset
Testing data	30% of dataset
Validation scheme	Cross validation with 2 folds
Misclassification costs	TP = 0, FN = 5, FP = 1, TN = 0

3.6. Data Preprocessing

Here, a systematic method was employed to preprocess the variables that remained after reducing correlations, where each variable was individually examined and the best technique was applied. Many of the variables have distinct features, which allowed for their individual assessment and analysis. The steps for data preprocessing are illustrated in Figure 6, and common techniques for each step are listed in bullet form. These steps were not applied to every variable, as each variable has its own unique characteristics, requiring careful consideration.

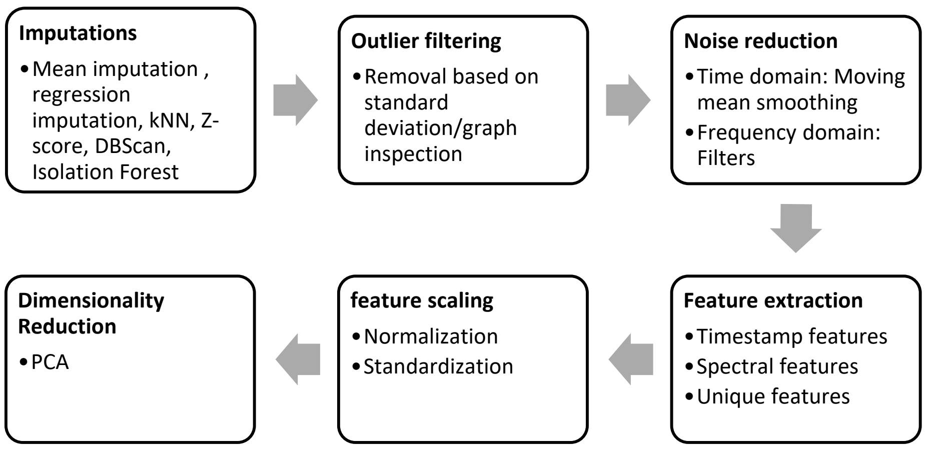

Imputations
• Mean imputation , regression imputation, kNN, Z-score, DBScan, Isolation Forest
Outlier filtering
• Removal based on standard deviation/graph inspection
Noise reduction
• Time domain: Moving mean smoothing
• Frequency domain: Filters
Feature extraction
• Timestamp features
• Spectral features
• Unique features
feature scaling
• Normalization
• Standardization
Dimensionality Reduction
• PCA

Figure 6. Data processing steps.

3.6.1. Imputations

Missing values can be found programmatically using different platforms; in MATLAB, the function *ismissing*(A) returns a logical array that indexes the positions in the input data A that contain missing values. Missing values can be in the form of not-a-number (NaN) values, not-a-datetime (NaT), an undefined value, or an empty space, and discretion is applied to determine what constitutes missing values in that dataset. In dealing with missing values, imputations can be made using various methods, including mean imputation, regression imputation, k-NN based imputation, Z-Score, DBScan, or Isolation Forest to handle such outliers before performing imputation. By taking these steps, we could ensure accurate and reliable imputations for our data. As a finding, the dataset used in this study has no missing values, and thus no missing value imputations were made.

3.6.2. Outlier Filtering

To observe outliers in a variable, it must be visualised either by plotting the data points against time to observe values that deviate from the general pattern of the signal or in a histogram to observe values that are more distant from the mean. Outliers are removed programmatically by removing values that fall within a specified range.

3.6.3. Noise Reduction

Linear filters are an effective way to remove/reduce noise. Filters used in this study were a median filter, a 1-dimensional digital filter, and a Gaussian-weighted moving average filter [30]. The choice of a filter requires that the variable be visualised and the type of filter selected based on the desired result.

a. Median filter smoothing

Median filtering is a nonlinear signal-processing technique that is very useful in data processing to suppress noise. The centre value in a sliding window is replaced by the median of the values in the window, given a discrete sequence $a1, a2, \ldots, a_N$ where N is odd, the median is the number in the sequence where $(N-1)/2$ elements are smaller or equal in value, and

$(N-1)/2$ elements are larger or equal in value [31].

In MATLAB, the built-in function *medfilt1* is used to perform this task. The function applies an nth order one-dimensional median filter to the input vector. The order of the filter impacts its performance, and an iterative process is used to determine the optimal order of the filter.

b. Moving mean smoothing

This is a widely used technique that can be applied to time series data of any type. A sliding window is passed through the time series data to calculate the average values in the new series. The formula for a simple moving average is

$$\bar{y}_t = \frac{y_t + y_{t-1} + \cdots + y_{t-n-1}}{n} \tag{1}$$

where y is the variable, t is the time period, and n is the number of time periods in the averaging window. The MATLAB code for this task uses the *smoothdata* built-in function, which is set to use a moving mean technique.

c. Digital filter

A 1-dimensional digital filter was utilised, using a rational transfer function defined by the numerator and denominator coefficients b and a [32]. The rational transfer function for the filter operates the input–output vectors in the Z-transform and is as follows:

$$Y_{(z)} = \frac{b(1) + b(2)z^{-1} + \cdots + b(n_b+1)z^{-n_b}}{1 + a(2)y(n-1) - \cdots - a(n_a+1)y(n-n_a)} X(z) \tag{2}$$

The rational transfer function handles both finite impulse response (FIR) and infinite impulse response (IIR) filters. Its representation using its direct-form II transposed implementation is shown in Figure 7, where *na* = *nb*.

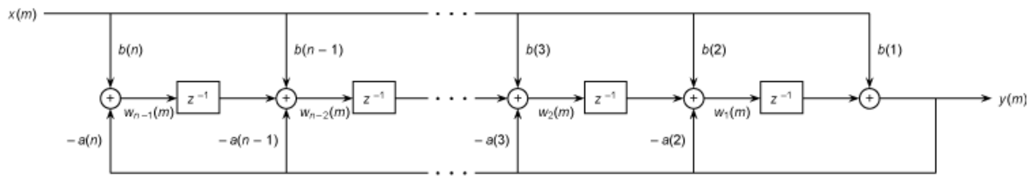


Figure 7. Representation of rational transfer function using its direct-form II transposed implementation [33].

In MATLAB, the built-in function *filter* is used to apply the filter to data.

a. Gaussian-weighted moving average filter

A Gaussian filter is a filter whose impulse response is a Gaussian function; its impulse response is given by

$$g(x) = \sqrt{\frac{a}{\pi}} e^{-ax^2} \tag{3}$$

In MATLAB, the built-in function *smoothdata* is used to apply this filter.

3.6.4. Feature Extraction

a. Timestamp Features

Certain events may be described by the time they occur, and thus timestamp features are extracted from the date/time feature in the dataset [34,35]. Timestamp features can be extracted in MATLAB using built-in functions. In this study, the hour numbers of the date/time and the number representing the day of the week were extracted. The built-in functions for hour and weekday were used to perform this task.

b. Spectral features

To extract frequencies within a time window, an FFT was computed from the data points. The *fft* function in MATLAB computes a discrete Fourier transform (DFT) of the input data using an FFT algorithm. The maximum value of the output represents the fundamental frequency, and this is the value that was populated in the new data set for the computed data window.

A Fourier transform can be applied to any type of signal, not just sinusoidal signals [15,36]. The Fourier transform is a mathematical tool that can be used to decompose a signal into its constituent frequency components. This means that it can be used to analyse any signal, regardless of whether it is sinusoidal or not. It is important to note that when a signal is non-sinusoidal, the output of the Fourier transform will be a complex number, so it will give the amplitude, phase, and frequency information of the signal.

The signal in Figure 8 consists of a noisy sinusoidal structure; it was processed by smoothing it first using moving mean smoothing and then extracting its FFT. Figure 9 is a power spectrum computed for the signal at samples 279,053 to 279,729 (676 data-point window), which shows the signal at a 0.078 Hz normalized frequency.

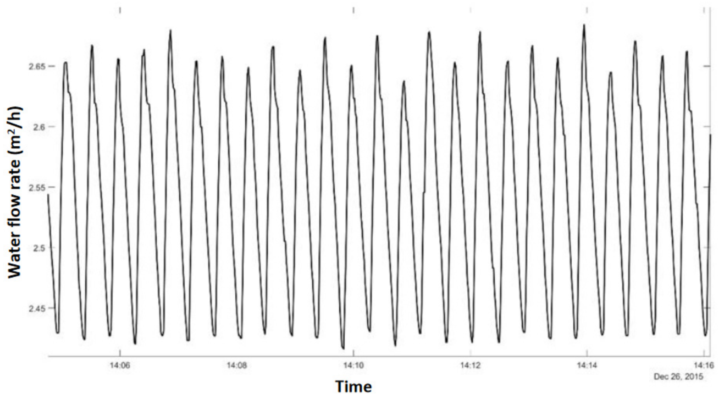


Figure 8. Noisy sensor signal consisting of a sinusoidal structure.

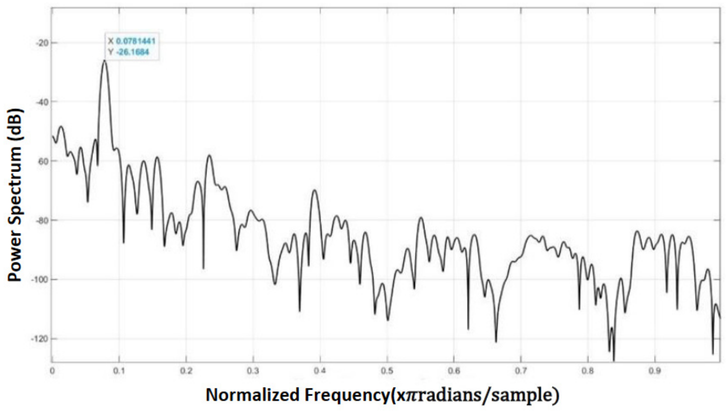


Figure 9. Power spectrum computed from a smoothed sensor signal window.

c. Unique features

Understanding the physical knowledge of the system aids in establishing distinguishing characteristics of the variables in the dataset. The system from which the data were collected must be understood such that when the variables are visualised and studied there is an idea of what distinguishing features can be extracted from the raw data [37]. Features can be a measure of shape such as the slopes or the number and values of local minima/maxima, statistical measurements such as the mean over a moving window, or other measurements that can be thought of upon visualising and studying the data.

- Extracting Peaks

Peaks can be used to determine the frequency of a signal, where within a sliding window frequency = (number of peaks)/(total time for the peaks to occur).

- Approximating slopes: dYdT = $\Delta y/\Delta t$

In some instances, a signal is composed of lines and vertices and therefore the slopes of the derivatives can be calculated to determine the changing slopes and populated as a new feature.

- Detrending

Detrending a signal can potentially improve a machine learning model by removing any linear or non-linear trends that may be present in the data [38]. This can help to make the data more stationary, which can in turn improve the accuracy and stability of the model. The built-in MATLAB function *detrend* is used to remove a linear trend in the data by removing the best straight-fit line from the data.

d. LIT101 data preprocessing

To illustrate the application of the feature extraction techniques, consider LIT101, a sensor that measures the raw water tank level in stage P1 of SWaT. Figure 10 shows an SSMP attack, where the aim is to overflow the raw water tank by setting the water level to a constant 700 mm. The steps taken to process the LIT101 sensor data were:

- Raw data were replaced with a smoothed signal. A median filter was chosen for its ability to keep edges. A 600th order and a truncate zero-padding method were applied to compute medians of smaller segments as they reached the signal edges.
- The signal appeared to be periodic, thus high-prominence peaks were extracted and differences between them populated as a new feature.
- The signal was composed of lines and vertices and therefore the slopes of the derivatives were calculated to determine the changing slopes and populated as a new feature. This implementation is illustrated in Figure 11. The data were first smoothed using a 200th order one-dimensional median filter with a truncate zero-padding method applied. The data were then resampled at a factor of 1/10. The difference between the resulting data points was computed and then the difference data were resampled at a factor of 10.

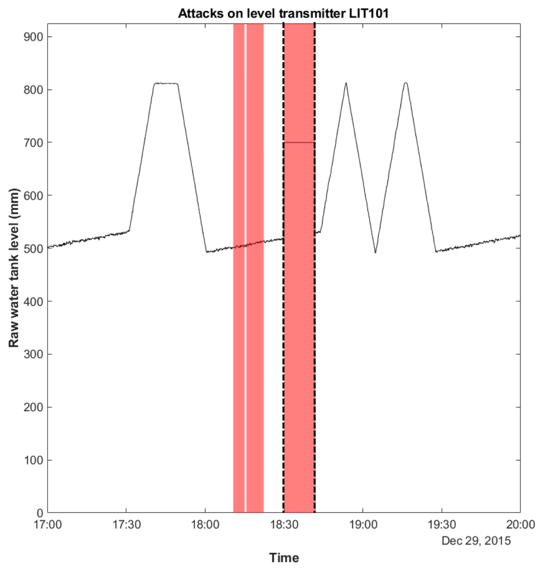


Figure 10. Attacks on level transmitter LIT101.

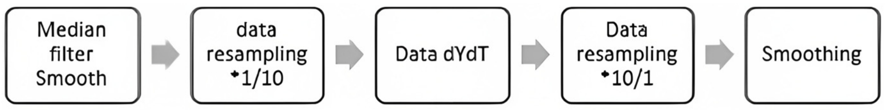


Figure 11. LIT101 sensor data process for extracting slopes.

3.6.5. Feature Scaling

For data standardisation, a z-score was used, where a data point's distance from the mean is measured in terms of the standard deviation [10]. This is performed so that distance-based algorithms can be trained on features with different units and ranges. To compute the z-score for a value x part of a variable X that has mean μ and standard deviation σ, the following equation is used:

$$z = \frac{(x - \mu)}{\sigma} \quad 0 \tag{4}$$

In MATLAB, the built-in function *zscore* is used to perform this task.

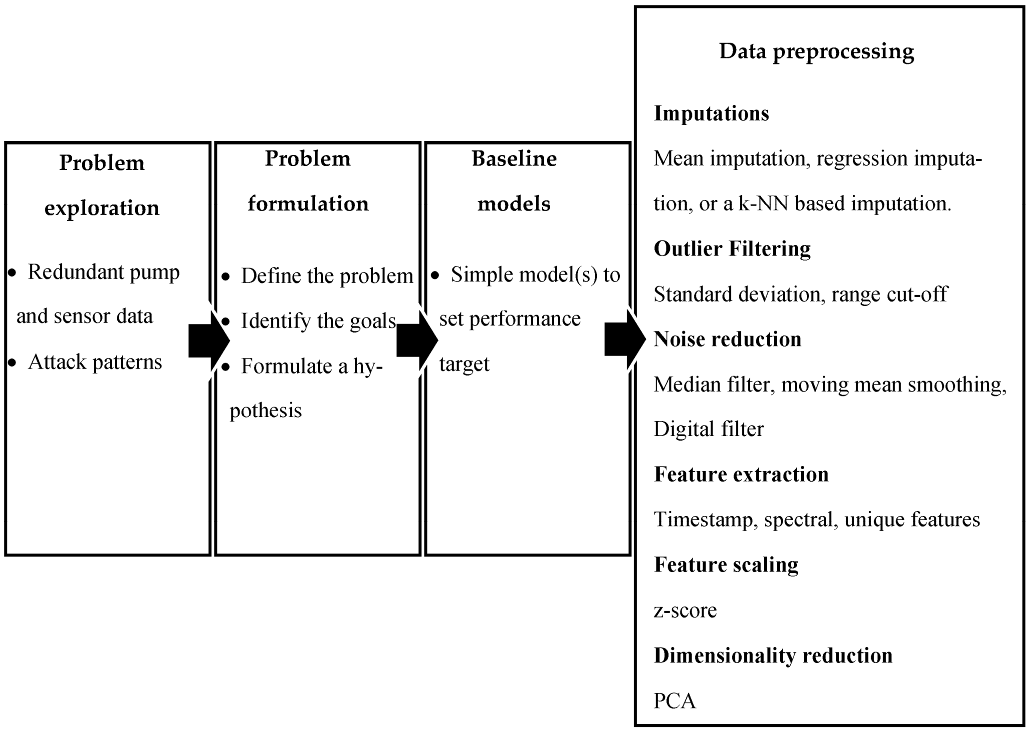
3.6.6. Dimensionality Reduction

PCA was used to reduce dimensionality in the dataset. The principal components must explain at least 99% of the variance to preserve much of the information.

3.7. Summary

In this section, we examined a systematic process for preparing critical water system infrastructure data to detect intrusion attacks using machine learning. The process begins with exploring the problem, where patterns, trends, and relationships in the data are identified, leading to an understanding of redundancies in the features that could pose issues of overfitting or increased variance. This information was then used in the problem formulation stage to develop a specific hypothesis about the anomalies being detected, determined to require supervised learning. Baseline models were introduced to set a performance target and evaluate the effectiveness of various machine learning algorithms and models, serving as a starting point for future improvement. Preprocessing then took place, with any missing values filled using methods such as mean imputation or k-NN-based imputation if necessary, outliers removed, and noise reduced through linear filters. Feature extraction was carried out to create a set of informative features for training a machine learning model. The data were then normalized through z-score normalization and dimensionality reduction through PCA could be performed if needed. The extracted features included timestamp, spectral, and unique features.

The workflow described is presented in Figure 12, which researchers working with similar data can apply.



Problem exploration
• Redundant pump and sensor data
• Attack patterns
Problem formulation
• Define the problem
• Identify the goals
• Formulate a hypothesis
Baseline models
• Simple model(s) to set performance target
Data preprocessing
Imputations
Mean imputation, regression imputation, or a k-NN based imputation.
Outlier Filtering
Standard deviation, range cut-off
Noise reduction
Median filter, moving mean smoothing, Digital filter
Feature extraction
Timestamp, spectral, unique features
Feature scaling
z-score
Dimensionality reduction
PCA

Figure 12. Workflow for treating process data from industrial control systems.

4. Experiments and Results

The work presented in this paper aims to provide ideal critical water system infrastructure-specific data preprocessing techniques for a resultant informative dataset to improve the performance of machine learning classification models. The output of this study is a data preprocessing framework that can be applied by researchers working with similar data. To provide a good framework, experiments were set up to determine the effectiveness of the preprocessing techniques.

4.1. Baseline Models

The trained baseline models described in Section 3 yielded the results provided in Table 2 below. The success of the data preprocessing steps was evaluated on the ability of the experiment models to perform better than the baseline models.

Table 2. Baseline model results.

Model	Accuracy	Precision	Recall	F1 Score	TTD
Tree	95.9%	85.9%	78.4%	82%	2447 s/40.8 min
Ensemble	95.3%	91.7%	66.6%	77.2%	4117 s/68.6 min

4.2. Experiment 1: Models Trained on New Dataset

The data preprocessing steps in Section 3 resulted in a larger dataset with increased dimensionality, which was meant to increase the informativeness of the dataset and thus improve the performance of the classifier. In this experiment the models were trained on the new dataset; the approach to conducting this experiment is depicted in Figure 13, where the dataset was first split using a 70:30 time-based splitting method and the training data was then used to train the models. The results are evaluated and documented in Table 3. A 70:30 train–test split was employed to ensure effective model training and accurate evaluation of model performance. This split strikes a balance between having a sufficiently large training set and a reasonably large test set.

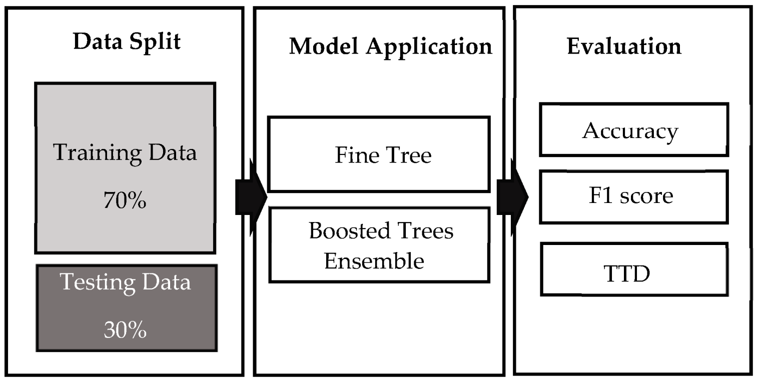


Figure 13. Experiment 1 approach.

Table 3. Experiment 1 results.

Model	Accuracy	Precision	Recall	F1 Score	TTD
Tree	96.5%	99.1%	71.8%	83.3%	3932 s/65.53 min
Ensemble	96.5%	99.1%	71.8%	83.3%	3932 s/65.53 min

4.3. Experiment 2: Feature Scaling

This experiment was conducted to determine the impact of standardisation on the algorithms' performances. To perform data standardisation, the built-in function *zscore* was used, where the input is the data matrix and the output is the z-score for each element in the input matrix, which are the standardised data.

The approach to conducting this experiment is depicted in Figure 14 and describes how the dataset was first standardised, thereafter it follows the same process in experiment 1 described previously. The models and the results are evaluated and documented in Table 4.

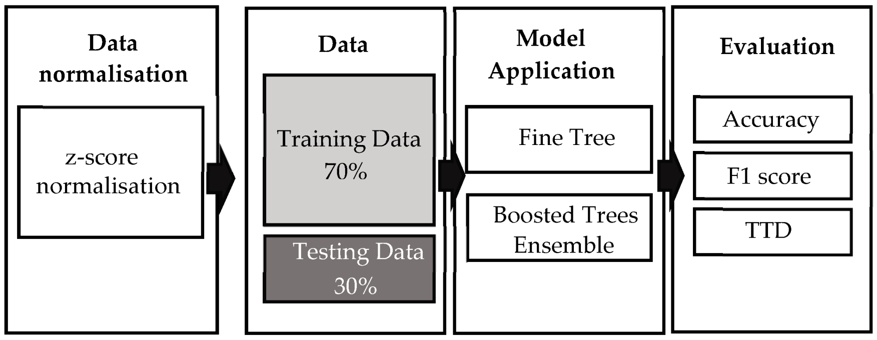


Figure 14. Experiment 2 approach.

Table 4. Experiment 2 results.

Model	Accuracy	Precision	Recall	F1 Score	TTD
Tree	96.5%	99.1%	71.8%	83.3%	−7 s/−0.1 min
Ensemble	96.5%	99.1%	71.8%	83.3%	3932 s/65.5 min

4.4. Experiment 3: PCA Applied

This experiment aimed to determine the effect of PCA. PCA was used to reduce the increased dimensionality that resulted from the data preprocessing steps and this experiment aimed to find out if this could be achieved without reducing the performance of the model. The approach to conducting this experiment is depicted in Figure 15, where PCA was applied to training data to explain 99% of the variance in the data before they were used to train the models. The trained models were evaluated on testing data with matching features to the training data, with reduced dimensionality. The results of this experiment are evaluated and documented in Table 5.

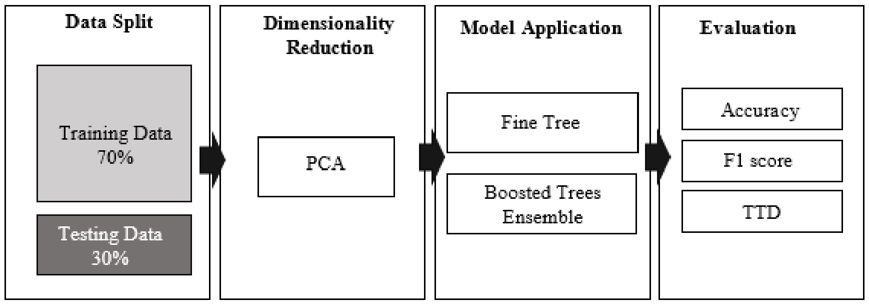


Figure 15. Experiment 3 approach.

Table 5. Experiment 3 results.

Model	Accuracy	Precision	Recall	F1 Score	TTD
Tree	86.7%	25.3%	5.6%	9.2%	33,705 s/561.8 min
Ensemble	95.5%	93.1%	67.4%	78.2%	3932 s/65.5 min

4.5. Experiment 4: Randomly Partitioned Data

This experiment aimed to randomly split the data rather than using the time-based data split used previously. The data were randomly split into 70% as training data and 30% as testing data. This method resulted in datasets that did not follow a time sequence; therefore, evaluating the TTD was not as straightforward anymore. To determine the TTD, a second part of the experiment was set up where the attack was isolated from the dataset before it was randomly partitioned and then added to the test data. The first part of the experiment gave an accurate representation of the accuracy similar to the same lengths of training and testing data in prior experiments used to train the models; the second part of the experiment gave a satisfactory time to detection evaluation. The approach to this experiment is depicted in Figure 16 and the results are evaluated and documented in Table 6.

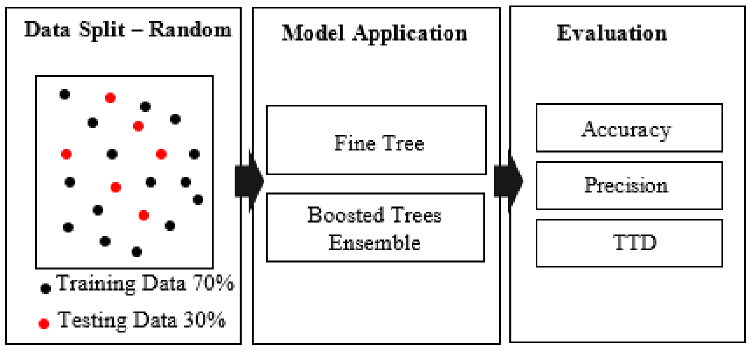


Figure 16. Experiment 4 approach.

Table 6. Experiment 4 results.

Model	Accuracy	Precision	Recall	F1 Score	TTD
Tree	96.5%	97.8%	97.8%	97.8%	0 s/0 min
Ensemble	96.5%	95.1%	97.5%	96.3%	0 s/0 min

5. Discussion

This section concludes the study by discussing the results and findings. In the methodology, baseline models were set up, forming a benchmark for comparison with the models applied in the different experiments. The baseline models were trained on unprocessed data. The aim of the study was to preprocess the data so that they yield improved results when applied to ML algorithms compared to the baseline models. The results of the four experiments discussed previously are discussed here.

5.1. Experiment 1: Models Trained on New Dataset

Feature extraction resulted in a larger dataset, and the question was whether the new dataset would result in improved performance of the models or not; this experiment aimed to answer this question. This resulted in an improved test accuracy and F1 score for both models. This was a good result, considering that it can still be improved using hyperparameters. The Fine Tree algorithm's TTD did worsen, however, unveiling a trade-off between classification accuracy and time to detection when using the new dataset. This can be attributed to the model being trained on irrelevant features, leading to overfitting and resulting in reduced performance; feature selection should be applied iteratively to determine the key features to train the model on. Furthermore, it can be noted that the results of both the Fine Decision Tree and Boosted Trees Ensemble models were found to be identical, indicating that the data did not have a complex structure that required the use of an ensemble Boosted Trees method. Therefore, in this case, the simpler Fine Decision Tree model provided similar performance to the Boosted Trees Ensemble method. Regardless, the results are satisfactory, and the ML modelling approach requires a careful consideration of classification algorithm to achieve the best results.

5.2. Experiment 2: Feature Scaling

This experiment was conducted to determine the impact of data standardisation on the performance of the algorithm. Z-score standardisation was used for this application.

The results showed that all three models performed better than the baseline models in accuracy, F1 score, and TTD. The worsened TTD of the Fine Decision Tree algorithm in Experiment 1 was improved in this experiment, as the attack was now detected 7 s before it happened and this can be interpreted as that the attack was detected immediately, meaning feature scaling improved the performance of the model.

Feature scaling will however not always give better results; in some cases such as this, it yields unchanged results for the Boosted Trees Ensemble algorithm but an improved TTD for the Fine Decision Tree algorithm. The careful choice of ML algorithms is important when feature scaling is to be applied, as some algorithms may not be affected at all by scaling, others may be negatively affected, and others may be positively affected. Distance-based methods such as k-NN and SVM, as well as deep learning methods such as neural networks, are positively affected by feature scaling [39]. An interesting observation is that feature scaling improved the training time compared to the non-scaled data; for this reason feature scaling should be applied when time is of the essence, especially when computational resources are limited.

5.3. Experiment 3: PCA Applied

Dimensionality reduction is an important part of ML modelling, and this experiment aimed to determine the effect of PCA on reducing dimensionality while keeping the informativeness of the dataset. The results showed that the Fine Decision Tree algorithm performed worse in all evaluation criteria than the baseline, while Boosted Trees Ensemble performed better in all aspects. The reduced dimensionality meant that the Fine Decision Tree algorithm had fewer data to learn on than it required, while the Boosted Trees Ensemble technique proved to be more robust than a single tree, as the literature review suggested.

5.4. Experiment 4: Random Partitioned Data

In previous experiments, data were split using a time-based method, while this experiment applied random splitting of the dataset into 70% and 30% randomly selected training and testing datasets, respectively, and this yielded the best results of all the experiments. All algorithms had improvements in accuracy, F1 score, and TTD.

5.5. What Can Be Learnt from the Results

- Data preprocessing in its entirety is a crucial part of ML modelling, as it has a direct effect on the results.
- Feature scaling can have a positive, negative, or neutral effect on the performance of the model and this requires careful judgement on whether to use it.
- While PCA reduces a dataset's number of features by retaining only the important ones, there is still a risk of losing useful information, which can affect other algorithms negatively.
- Feature selection can be applied after feature extraction on a dataset that contains both original and new features.

5.6. Limitations and Recommendations

This study focused on the data preprocessing stage of ML modelling; only three models were applied and hyperparameters were not used. An extension of this research must include more models to be evaluated and use hyperparameters.

Deep learning algorithms are beyond the scope of the work conducted in this study and were therefore not applied; they bear an advantage of automatic feature extraction and as such in future work it will be worth comparing manually extracted features to the automatically extracted features from deep leaning algorithms.

Although the SWaT dataset was crucial to the fulfilment of this study, it has limitations, as the data were collected over a set number of days and contained a set number of attacks. A virtual model or digital twin of a water treatment system can be created with an inclusion of disturbances and custom attacks that the user can configure. This way, future researchers can have access to unlimited and unique data.

6. Conclusions

The goal of this study was to enhance cybersecurity research on cyber-physical systems in critical water system infrastructure by offering a data preprocessing workflow and techniques that can be applied to similar data gathered from a similar environment to that used in the study of water treatment. The techniques outlined in this study, particularly feature engineering techniques, can be applied to data collected from common process sensors such as flow, level, and chemical analyser sensors, as well as pump actuators.

The dataset used in this study was collected under ideal conditions, and the added improvement from the experiments is a good indication of the impact that data preprocessing can have, even on a near-perfect dataset. Feature scaling may improve performance in some cases and should be considered when dealing with distance-based methods or neural networks. Having features on a similar scale can improve training time; for these reasons, feature scaling is a crucial part of data preprocessing and it should always be considered.

PCA is also an important part of data preprocessing; however, it may not always affect performance positively. It is most effective when variables are strongly correlated and it does not function effectively to reduce data when the association between variables is weak. In some cases, applying PCA and discarding some data has a risk of losing the informativeness of the dataset.

This study revealed that training the models on a randomly split dataset yields the best results. However, this brings a challenge of accurately evaluating the TTD. An extension to the research conducted in this study is to shift the focus to identifying a suitable model for critical water system infrastructure data, which also includes adding complexity to the models until the best performance is achieved. Following this, evaluation techniques can be investigated, resulting in workflows for data preprocessing, ML modelling, and algorithm evaluation for novice and expert researchers in cybersecurity research in critical water system infrastructure to apply.

Author Contributions: Conceptualization, I.V.M., D.T.R. and A.M.A.-M.; methodology, I.V.M.; software, I.V.M.; validation, I.V.M.; formal analysis, I.V.M.; investigation, I.V.M.; resources, I.V.M., D.T.R. and A.M.A.-M.; data curation, D.T.R.; writing—original draft preparation, I.V.M.; writing—review and editing, I.V.M., D.T.R. and A.M.A.-M.; visualisation, I.V.M.; supervision, D.T.R. and A.M.A.-M.; project administration, D.T.R. and A.M.A.-M.; funding acquisition, D.T.R. and A.M.A.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used in this study are from the iTrust centre of research in cybersecurity and Singapore University of Technology and Design. More information is available here: https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/ (accessed on 12 January 2023).

Acknowledgments: We would like to acknowledge the iTrust centre for research in cybersecurity and Singapore University of Technology and Design for allowing access to their SWaT dataset and the Telkom Centre of Excellence at the University of Cape Town (UCT) for their resources and support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ramotsoela, D.T.; Hancke, G.P.; Abu-Mahfouz, A.M. Attack detection in water distribution systems using machine learning. *Hum.-Cent. Comput. Inf. Sci.* **2019**, *9*, 13. [CrossRef]
2. Talcott, C. Cyber-physical systems and events. In *Software-Intensive Systems and New Computing Paradigms*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5380, pp. 101–115.
3. Humayed, A.; Lin, J.; Li, F.; Luo, B. Cyber-Physical Systems Security—A Survey. *IEEE Internet Things J.* **2017**, *4*, 1802–1831. [CrossRef]
4. Ericsson, G.N. Cyber security and power system communication essential parts of a smart grid infrastructure. *IEEE Trans Power Deliv.* **2010**, *25*, 1501–1507. [CrossRef]

5. Zhang, Y.; Lee, W.; Huang, Y.A. Intrusion detection techniques for mobile wireless networks. *Wirel. Netw.* **2003**, *9*, 545–556. [CrossRef]
6. Pfleeger, C. *Security in Computing*, 5th ed.; ACM, Inc.: Upper Saddle, NJ, USA, 1997. Available online: https://dl.acm.org/doi/book/10.5555/48805 (accessed on 18 March 2023).
7. Luengo, J.; García-Gil, D.; Ramírez-Gallego, S.; García, S.; Herrera, F. *Big Data Preprocessing*; Springer: Berlin/Heidelberg, Germany, 2020.
8. Mboweni, I.V.; Ramotsoela, D.T.; Abu-Mahfouz, A.M. A machine learning approach to intrusion detection in water distribution systems—A review. In Proceedings of the 47th Annual Conference of the IEEE Industrial Electronics Society (IECON), Toronto, ON, Canada, 13–16 October 2021.
9. García, S.; Luengo, J.; Herrera, F. Feature selection. *Intell. Syst. Ref. Libr.* **2015**, *72*, 163–193.
10. Fan, C.; Chen, M.; Wang, X.; Wang, J.; Huang, B. A Review on Data Preprocessing Techniques Toward Efficient and Reliable Knowledge Discovery From Building Operational Data. *Front. Energy Res.* **2021**, *9*, 652801. [CrossRef]
11. Ullah, Z.; Naqvi, S.R.; Farooq, W.; Yang, H.; Wang, S.; Vo, D.V.N. A comparative study of machine learning methods for bio-oil yield prediction—A genetic algorithm-based features selection. *Bioresour. Technol.* **2021**, *335*, 125292. [CrossRef]
12. Ashouri, M.; Fung, B.C.M.; Haghighat, F.; Yoshino, H. Systematic approach to provide building occupants with feedback to reduce energy consumption. *Energy* **2020**, *194*, 116813. [CrossRef]
13. Nawi, N.M.; Atomi, W.H.; Rehman, M.Z. The Effect of Data Pre-processing on Optimized Training of Artificial Neural Networks. *Procedia. Technol.* **2013**, *11*, 32–39. [CrossRef]
14. Halimaa, A.A.; Sundarakantham, K. Machine Learning Based Intrusion Detection System. In Proceedings of the 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 23–25 April 2019; pp. 873–888.
15. Liu, Y.; Ma, X.; Li, Y.; Tie, Y.; Zhang, Y.; Gao, J. Water pipeline leakage detection based on machine learning and wireless sensor networks. *Sensors* **2019**, *19*, 5086. [CrossRef]
16. Bijlsma, S.; Bobeldijk, I.; Verheij, E.R.; Ramaker, R.; Kochhar, S.; Macdonald, I.A.; van Ommen, B.; Smilde, A.K. Large-scale human metabolomics studies: A strategy for data (pre-) processing and validation. *Anal. Chem.* **2006**, *78*, 567–574. [CrossRef]
17. Zhu, X.; Wu, X. Class noise vs. attribute noise: A quantitative study. *Artif. Intell. Rev.* **2004**, *22*, 177–210. [CrossRef]
18. Kang, Q.; Chen, X.S.; Li, S.S.; Zhou, M.C. A Noise-Filtered Under-Sampling Scheme for Imbalanced Classification. *IEEE Trans. Cybern.* **2017**, *47*, 4263–4274. [CrossRef]
19. Li, W.; Mo, W.; Zhang, X.; Squiers, J.J.; Lu, Y.; Sellke, E.W.; Fan, W.; DiMaio, J.M.; Thatcher, J.E. Outlier detection and removal improves accuracy of machine learning approach to multispectral burn diagnostic imaging. *J. Biomed. Opt.* **2015**, *20*, 121305. [CrossRef] [PubMed]
20. Lam, J.C.; Wan, K.K.W.; Cheung, K.L.; Yang, L. Principal component analysis of electricity use in office buildings. *Energy Build.* **2008**, *40*, 828–836. [CrossRef]
21. Secure Water Treatment—iTrust. Available online: https://itrust.sutd.edu.sg/testbeds/secure-water-treatment-swat/ (accessed on 12 January 2023).
22. Wankhede, S.B. Anomaly Detection using Machine Learning Techniques. In Proceedings of the 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), Pune, India, 29–31 March 2019.
23. Prasad, N.R.; Almanza-Garcia, S.; Lu, T.T. Anomaly detection. *Comput. Mater. Contin.* **2009**, *14*, 1–22.
24. Ariyaluran Habeeb, R.A.; Nasaruddin, F.; Gani, A.; Targio Hashem, I.A.; Ahmed, E.; Imran, M. Real-time big data processing for anomaly detection: A Survey. *Int. J. Inf. Manag.* **2019**, *45*, 289–307. [CrossRef]
25. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422.
26. Breiman, L.; Friedman, J.H.; Olshen, R.A.; Stone, C.J. *Classification and Regression Trees*, 1st ed.; Routledge: New York, NY, USA, 1984.
27. Quinlan, J.R. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [CrossRef]
28. Freund, Y.; Schapire, R.E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
29. Rokach, L. Ensemble-based classifiers. *Artif. Intell. Rev.* **2010**, *33*, 1–39. [CrossRef]
30. Lyons, R.G. *Understanding Digital Signal Processing*, 2nd ed.; Prentice Hall PTR: Hoboken, NJ, USA, 2004.
31. Varoslavskiy, L.P. Digital Image Processing. *Telecommun. Radio Eng. (Engl. Transl. Elektrosvyaz Radiotekhnika)* **1977**, *31–32*, 42–47.
32. Oppenheim, A.V.; Schafer, R.W.; Buck, J.R. *Discrete-Time Signal Processing, 2nd ed*; Prentice Hall PTR: Hoboken, NJ, USA, 1999.
33. MathWorks. Filter. Available online: https://www.mathworks.com/help/matlab/ref/filter.html (accessed on 6 January 2023).
34. Han, Z.; Zhao, J.; Leung, H.; Ma, K.F.; Wang, W. A Review of Deep Learning Models for Time Series Prediction. *IEEE Sens. J.* **2021**, *21*, 7833–7848. [CrossRef]
35. Rubin, A.; Geva, N.; Sheintuch, L.; Ziv, Y. Hippocampal ensemble dynamics timestamp events in long-term memory. *Elife* **2015**, *4*, e12247. [CrossRef] [PubMed]
36. Lakhina, A.; Crovella, M.; Diot, C. Diagnosing network-wide traffic anomalies. *Comput. Commun. Rev.* **2004**, *34*, 219–230. [CrossRef]
37. Vishnoi, V.; Kumar, K.; Kumar, B. A comprehensive study of feature extraction techniques for plant leaf disease detection. *Multimed. Tools Appl.* **2022**, *80*, 367–419. [CrossRef]

38. Engel, J.; Gerretzen, J.; Szymańska, E.; Jansen, J.J.; Downey, G.; Blanchet, L.; Buydens, L.M. Breaking with trends in pre-processing? *TrAC Trends Anal. Chem.* **2013**, *50*, 96–106. [CrossRef]
39. Latyshev, E. Sensor Data Preprocessing, Feature Engineering and Equipment Remaining Lifetime Forecasting for Predictive Maintenance. In Proceedings of the International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2016), Moscow, Russia, 11–14 October 2016.