UNIVERSITY of BRADFORD

Faculty of Engineering and Informatics
Department of Computer Science

Machine learning for anomaly detection in network traffic

Dissertation Project Report
Yahya Feroze
21023551

A thesis submitted in part fulfilment of the degree of <BSc (Hons) Computer Science>
Supervisor: Prof. I.A.M. Academic

25th October 2023

Declaration

The candidate confirms that the work submitted is his/her own and that appropriate credit has been given where reference has been made to the work of others. The candidate agrees that this report can be electronically checked for plagiarism.

Yahya Feroze

Table of Contents

1. Abstract

Millions of people around the world have used the internet for browsing or to communicate with each other, this consequently led to thousands of cyber-attacks online in the past two decades. Attackers are able to steal data or send large quantities of spam and phishing emails causing servers to overload and eventually crash. Despite the potential of signature-based methods in mitigating these attacks, they may not be effective against zero-day attacks. A solution to this is to use Anomaly-based approaches using machine learning to detect both signature and zero-day attacks. Chosen for its relevance and diverse range of attacks, in 2017 a data set has been created showing the evaluation of intrusion attacks that occurred throughout the week. Feature selection then was conducted on the CICID2017 dataset using a random forest classification algorithm. This study will apply k-nearest neighbours () and random forest algorithms() to analyse and highlight the evolution of machine learning's network security and the potential it has in addressing cyber threats.

2. Table of Contents

The rise in cyber threats has increased as the internet continues to evolve, to overcome these threats there are 2 types of methods that help maintain information security: Signature-based methods and anomaly-based detection.
Signature-based methods rely on a created database that compares the signatures of known cyber threats against scanned files and network traffic. While this method is successful, the database would need to be constantly updated with new signatures as cyber threats evolve. Furthermore, even if the database is constantly updated it can be still vulnerable to previously unseen attacks (zero-day attacks). Whereas anomaly-based detection can detect attacks it has not seen before as it focuses on detecting unusual behaviour in network traffic. therefore, effective against zero-day attacks.

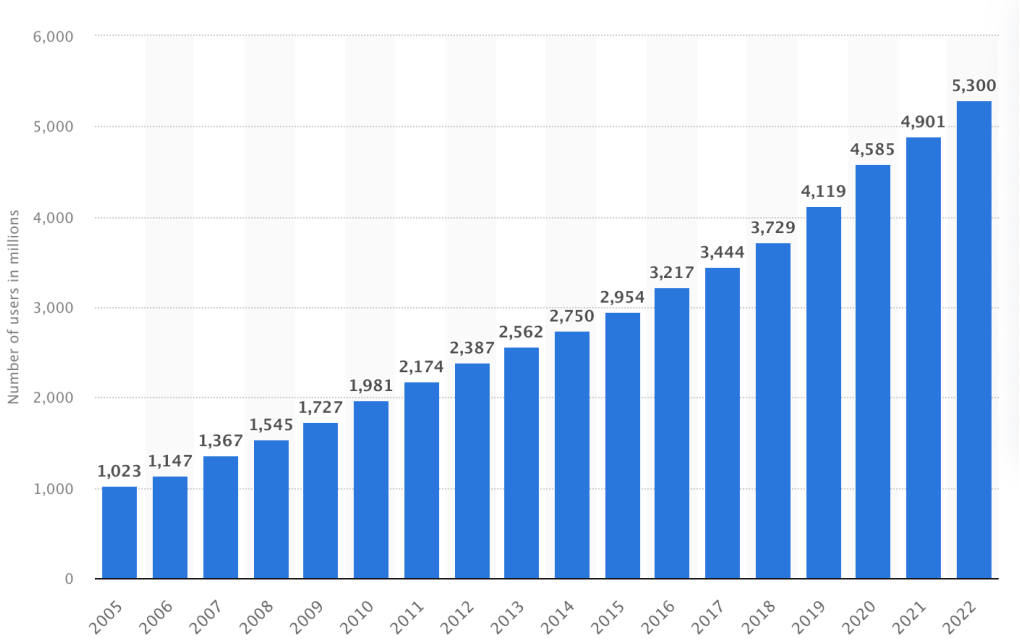


Figure1. shows the internet usage increasing according to years (2005,2022) [1]

In addition, increasing day by day, more than half of the internet's data is protected with encryption using the protocols: Secure Sockets Layer and Transport Layer Security. [3] This further means that signature-based methods are less effective as they are unable to monitor the contents of the encrypted internet stream. On the other hand, the anomaly-based approach will analyse data by its properties such as connection time, size, and number of packets. Therefore, does not depend on the content of the message and can effectively analyse data that is encrypted, demonstrating how it can adapt to security protocols. Given these strengths anomaly-based detection methods in used extensively in detecting and preventing network attacks.

This study aims to develop a system that can swiftly detect anomalies in network traffic using effective methods of machine-learning.

Goals and objectives

Goals that should be achieved by the end of this study:

- To conduct a comprehensive study of machine learning algorithms that can be used to detect network anomalies.
- Be able to identify network attacks effectively by using methods involving machine learning for network anomaly detection.
- comparing the outcome with those of previous research in this field.
- Contribute to academic literature through network anomaly detection, aligning with previous studies.

Objectives:
The study sets out to achieve the following objectives:

- Examine previous work by doing extensive field research in network anomaly detection.
- Choose the most suitable dataset after thorough research on available alternatives.
- Selecting the appropriate algorithm for machine learning through detailed investigation.
- Choosing the right software platform for the research to be performed
- Picking suitable hardware necessary for the study to take place.
- Establish the correct criteria for evaluating the study.

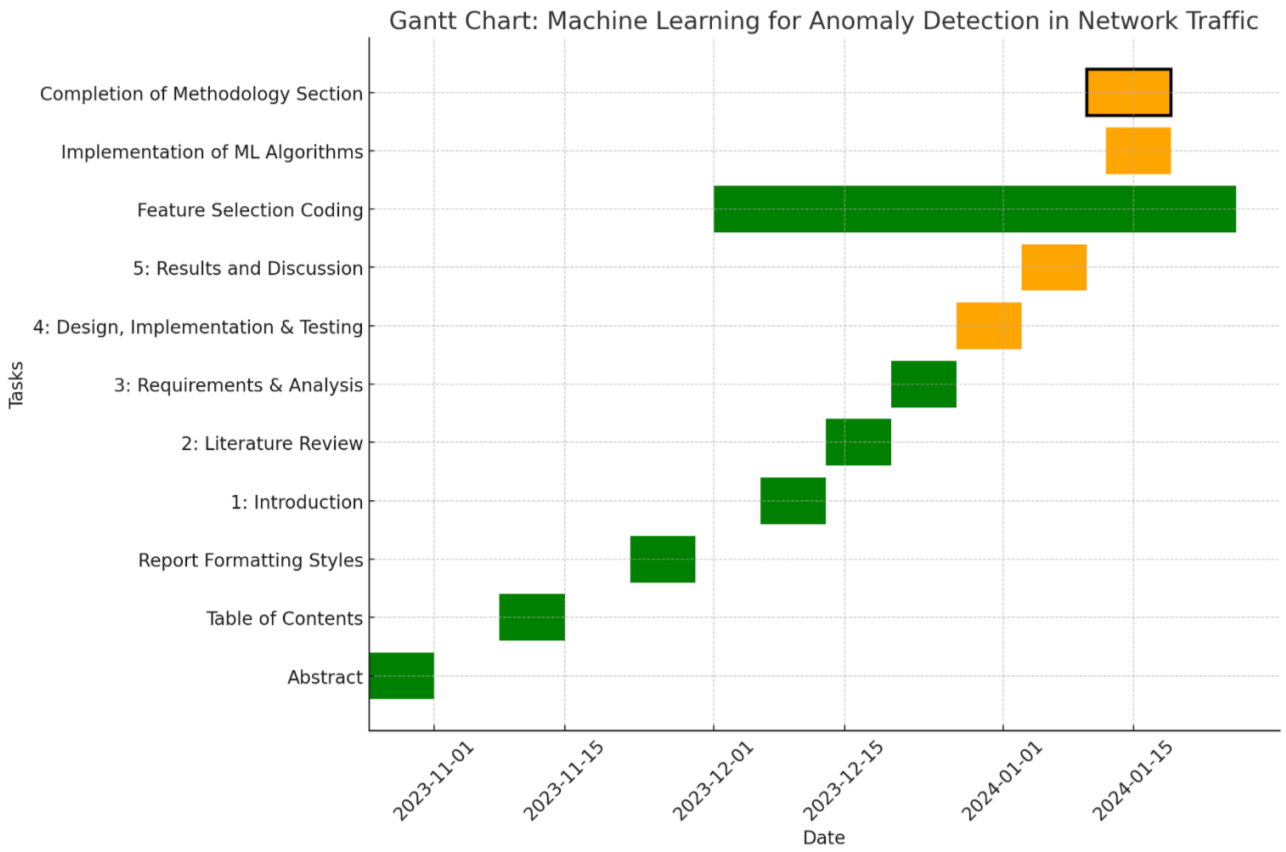


Figure 2 shows Gantt chart displays the progress of my project over 2 months.

structure of report

Background and related work

This section will give a better understanding by introducing preliminary information such as details about various data sets and different types of anomaly attacks. Additionally, a detailed description of various attacks is further discussed with information about machine learning algorithms. To conclude this section a discussion of similar studies is reviewed in the literature.

Methodology

The methodology will be divided into two subsections: tools and methods and implementation.
The first section will provide a detailed description of the software and hardware used in the study, as well as the performance and evaluation methods. implementation will include the steps of cleaning and pre-processing the dataset and dividing the dataset into training and testing. The next step in the process involves selecting the appropriate features and implementing machine learning algorithms.

Result and discussion

The results from the previous implementation step are shared and their cause-effect relationships as well as alternative approaches are discussed.

Evaluation

in this section, the data obtained is compared to studies from the literature review.

Conclusion

The conclusion will briefly summarise the study and also talk about possible developments and innovations that could be applied to the study in the future.

Background and related work

To detect anomalies using machine learning, datasets need to include a large amount of harmful and harmless traffic so that the step of splitting data into training and testing can be carried out. However, because of privacy issues real network traffic cannot be shared or used publicly. the need for data means that many datasets are being produced and continue to be developed. this section will evaluate

different datasets comparing them to each other and find the most suitable one for use in implementation in the next step.

KDD 99 [4]

The kdd99 dataset is a well-known dataset in the network security space for intrusion detection systems. created by the University of California, Irvine, it was developed for the third international knowledge discovery and data mining tools competition (KDD Cup 99). The dataset was specifically created to help in developing systems to prevent intrusions in network security. kdd99 dataset was based on the DAPRA98 dataset created in 1998, which consisted of normal and malicious network traffic simulated to represent a typical US Air Force LAN environment. 21 properties have been created by applying a feature extraction process to be used by machine learning methods [3].

The structure of the KD99 dataset is divided into two, a training set which includes 4,898,431 records of data, and a testing set containing 311,029 records. kd99 features 38 simulated attack types that were able to cover a wide range of intrusion scenarios, 14 of them being unknown attacks and unique to the test set. this was a crucial aspect for evaluating the ability of machine learning models to detect new or unseen types of network intrusions.

While the KDD99 intrusion detection research has played an important role in the progression of today's day, it still faces some criticism over the years such as the redundancy in data and its artificially created nature, there are doubts regarding its ability to fully simulate actual network traffic in real-world scenarios. But despite these limitations, KDD99 dataset is still used in numerous research as a reference point in anomaly detection in network traffic [5].

NSL-KDD [6]

An improved dataset was created named NSL-KDD, which addressed some notable limitations of KDD99. Despite being a good alternative to DAPRA98, KDD99 had some major issues that needed to be worked on, such as its high redundancy containing many repetitive entries which could affect research results and the performance of the machine learning algorithms. And its large size led to researchers only using parts of the dataset which means that they could not capture all the dataset's properties.[7]

To overcome these drawbacks in 2009 the NSL-KDD was created by Tavallaee et al, this dataset was an improvement as it eliminated any mistakes and repetition that were found in the kdd99 dataset.

NSL-KDD was structured into four parts, which were categorised under two main sections, training, and testing. The four parts included: a full training dataset (KDD Train+), 20% of the training data (KDDTrain+_20Percent), test data (KDDTest+) and a smaller version of the test data that includes all difficulty levels (KDDTest-21).[8]

The dataset is also recognised for its more manageable size, making it a valuable resource for network intrusion detection. it is overall more balanced for testing against intrusion threats.[9]

ISCX2012 [10]

The ISCX2012 dataset was created to address the existing dataset's limitations, specifically their outdatedness and non-representative environment of real-world data. The dataset was created using a seven-day internet traffic stream from a testbed setup by the Canadian Institute for Cybersecurity. This was a significant stride towards using real-world datasets in the future.[11]

The dataset includes many different protocols such as:

- FTP (file transfer protocol),
- SSH (secure shell), POP3 (POST OFFICE PROTOCOL 3)
- SMTP (simple mail transfer protocol),
- IMAP (INTERNET message ACCESS PROTOCOL,
- HTTP (Hypertext transfer protocol).

The features also included real and malicious network traffic. All the data in the ISCX2012 dataset are labelled assisting in easier analysis and research on intrusion detection.
The dataset contains diverse attack types, this includes infiltration, DOS (denial of service), DDOS (distributed denial of service) and brute force SSH attacks. Providing a broad recourse for various cyber threats.
However, there are still some limitations that need to be addressed for the ISCX2012 dataset, SSL, and TLS (traffic socket layer/transport layer security) traffic are needed, which is a significant portion of present-day internet traffic. This could suggest that studying this dataset may not meet the needs of modern network traffic environments that extensively use encryption.[11]
To summarise, the ISCX 2012 dataset represents a significant step in using real-world data for network security research as it provides a more realistic and contemporary resource for the development and evaluation of intrusion detection systems. However, its lack of SSL/TLS traffic is an important consideration for researchers focusing on current and future network security challenges.

CICIDS2017 [12]

The final dataset that will be addressed is CICIDS2017. Created by the Canadian Institute for Cyber Security at the University of New Brunswick. this dataset consisted of a 5-day (3rd of July – 7th of July 2017) data stream on a network that was created by computers using operating systems such as Windows Vista/7/8.1/10, Mac, Ubuntu 12/16, and Kali. [3]

The table below shows details of the dataset (table 1)

Flow recoding days (working hours)	Duration	Attack name	Flow count	Csv file size	Pcap file size
Monday	All day	None	529918	257MB	10GB
Tuesday	All day	FTP Patator/SSH Patator	445909	166MB	10GB
Wednesday	All day	• Dos • hulk • Dos gold eye • Dos slow Loris • dos HTTP test • heart bleed	692703	272MB	12GB
Thursday	Morning	Web Attacks (brute force XSS, SQL injection)	170366	87.7MB	7.7GB
	Afternoon	Infiltration	288602	103MB	
Friday	Morning	Bot	192033	71.8MB	8.2GB
	afternoon	DDOS	225745	92.7MB	
	Afternoon	Port scan	286467	97.1MB	

This dataset has many advantages such as: containing real-world data from a testbed of real computers, showing the operating systems' diversity between attacker and victim computers, raw data (Pcap files) and processed data (CSV files) are provided, cicids2017 includes 85 extracted features all being labelled for machine learning and containing a wide up to date range of attacks. The dataset also covers many protocols like HTTPS, FTP, HTTP, SSH and email protocols.
However, with any dataset, there is still room for improvement for example: raw and processed data files are very large (49GB), unlike some other datasets like KDD99 and NSL-KDD, there are no separate training and testing files meaning users would have to create their own. The development of DARPA98, KDD99 and NSL-KDD show the continuous improvement they have gone through, with

each new dataset developed by addressing flaws and the limitations of predecessors. On the other hand, the CICIDS2017 is relatively recent and has not undertaken extensive analysis which could suggest that it may have some minor errors. This is further discussed in detail in the data cleansing section.
This section concludes with the selection of the processed CICIDS2017 dataset (CSV files) for use in the implementation phase following a comparative analysis.
its extensive range of protocols, attacks and how fairly recent the dataset is playing a key part in choosing this preference. Moreover, since there are only a few studies using this dataset, involvement with the dataset may provide an important addition to existing research. Primarily focusing on Friday morning dataset which contains attacks from botnet.

Anomaly and attack types [13]

An anomaly is a sample that is not defined with characters that a normal sample contains, for a normal sample the rules must be specified and valid for the anomaly to be understood. this is analysed under three headings:

Point anomaly: This anomaly type refers to a single data point that significantly differs from the rest of the dataset. To detect this, statistical methods could be used to find outliers. An example of this could be in security where a point anomaly may signal fraudulent activity on a credit card if the spending pattern of a user changes drastically.[13]

Contextual anomaly: this anomaly usually occurs when there is unusual behaviour in data in specific scenarios, However, this type of anomaly should be approached with caution as it may sometimes just be noise. for example, spending a large amount of money whilst you are on holiday is normal and shouldn't be flagged without context.[13]

Collective anomaly: refers to a group of data points that deviate from the patterns that are expected on a dataset, an example could be an increase in spending on Valentine's Day, which is common for the flower industry. So, this should be taken into consideration before being flagged without context.[14]

Network security and attack types [13,14]

The main objective of network security is to protect against attacks that are based on three fundamental principles.

1. **Confidentiality:** only authorized users should have access to information to keep the network secure there are safety measures that are put in place to help prevent unauthorized access. One of the most common methods used to maintain confidentiality is encryption, making sure that data cannot be intercepted making sure it cannot be understood by unauthorized parties.

2. **Integrity:** altering data such as modifying or deleting should only be done by authorized users. to detect any unauthorized changes integrity checks and hash functions are used.

3. **Availability:** users who are authorized should constantly have access to the information system and resources. Redundant systems and regular backups can help maintain availability in the face of attacks such as Denial of Service (DoS).

   Network attacks are deliberate attempts to breach confidentiality, integrity, and availability, and they can be broadly categorized into four types:

Denial of service (DOS)

with this type of attack, attackers overload a system's resources to deny normal user's access. One common tactic is to flood a web server with an excessive number of requests. DOS can be divided into two sub-types, bandwidth depletion and resource depletion. Bandwidth depletion works by overwhelming the user's (victim) internet bandwidth.[13] Whereas resource depletion will target a system's memory and processing power with a lot of packages (tasks). Network hardware and software solutions can be used to mitigate DoS attacks by filtering out malicious traffic.[15]

Probe (information gathering)
These attacks aim to Gather sensitive information about the target's network, Attackers can obtain information such as network structure, device types, and operating systems. this intelligence is often used to plan future attacks that can be more of a threat.
U2R (User to root)[13]
in this type of attack, attackers try to gain administrative control over resources. methods that could be utilised are brute force attacks or exploiting system vulnerabilities. To prevent these attacks users should regularly update their password policies.[13]

R2U/R2L (Remote to user/local)
Attackers attempt to gain unauthorized access to networks to send packets and execute commands. They use methods such as brute force or exploiting vulnerabilities similar to U2R attacks. This is why Network monitoring and intrusion detection systems are critical in identifying and mitigating such threats.[13]

Robust security measures are crucial to protect network resources against various attack types that pose significant risks to network security, compromising confidentiality, integrity, and availability.

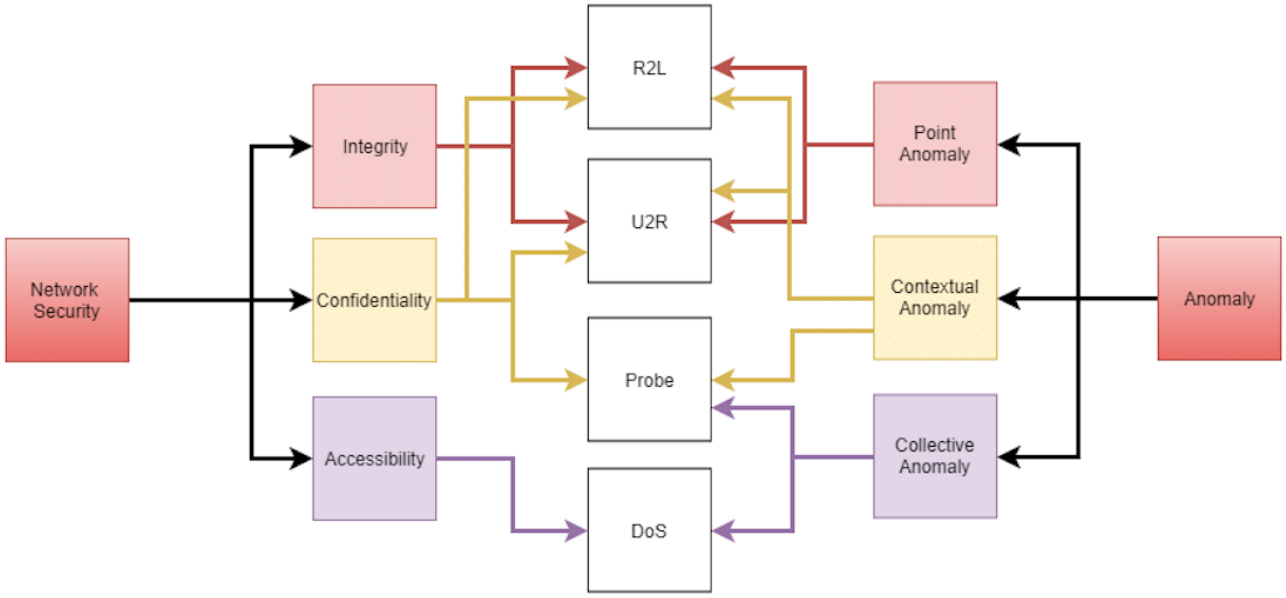


Figure 3 shows the correlation between network anomalies and network attacks.

Classifying network attacks by the type of anomalies they create helps aid in detection. For example, DOS (denial of service) attacks usually disrupt network operations as they are known for increasing the amount of data flow and packet numbers. Due to the large volume of network traffic, they generate, these attacks can be considered as **collective anomalies.** Whereas U2R (User to root) and R2U/R2L (Remote to user/local) attackers target specific users, ports, or systems this is purposely done so that they can gain unauthorised access. They are often classified as contextual anomalies when they occur under specific conditions, or as point anomalies when they are an isolated incident. Additionally, Probe attacks, which are reconnaissance activities can cause dense traffic and are aimed at gathering information. They are considered as both contextual, due to their targeted nature, and collective anomalies as they create a large volume of traffic. By understanding the type of anomalies that are associated with different attacks, better security systems can be designed to help prevent these attacks from happening for example using a real-time monitoring system that can monitor network traffic in real time and detect and respond quickly to any anomaly that is identified. Also having strong security protocols can help prevent potential exploits that

can lead to these types of network anomalies. The relationship between network anomalies and network attacks is summarized in Figure 3.

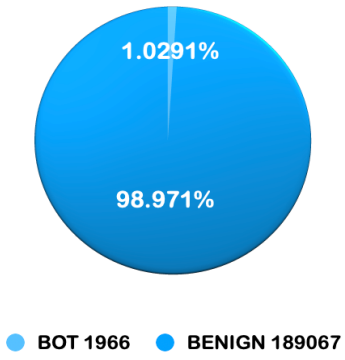
Botnet Attack[3,16,17]

In this section, the types of attacks that the data set contains are examined.

A botnet is a network of computers that have been infected with malware turning them into 'zombies' or 'bots. These infected computers are remotely controlled by a 'bot master' which can be used to carry out any harmful activities, for example, sending spam (unsolicited emails)

and executing DDOS (distributed denial of service) attacks. Detecting such attacks can only be done if the network is active, as a passive botnet displays no network activity. Botnets can usually be indicated with signs such as the number of bytes per packet and the presence of packets with the PSH flag, during active phases. Within the CICIDS2017, botnet attacks were conducted using a tool written in Python known as the Ares attack tool. This attack takes place on Friday morning records within the CICIDS2017 dataset.

To prevent such attacks, it is suggested to implement strong security measures for example, firewalls and antivirus software. Additionally, we can further prevent botnet attacks from occurring by educating users on safe browsing practices. Users should avoid clicking suspicious links, as this can reduce the risk of botnet propagation. Monitoring traffic constantly for any unusual behaviour can also help in detecting botnets early.
Creating and controlling botnets is illegal and shouldn't be done, being a part of a botnet-infected computer can have legal repercussions even if the users are unaware.
Botnets can have a significant impact on a network's global infrastructure as they can carry out large-scale attacks like DDOS which can lead to websites and services being taken down as they are compromised.



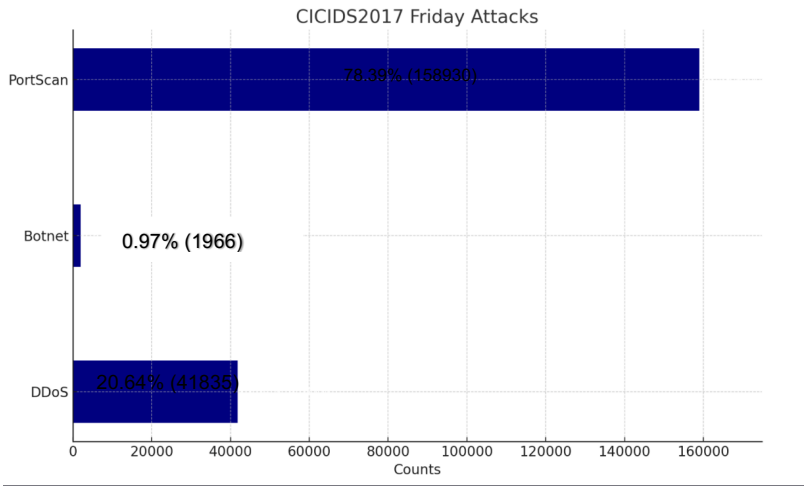


Figure 4 shows the distribution of attacks that occurred on Friday.
Figure 5 shows the distribution of botnet attacks that occurred on Friday morning.

Machine learning [18]

Machine learning teaches computers to learn from data by merging the scientific approach with creative problem solving. this involves training computers on a dataset and evaluating the performance on a separate test set. this aims to minimise human intervention by automating the learning process. Advantages over a more classical approach are listed below:

- Capable of interpreting vast and complex datasets.
- They can solve intricate problems that cannot be solved by classic methods.
- They can provide solutions that require little or no additional updates from another person.

- Machine learning can adapt to different environments by learning directly from data continuously evolving.

Machine learning algorithms can be classified into four groups based on the type of training data and supervision received. These groups are supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning []. Supervised learning involves labelled data, while unsupervised learning involves unlabelled data. Semi-supervised learning is a combination of both labelled and unlabelled data, and reinforcement learning involves learning from feedback received through interactions with an environment.[18]

Supervised machine learning: trains on data that has already been classified and labelled, they use labels to learn how to predict outcomes. Then they are tested on new data to evaluate their predictive accuracy this is done by comparing the algorithm's predictions against the label. The need for labelled data can lead to a high cost, as it often requires manual labelling or the use of specialised services. A critical step in supervised machine learning is ensuring that data is handled properly through the pre-processing stage, this involves techniques that handle missing values and feature selection.
Cross-validation methods are essential to ensure that a model can generalize well, instead of memorizing the training data. This robustness is very crucial, given the broad scope of applications of supervised learning in various domains. It leverages past data to forecast outcomes in a wide range of tasks, such as financial risk assessment and medical diagnosis. Popular supervised learning algorithms include Decision Trees, K-Nearest Neighbours, and Random Forests, chosen for their ability to handle complex datasets and provide interpretable results.[18]

Unsupervised machine learning: does not require labelled data, this would make the process less reliant on manual classification or external labelling services. These algorithms group the data based on inherited similarities, identifying patterns as well as natural clusters, often through iterative methods to refine these groupings over multiple iterations.
Unsupervised learning has broad applications, particularly excelling in areas like anomaly detection and relationship learning. In anomaly detection, it is highly effective in identifying outliers or unusual patterns within datasets, a crucial task for identifying fraud or network intrusions. Meanwhile, in relationship learning, unsupervised learning proves invaluable for uncovering hidden associations between variables in datasets, revealing insights that might not be apparent through manual analysis. These capabilities make unsupervised learning a powerful tool for a wide range of data analysis tasks.[18]
because unsupervised learning does not require labelling means that it is more economical compared to supervised learning. Unsupervised learning is exceptionally good in extracting insights from unstructured data where relationships and patterns aren't apparent. Examples of this algorithm may Include k-means clustering for group identification, PCA for reducing data complexity, and the Apriori algorithm for learning association rules.

Semi-supervised learning: combines elements of both supervised and unsupervised learning methodologies. Labelling is done to only a small subset of data and the majority of the dataset remains unlabelled. Advantages include merging a high accuracy of supervised learning due to the reduced amount of labelled data with the economic benefit of unsupervised learning because less manual labelling is required. This method is useful in scenarios where obtaining fully labelled data is too expensive. Semi-supervised learning utilises larger datasets by incorporating labelled and unlabelled data which can lead to generalisation in models. They are commonly used in scenarios like web content classification where unlabelled data is available.
This approach can significantly improve learning accuracy when only limited labelled data is available, a common situation in many real-world datasets. Semi-supervised learning stands out for its ability to harness the strengths of both supervised and unsupervised learning, making it a versatile and cost-effective choice for many practical machine-learning applications.

Reinforcement learning:
As a basic principle, this approach is very different compared to the other three. As the system receives a penalty for each wrong choice it makes during training as well as a reward for the right choices. the algorithm constructs its own rules.
For this project, we'll be employing supervised learning techniques, leveraging the strengths of our well-labelled dataset. This strategy aims to maximize performance benefits while avoiding the higher costs often associated with extensive data labelling.

Decision trees
Decision trees are a widely used classifier in machine learning due to their straightforward and interpretable rule-based structure. Each decision tree consists of nodes, branches, and leaves. nodes include a root node and sub-nodes which each contain a decision-making statement. Branches from each node lead to the next decision point, they are usually only two however can be more in certain variants. The process concludes in leaves as it represents the final decision or classification.
The algorithm progresses through the tree by making decisions at each node, following the branch related to the outcome of the decision, until a leaf is reached. The rules in a decision tree are clear and easy to understand, making them a transparent option for classification tasks. They can handle both numerical and categorical data and are used in various domains for classification and regression tasks.[19]
A divide-and-conquer approach is typically used in decision tree methods, effectively simplifying large and complex datasets into smaller and more manageable groups. One challenge that may occur when using decision trees is that the model becomes overfitting and prevents getting fair results. Another disadvantage in decision trees is that errors in earlier stages can lead to incorrect branches giving misleading results.
To stop these issues from happening it is essential to pre-process the data before using it in a decision tree algorithm which ensures reliability and accuracy in the classification process.
By utilising decision trees, machine learning models can efficiently classify data through a series of simple, interpretable decisions, leading to robust and understandable outcomes.

Random forest
Random Forest is a machine-learning approach that uses decision trees. In this method a "Forest" is constructed with many decision trees, each differing in structure. A specified number of decision trees (N) are created, with each tree being trained on a randomly resampled subset of the training data. Predictions from each tree are accumulated through a voting process, with the most common prediction among the trees chosen as the final output.

The random forest has many advantages. These can be listed as follows [20]:

- **Effectiveness with Complex Data**: Handles large and complicated datasets efficiently.
- **Reduced Overfitting**: Less prone to overfitting compared to individual decision trees.
- **Versatility**: Applicable to various types of machine learning problems.
- **Handling Missing Values**: Capable of managing missing data by imputing values.
- **Feature Importance**: Calculates and utilizes the importance level of variables, useful for feature selection.

On the other hand, there are still some challenges that should be considered before using a random forest algorithm such as the algorithm complexity as it is made up of a lot of decision trees the structure can be quite complex which would additionally interpret the model may be difficult because of the complexity.

To summarise the Random Forest method, with its ensemble approach, offers a powerful and reliable way to conduct machine learning tasks, especially when dealing with complex datasets requiring accurate decision-making.

Related works

this section focuses on various studies that use machine learning to detect anomalies. The use of machine learning performance and dataset ratios are given in each study. focusing on different machine-learning algorithms and datasets.

In a study conducted in 2005 by Chebrolu et al, a hybrid approach was taken combining Bayesian networks and classification regression trees, to produce a more efficient detection process by reducing the number of features in the dataset. This was applied to a KDD cup 99 data set that obtained the accuracy: Normal (Benign): 100%, Probe: 100%, DOS: 100%, U2R: 84% and R2L: 84%. [21]

In a 2007 study, the Support Vector Machines (SVM) method was applied to the DARPA 98 dataset, yielding high accuracy rates for Normal (98%), Probe (88%), and DOS (84%) classifications. However, the study noted a significant decrease in accuracy for U2R and R2L attacks, with rates dropping below 20% (specifically 0% and 18%). To enhance the performance of the SVM method, the study incorporated a Dynamically Growing Self-Organizing Tree (DGSOT), aiming to improve the algorithm's effectiveness across all attack types.[22]

In 2011, Suresh and Anita conducted a study focusing on DDoS (Distributed Denial of Service) attacks, utilizing the CAIDA dataset. The research implemented various machine learning algorithms including Fuzzy C-Means, Naive Bayesian, SVM (Support Vector Machine), K-Nearest Neighbours (KNN), Decision Tree, and K-means Clustering. The study reported notable success rates for these algorithms: 98.7% for Fuzzy C-Means, 97.2% for Naive Bayesian, 96.4% for SVM, 96.6% for KNN, 95.6% for Decision Tree, and 96.7% for K-Means Clustering.[23]

The study in 2012 used the Naive Bayes Classifier, merged with the feature selection method to achieve high performance in four types of attacks on the NSL-KDD dataset. (DoS: 98.7%, Probe: 98.8%, R2L: 96.1%, U2R: 64%). [24]

In a 2013 study, researchers developed a novel architecture by integrating K-Means Clustering and Naïve Bayes Classifier to address the issue of false alarms (where non-threatening data is mistakenly flagged as a threat) in machine learning. This approach was evaluated using the ISCX 2012 Intrusion Detection Evaluation Dataset and demonstrated impressive results, achieving a high-performance rate of 99.8% and a significantly low false alarm rate of 0.13%.[25]

In a comprehensive 2017 study, researchers utilised seven widely used machine learning methods to identify 15 different types of cyber-attacks. The methods included Naive-Bayes (NB), Random Forest (RF), K-Nearest Neighbours (KNN), Multilayer Perceptron (MLP), Adaboost, ID3, and Quadratic Discriminant Analysis (QDA). They employed the CICIDS2017 dataset for this analysis. The performance ratios achieved were: Naive-Bayes at 84%, KNN at 96%, RF at 97%, MLP at 76%, Adaboost at 77%, ID3 at 98%, and QDA at 92%.

To briefly summarize, over the past decade, there have been numerous studies focused on network security using anomaly-based detection methods. These studies have utilised a diverse array of datasets including KDD99[4], NSL-KDD [6], ISCX 2012[10], and CICIDS2017[12], along with various machine learning algorithms like Random Forest[,]. This research reflects the technological advancements in network security, leading to innovative solutions against emerging cyber threats.

This current study aligns with the methodology of Sharafaldin et al[] for several reasons:

- Recency: It utilises recent advancements, making it relevant to current technological contexts.
- Updated Dataset: The dataset used is contemporary and includes a wide range of protocols for generating normal data.
- Comprehensive Attack Spectrum: It encompasses a broad and up-to-date range of cyber-attack types.
- Diverse Machine Learning Techniques: The reviewed studies, employ a wide variety of machine learning methods, indicating a comprehensive approach to anomaly detection.
- The study aims to offer solutions that are up-to-date and effective in addressing the current challenges in network anomaly detection, reflecting the ongoing evolution in the field of network security.

Methodology

Tools and methods

Software platform used:

- **Python**
- **pandas**
- **sklearn**
- **NumPy**

Python is an open-source object-orientated programming language, which is known for its simplicity. its syntax makes code writing and analysis straightforward. some benefits to Python are many resources such as books and web pages make it easily accessible, and it is compatible with numerous libraries that would with machine learning applications Python3.11.5 has been chosen for this work due to these advantages, making it a preferred language for machine learning tasks. Python's user-friendly nature and robust ecosystem, including extensive library support, make it an ideal choice for developing machine learning applications.
Pandas, a robust library for data analysis, operates within the Python environment. It's particularly adept at handling large datasets, simplifying tasks like data filtering, and the efficient modification of columns and rows. These features make Pandas an invaluable tool, hence its selection for use in this project.
Scikit-learn, a library for machine learning, is designed to be used in conjunction with Python. It provides users with a broad array of machine-learning algorithms, making it an important tool in the field. The library is known for its comprehensive documentation and includes all the necessary algorithms required for this project, offering an extensive range of functionalities for various machine-learning applications.

NumPy[], a Python library that allows you to perform mathematical and logical operations quickly and easily, has been used in calculations in this work.

Hardware platform

One important metric for assessing machine learning algorithms is their execution time. However, it's important to note that execution time can vary significantly based on the computer's performance where the algorithms are run. For this reason, sharing the technical specifications of the computer used during the implementation phase is essential. This includes detailing the hardware and system configuration of the computer utilised in the application process.

Model Name:	MacBook Pro
Model Identifier:	MacBookPro18,3
Chip:	Apple M1 Pro
Total Number of Cores:	10 (8 performance and 2 efficiency)
Memory:	16 GB

Performance and evaluation methods [26]

This study's outcomes are assessed using four key metrics: accuracy, precision, f-measure, and recall. Each of these metrics' ranges from 0 to 1, with values closer to 1 indicating higher performance and values near 0 indicating lower performance.

Accuracy: The ratio of successfully categorized data to total data [26].

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

Precision: The ratio of successfully classified data as the attack to all data classified as the attack [26].

$$\text{Precision} = \frac{TP.}{FP+TP}$$

Recall (Sensitivity): The ratio of data classified as an attack to all attack data [26].

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

F-measure (F-score/F1-score): The harmonic-mean of sensitivity and precision. This concept is used to express the overall success[]. so, in this study, when analysing the results, it will be focused, especially on the F1 Score.

$$\text{F- measure} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$$

In calculating these four items, the four values summarized below are used:

- **TP**: True Positive (Correct Detection) The attack data classified as attack.
- **FP**: False Positive (Type-1 Error) The benign data classified as attack.
- **FN**: False Negative (Type-2 Error) The attack data classified as benign.
- **TN**: True Negative (Correct Rejection) The benign data classified as benign.

This distribution is presented by visualizing the Confusion matrix in Figure 5,

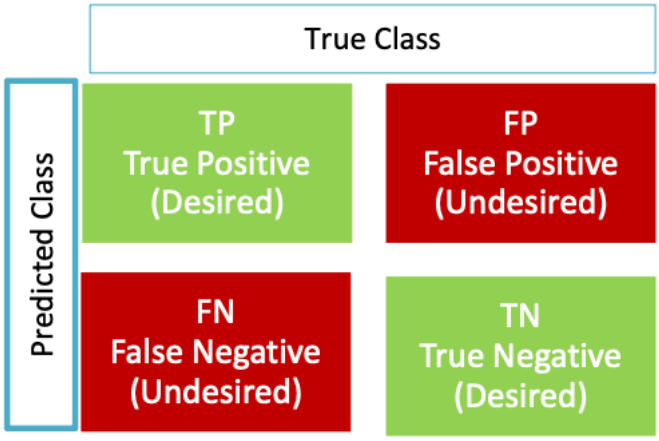


Figure 5

Beyond these four evaluation metrics, processing time has also been included as a crucial factor in algorithm selection, although it isn't traditionally considered a measure of success.

Implementation

The implementation process starts by cleaning the dataset to remove any inconsistencies or errors. the next step after the data set is cleansed is to split the data into training and testing. this is then followed by feature selection, removing any unnecessary features to make the dataset easier to read and run the machine learning algorithm. The final stage is to implement the machine learning algorithm so that it can detect anomalies. Figure 6 below illustrates the entire process.

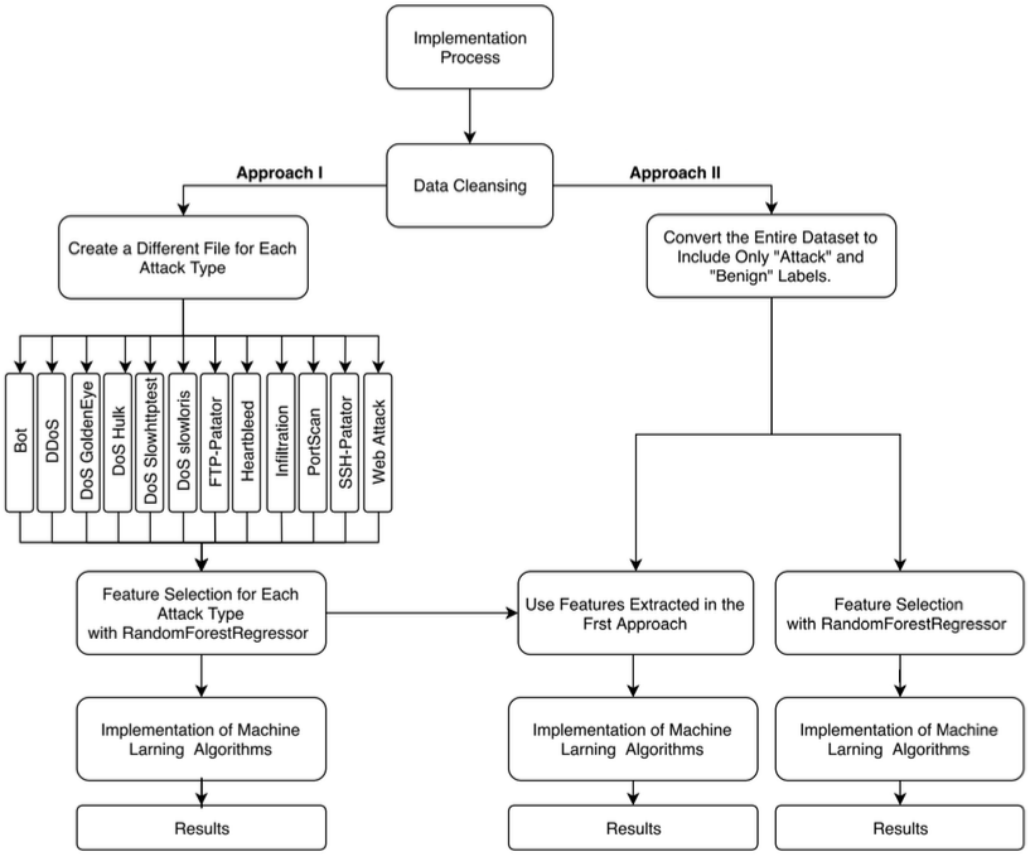


Figure 6. Implementation process

Data pre-processing/ cleansing

The dataset often requires modifications before it is used for implementation making it more efficient. In this case Friday morning CICIDS2017 dataset is corrected as some data is edited.

the dataset file contains 191,033 stream records, The first step in the pre-processing process will be to delete any unnecessary records.

In the dataset, an issue was identified with the feature columns; the dataset file includes 86 columns describing various flow properties like Flow ID, Source IP, Source Port, etc. Notably, a redundancy was discovered in the 'Fwd Header Length' feature (representing the total bytes used in the forward direction of data flow), which was duplicated in both the 41st and 62nd columns. To resolve this, the duplicate column (column 62) was removed, therefore correcting the error, and streamlining the dataset.

Another necessary modification in the dataset involves transforming properties that contain categorical and string values (such as Flow ID, Source IP, Destination IP, Timestamp, and External IP) into numerical form for compatibility with machine learning algorithms. This conversion can be efficiently achieved using the Label Encoder () function from Sklearn's library. By doing so, various string values that are initially incompatible with machine learning operations are assigned integer values ranging from 0 to n-1, rendering them more suitable for algorithmic processing.

However, although the “Label” tag is a categorical feature, no changes have been made on it. The reason is that during the processing, the original categories are needed to classify the attack types in different forms and to try different approaches.

Finally, some minor structural changes should be made to the dataset, including:

"Flow Bytes/s", "Flow Packets/s" features include the values "Infinity" and "NaN" in addition to the numerical values, which can be modified to -1 and 0 respectively to make them suitable for machine learning algorithms.

Creation of training and testing

In machine learning, data plays a crucial role in both training the algorithm and evaluating its capabilities. The training component involves using a specific set of data to develop and refine the algorithm's ability to make predictions or decisions. Once trained, the algorithm's effectiveness is then assessed by applying it to a separate set of test data. The algorithm's performance is essentially evaluated by how accurately it can handle and process this test data, reflecting its overall proficiency in machine learning tasks.

The CICIDS2017 dataset, utilised in this study, initially comes as a single dataset rather than having separate sets for training and testing. As a result, it's necessary to divide it into two distinct parts: one for training the algorithm and the other for testing its efficacy. For this division, the Sklearn command 'train_test_split' is employed, which segments the data into two groups based on specifications set by the user. Common practice, and the approach followed here, is to allocate 20% of the data for testing and the remaining 80% for training. Additionally, 'train_test_split' randomly assigns data to each group, ensuring diverse and representative samples for both training and testing. This methodology is part of a process known as cross-validation, which is critical for assessing the true performance of the machine learning model.

Feature selection

The features in the data set are evaluated to find out which features are important to define a botnet attack. The table below gives a detailed description of each feature.

In this work, the Random Forest Classifier from Sklearn is employed to determine the importance weights of various features. This algorithm constructs a 'decision forest' where each feature within the model is assigned an importance weight, indicating its usefulness in building the decision trees. Upon completion of the process, these feature importance weights are compared and ranked. The aggregate of all feature weights constitutes the overall importance within the decision tree, providing insights into the relative significance of each feature.

However, when calculating these weights, 8 specific features among the 85 in the dataset—Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp, External IP—should be excluded from consideration. This is because these features, while traditionally used, might not be reliable indicators in this context. For instance, attackers might avoid well-known ports to evade detection or use fake IP addresses. Additionally, the dynamic use of ports and the transmission of various applications over the same port can render port numbers an unreliable metric.

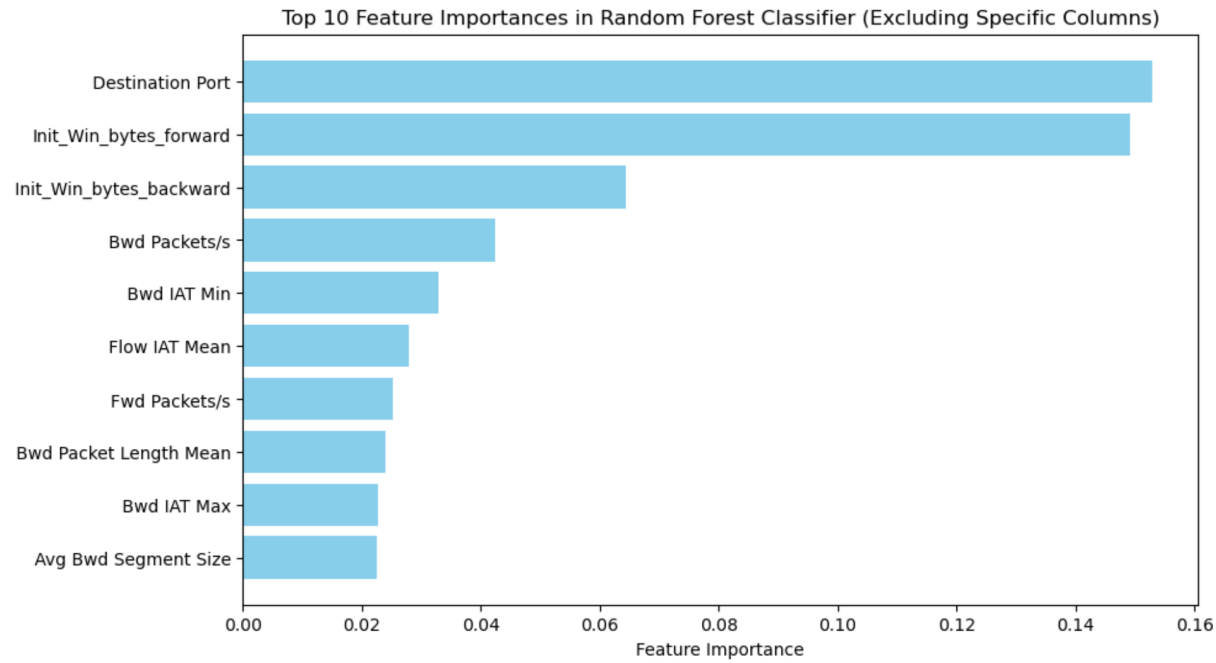
Therefore, in assessing the importance of attributes for detecting attacks, it's more effective to omit potentially misleading features like IP addresses and port numbers. Instead, focusing on more generic and invariant attributes provides a clearer understanding of the data's nature and whether it represents an attack. This approach ensures a more accurate and relevant analysis by emphasizing features that more reliably indicate anomalous activity.

In this work, the Random Forest Classifier from Sklearn is employed to determine the importance weights of various features. This algorithm constructs a 'decision forest' where each feature within the model is assigned an importance weight, indicating its usefulness in building the decision trees. Upon completion of the process, these feature importance weights are compared and ranked. The aggregate of all feature weights constitutes the overall importance within the decision tree, providing insights into the relative significance of each feature.

However, when calculating these weights, 8 specific features among the 85 in the dataset—Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp, External IP—should be excluded from consideration. This is because these features, while traditionally used, might not be reliable indicators in this context. For instance, attackers might avoid well-known ports to evade detection or use fake IP addresses. Additionally, the dynamic use of ports and the transmission of various applications over the same port can render port numbers an unreliable metric.

Therefore, in assessing the importance of attributes for detecting attacks, it's more effective to omit potentially misleading features like IP addresses and port numbers. Instead, focusing on more generic and invariant attributes provides a clearer understanding of the data's nature and whether it represents an attack. This approach ensures a more accurate and relevant analysis by emphasizing features that more reliably indicate anomalous activity.

Below in figure 8 are the feature selection results showing the top ten features importance’s




```
Top four most important features (excluding specific columns):
 Destination Port: 0.1530
Init_Win_bytes_forward: 0.1492
 Init_Win_bytes_backward: 0.0643
 Bwd Packets/s: 0.0423
```

Figure 8

FETURES TABLE 2

№	Feature Name	Feature Description
1	Flow ID	Flow ID
2	Source IP	Source IP
3	Source Port	Source Port
4	Destination IP	Destination IP
5	Destination Port	Destination Port
6	Protocol	Protocol
7	Timestamp	Timestamp
8	Flow Duration	Duration of the flow in Microsecond
9	Total Fwd Packets	Total packets in the forward direction
10	Total Backward Packets	Total packets in the backward direction
11	Total Length of Fwd Packets	Total size of packet in forward direction
12	Total Length of Bwd Packets	Total size of packet in backward direction
13	Fwd Packet Length Max	Maximum size of packet in forward direction
14	Fwd Packet Length Min	Minimum size of packet in forward direction
15	Fwd Packet Length Mean	Mean size of packet in forward direction
16	Fwd Packet Length Std	Standard deviation size of packet in forward direction
17	Bwd Packet Length Max	Maximum size of packet in backward direction
18	Bwd Packet Length Min	Minimum size of packet in backward direction
19	Bwd Packet Length Mean	Mean size of packet in backward direction
20	Bwd Packet Length Std	Standard deviation size of packet in backward direction
21	Flow Bytes/s	Number of flow bytes per second
22	Flow Packets/s	Number of flow packets per second
23	Flow IAT Mean	Mean length of a flow
24	Flow IAT Std	Standard deviation length of a flow
25	Flow IAT Max	Maximum length of a flow
26	Flow IAT Min	Minimum length of a flow
27	Fwd IAT Total	Total time between two packets sent in the forward direction
28	Fwd IAT Mean	Mean time between two packets sent in the forward direction
29	Fwd IAT Std	Standard deviation time between two packets sent in the forward direction
30	Fwd IAT Max	Maximum time between two packets sent in the forward direction
31	Fwd IAT Min	Minimum time between two packets sent in the forward direction
32	Bwd IAT Total	Total time between two packets sent in the backward direction
33	Bwd IAT Mean	Mean time between two packets sent in the backward direction
34	Bwd IAT Std	Standard deviation time between two packets sent in the backward direction
35	Bwd IAT Max	Maximum time between two packets sent in the backward direction
36	Bwd IAT Min	Minimum time between two packets sent in the backward direction
37	Fwd PSH Flags	Number of packets with PUSH
38	Bwd PSH Flags	Number of times the PSH flag was set in packets travelling in the backward direction (0 for UDP)
39	Fwd URG Flags	Number of times the URG flag was set in packets travelling in the forward direction (0 for UDP)
40	Bwd URG Flags	Number of times the URG flag was set in packets travelling in the backward direction (0 for UDP)
41	Fwd Header Length	Total bytes used for headers in the forward direction
42	Bwd Header Length	Total bytes used for headers in the backrward direction
43	Fwd Packets/s	Number of forward packets per second
44	Bwd Packets/s	Number of backward packets per second

conclusion

As we conclude the current phase of our study, we have successfully navigated through the crucial steps of data pre-processing, cleansing, and feature selection using the CICIDS2017 dataset. Our journey so far has laid a solid foundation for the application of machine learning algorithms, which is the next significant step in our research.

The upcoming phase of implementing machine learning algorithms is pivotal. It involves applying various algorithms like Random Forest, and K-Nearest Neighbours, to the prepared dataset. This step is critical as it will enable us to evaluate the effectiveness of these algorithms in accurately detecting network anomalies based on the features selected.

Post-implementation, a thorough analysis of the results will be essential. We will evaluate the algorithms' performance based on accuracy, precision, recall, and other relevant metrics. This analysis will not only shed light on the effectiveness of each algorithm but also highlight areas for further improvement.

In conclusion, while significant progress has been made in the initial stages of this study, the forthcoming implementation of machine learning algorithms and the subsequent analysis of their performance remain. These next steps are crucial in advancing our understanding of network anomaly detection and enhancing the effectiveness of cybersecurity measures.

references

[1] https://www.statista.com/statistics/273018/number-of-internet-users-worldwide/

[2] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, 2005, pp. 333-342: Australian Computer Society, Inc.

[3] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a reliable intrusion detection benchmark dataset," *Software Networking,* vol. 1, no. 1, pp. 177- 200, 2017.

[4] "KDD Cup 1999 Data," *University of California, Irvine,* [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. [Accessed 05 Aug 2018].

[5] A. Özgür and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," *PeerJ PrePrints,* vol. 4, p. e1954v1, 2016.

[6] "NSL-KDD dataset," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: http://www.unb.ca/cic/datasets/nsl.html. [Accessed 06 Aug 2018]

[7] A. Özgür and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," *PeerJ PrePrints,* vol. 4, p. e1954v1, 2016.

[8] "NSL-KDD dataset," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: http://www.unb.ca/cic/datasets/nsl.html. [Accessed 06 Aug 2018].

[9] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on*, 2009, pp. 1-6: IEEE.

[10] "Intrusion detection evaluation dataset (ISCXIDS2012)," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: http://www.unb.ca/cic/datasets/ids.html. [Accessed 07 Aug 2018].

[11] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *computers & security,* vol. 31, no. 3, pp. 357-374, 2012.

[12] "Intrusion Detection Evaluation Dataset (CICIDS2017)," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: http://www.unb.ca/cic/datasets/ids-2017.html. [Accessed 08 Aug 2018].

[13] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications,* vol. 60, pp. 19-31, 2016.

[14] W. Stallings, L. Brown, M. D. Bauer, and A. K. Bhattacharjee, *Computer security: principles and practice*. Pearson Education, 2012.

[15]M. Chhabra, B. Gupta, and A. Almomani, "A novel solution to handle DDOS attack in MANET," *Journal of Information Security,* vol. 4, no. 03, p. 165, 2013.

[16]"Ares," *GitHub, 08-Dec-2017.* [Online]. Available: https://github.com/sweetsoftware/Ares. [Accessed: 12 Aug 2018].

[17] A. Sperotto, *Flow-based intrusion detection*. Citeseer, 2010

[18]A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.", 2017.

[19] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in*

[20] L. Breiman, "Random forests," *Machine learning,* vol. 45, no. 1, pp. 5-32, 2001.

[21] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Computers & security,* vol. 24, no. 4, pp. 295-307, 2005.

[22] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," *The VLDB journal,* vol. 16, no. 4, pp. 507-521, 2007.

[23]M. Suresh and R. Anitha, "Evaluating machine learning algorithms for detecting DDoS attacks," in *International Conference on Network Security and Applications*, 2011, pp. 441-452: Springer.

[24] S. Mukherjee and N. Sharma, "Intrusion detection using naive Bayes classifier with feature reduction," *Procedia Technology,* vol. 4, pp. 119-128, 2012.

[25] W. Yassin, N. I. Udzir, Z. Muda, and M. N. Sulaiman, "Anomaly-based intrusion detection through k-means clustering and naives bayes classification," in *Proc. 4th Int. Conf. Comput. Informatics, ICOCI*, 2013, no. 49, pp. 298-303.

[26]M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *Ieee communications surveys & tutorials,* vol. 16, no. 1, pp. 303-336, 2014

3. Appendices

Appendices may contain source code, test data, etc. It is usual to letter these appendices by A,B,C ... and each Appendix must have an appropriate title.
The following should be included in the report:

- Appendix A. The original specification of your project, as agreed with your supervisor.
- Appendix B. A statement about the relevance of your project to the degree course you are doing.
- Appendices C, D, E,... appendices required by your project work: extra screen shots, flow charts, UML diagrams, workflows, data etc.

4. Regulations

The Report shall constitute an ordered, critical exposition of knowledge in an approved field falling within the subject matter of the course and it should afford evidence of reasoning power and

knowledge of the relevant literature. It must be in English and presented in a satisfactory literary form.
The general University regulations concerning submission of reports are found by clicking Submission of assessed work.
NB. When you have completed the report, and any time you updated its content, save it, then go to the Table of Contents at the top, right click, and choose 'Update Field' – 'Update Entire Table' and click on 'Ok'. Then delete this instruction and Save.