# THE POSSIBILITIES OF REAL-TIME CYBERATTACK DETECTION THROUGH MACHINE LEARNING METHODS

**ALDO CANI[1], AURORA SIMONI[2]**

[1]The Directorate of Information Technology, Special Prosecution Office against Corruption and Organized Crime, Tirana, Albania

[2]Department of Applied Mathematics, Faculty of Natural Sciences, University of Tirana, Tirana, Albania

e-mail: aldocani92@gmail.com

***Abstract***

*Cyberattacks are already a threat that continues to grow every day for institutions, universities, organizations or even simple Internet users. Attacks that succeed often cause data deletion, encryption and denial of access causing huge losses which in many cases are unrecoverable. Intrusion Detection System (IDS) technologies can help to detect attacks but in the case of sophisticated cyberattacks, they do not give the expected results because these types of attacks require new technologies mainly based on artificial intelligence. A promising method for preventing attacks is the Machine Learning (ML) method, which has achieved very good results for identifying and preventing cyber security threats. These methods improve cyber security based on data traffic analysis. To achieve high results, ML algorithms must be trained with large datasets in order to identify patterns of cyberattack behavior. This process enables them to detect attacks in real time, helping to prevent the possible consequences of the attack. In this paper, we will discuss the attacks and their types, and we will also perform machine learning identification when these attacks occur. We will analyze their detection methods as well as analyze the best algorithms for detecting attacks in real time. The study is a valuable resource for those who wish to deeply understand these attacks and their detection through machine learning.*

***Key words:*** *Cyber Attacks, machine learning, attack detection, classification, object identification, cyber security.*

***Përmbledhje***

*Sulmet kibernetike tashmë janë një kërcënim që vazhdon të rritet çdo ditë për*

*institucionet, universitetet, organizatat apo edhe për përdoruesit e thjeshtë të internetit. Sulmet që arrijnë të jenë të suksesshme shpesh shkaktojnë fshirje të të dhënave, enkriptim dhe mohimin e aksesit, duke shkaktuar humbje të mëdha që në shumë raste janë të parekuperueshme. Teknologjitë e Sistemeve të Zbulimit të Ndërhyrjeve (IDS) mund të ndihmojnë në zbulimin e sulmeve, por në rastin e sulmeve kibernetike të sofistikuara, ato nuk japin rezultatet e pritura, pasi këto lloj sulmesh kërkojnë teknologji të reja kryesisht të bazuara në inteligjencën artificiale. Një metodë premtuese për parandalimin e sulmeve është metoda e Machine Learning (ML), e cila ka arritur rezultate shumë të mira për identifikimin dhe parandalimin e kërcënimeve të sigurisë kibernetike. Këto metoda përmirësojnë sigurinë kibernetike bazuar në analizën e trafikut të të dhënave. Për të arritur rezultate të larta, algoritmet e ML duhet të trajnohen me grupe të mëdha të dhënash për të identifikuar modelet e sjelljes së sulmeve kibernetike. Ky proces u mundëson atyre të zbulojnë sulmet në kohë reale, duke ndihmuar në parandalimin e pasojave të mundshme të sulmit. Në këtë punim, ne do të diskutojmë sulmet dhe llojet e tyre, dhe gjithashtu do të realizojmë identifikimin e tyre përmes ML kur këto sulme ndodhin. Do të analizojmë metodat e zbulimit të tyre si dhe do të shqyrtojmë algoritmet më të mira për zbulimin e sulmeve në kohë reale. Studimi është një burim i vlefshëm për ata që dëshirojnë të kuptojnë më thellë këto sulme dhe zbulimin e tyre përmes metodave Machine Learning.*

***Fjalët kyçe:*** *Sulme kibernetike, machine learning, zbulimi i sulmeve, klasifikimi, identifikimi i objekteve, siguri kibernetike.*

## Introduction

Internationally, cyberattacks now constitute a significant risk for both people and organizations as well as for government institutions. Such attacks end up expensive and interfering with data loss, etc. Today's advanced cyber-attacks are no longer being caught easily by traditional IDS which relies on signature to detecting the anomaly.

Machine Learning (ML) as an approach for detecting cyberattacks looks very promising. Through training massive sets of known attacks versus legitimate traffic data, ML algorithms can learn the characteristics of malicious activity. This enables detection of novel signatures of new malware (and previously unknown attacks).

This paper will talk about how to use ML in real-time cyberattacks detection. Various types of Machine Learning (ML) Algorithms can be used to perform

this task and we'll revisit its challenges as well as opportunities (Tavallaee et al., 2009).

**Literature and dataset**

We used the NSL-KDD dataset which is the first publicly available dataset of multiple network traces with diverse intrusions (Tavallaee et al., 2009). It involves KDD'99, an adapted version of the data set used in the 1999 Knowledge Discovery and Data Mining (KDD) Cup competition (Hettich & Bay, 1999). This NSL-KDD dataset is alleviated with redundant and duplicated records in test sets compared to KDD'99 (Tavallaee et al., 2009).

The NSL dataset is comprised of 41 features extracted from network traffic logs. The features are divided into four categories (Tavallaee et al., 2009).

Basic features: These attributes determine the fundamentals of the network's traffic that are Protocol, source-IP-address, destination-IP-address.

Content features: These features are about what is transmitted in the network traffic (e.g., how many bytes have been sent, how much data has been sent.

Time features: These characteristics capture the temporal aspect of the network traffic, i.e., the length of time for the connection and how many connections per second.

Traffic features: These are aggregated statistics about traffic behavior, including number of connections from a specific source Ip address, and number of connections towards one destination Ip address.

The NSL-KDD dataset is divided into two sets: a train dataset and test dataset. The Training set has 125,972 Records and Test Set has 22,540 Records. The training set includes normal AND malicious traffic, while the test set consists of malicious traffic (Tavallaee et al., 2009)

The NSL-KDD dataset is very popular for research since it is huge in size, contains various features (e.g., 9,053 features) and comes with good annotation information (Tavallaee et al., 2009). This dataset is used to evaluate several machine learning schemes for NID."

NSL-KDD dataset can be used for NID (Network Intrusion Detection), information security, and cybersecurity research:

We can prepare and test SVMs (support vector machines), decision trees and random trees using this data set on NID.

Anomaly detection: This work demonstrates an example of how the NSL-KDD dataset can be utilized to create anomaly detection algorithms with detecting malicious network traffic as an end goal. Anomaly detection algorithms detect network traffic outside the normal traffic pattern.

Botnet detection: The NSL-KDD dataset can be employed to design botnet detection algorithms as botnets in the network traffic (Gelenbe & Nakip, 2023 ). These are the collections of infected PCs utilized for assaults, for example, DoS attacks and smash missions.

Denial-of-service attack detection: Denial of service attack detection system can be developed with using NSL-KDD dataset (Gelenbe & Nakip, 2023). A denial-of service attack is an attempt to render a computer or network unavailable to its intended users.

Malware detection: This allows you to build malware detection algorithms to detect malware infections in the dataset (NSL-KDD) (Gelenbe & Nakip, 2023).

**Cybersecurity issue**

Cybersecurity problems are worrying people as well as businesses and governments globally. Cybercrime costs approximately $6 trillion in the world on average in 2021 (Morgan, 2017). Victims of cyberattacks can face dire consequences such as financial loss, leaking of personal information, and reputational damages.

Here are some of the most common cybersecurity issues, with statistics to support their prevalence and impact:

*Malware*, malicious software which can harm or cripple computer systems and networks. Kaspersky's systems detected approximately 122 million malicious files in 2022, 6 million more than last year. (Kaspersky, 2022) (e.g. Malware causes a lot of damage such as data loss, computer crashing, and monetary loss.

*Phishing*, an attack wherein the attacker attempts to deceive the users into surrendering confidential data, e.g., passwords, credit card numbers, etc. Phishing According to Accenture's Cyber Threatscape - Autumn & Winter Report (2021), phishing attacks represent 38% of the total number of data breaches in 2022 (Burbidge, 2022). Phishing is a very potent attack strategy because it preys on the weakness of humans — they trust and often crave riches.

*Denial-of-service (DoS) attacks:* Denial of service (DoS) attack is an attempt to make the computer or any network unavailable for the purpose it's made for. As of 2022, there were more than 10 million DoS attacks (Stamford, 2023) "DoS attacks can negatively affect the operations of a business or organization causing downtime and losses.

*Man in the middle attacks*: Man in the middle attacks, where an intermediary interrupts the communication between two separate parties. Man in the middle attacks are responsible for approximately 15% of 2022's data breaches (Burbidge, 2022). Man in the middle attacks can be hard to detect as they are often completely unbeknownst to the victims.

*Supply chain attacks*, a supply-chain means an attack in which the attacker wants to exploit one of a company's Suppliers in order to get to the organization. The percentage of supply chain attacks responsible for 22% of all data breaches in 2022 (Burbidge, 2022). Third party suppliers are now more important to businesses than ever, and supply chain attacks continue to escalate in severity.

*Zero-day attacks*, Zero-day exploit where an attacker exploits a vulnerability in a program whose vendor is unaware of its existence. Since there is no fix for vulnerability these attacks can't be patched and hence, they are difficult to defend. (NIST, 2024).

*Ransomware attacks*, Ransomware is a class of assaults in which an aggressor performs encryption on all or some portion of the casualty's information and solicits an expense (Ransom) from the casualty as a trade-off to Ransomware attacks may cause significant damages to businesses and organizations in ways of costs and delays. (CISA, 2021)

*Watering hole attacks*: The Attacker targets websites visited frequently by the targeted user pool. Victim lands on this injection when visits the hacked site. (Burbidge, 2022).

*Advanced persistent threats* (APTs): APTs are highly sophisticated attacks executed by sponsored and organized teams of experts. In contrast to exploits in vulnerabilities, APTs focus on specific organizations or groups of people and often go unnoticed for months, or even years. (Stamford, 2023)

*Cryptojacking*, a term for cryptocurrency mining by illegitimate means using malicious code to benefit from their victim's devices' resources. Cryptojacking will significantly degrade the performance and drain a large amount of power from the victim's computer. (Tekiner et al., 2021).

*Deepfakes*: What we mean by "deep fake" is a deep-fake video where even though it appears to be someone speaking for him and saying the things he didn't, and/or the actor (b) a deep-fake voice clip: the same actor sounds like another the tech can be abused to disseminate fake news, malign folks or even perpetuate scams. (Abbas & Taeihagh, 2024).

*Social engineering attacks:* Social engineering attacks is the type of attack where attacker abuses victim's behavior using mind game strategy to discard personal information while executing an act that compromises security. Why social engineering is so effective: It's human nature to want to help, and most of us are predisposed to believe other people. (CISA, 2021).

**Responding to cyberattacks**

According to Gartner Research firm's report, the global cybersecurity market size will be USD$170.4 billion by 2023 (Stamford, 2023). These companies have grown due to two reasons, firstly, we have a lot of cracking heads out there who do their job quite perfectly for which there is an amount, and secondly, just sheer paranoia in sharing any data on the internet.

With the growing threat landscape and sophistication of the number and type of attacks on systems, ML is gaining huge importance in the cybersecurity space. After all, research from IBM (Consulting, 2011) tells us that, 79 percent of those currently leveraging ML for enhancing its corporate security position.

ML is being used in a variety of ways to improve cybersecurity, including:

*Detecting and blocking malware*, Using trained malicious model samples and behavior during testing phase. Helpful in discovering and eliminating the malicious code (malware) when it's still not harming something. According to Forrester research report, Machine learning solutions have the ability to identify 99% of new malware compared with 70%, where the latter based on Signature-based detection techniques.

*Detecting and preventing intrusion*, this you can additionally utilize with the ML algorithm in order to detect breaches in computer system security and/or network security. It prevents attacks such as DDoS attack and man in the middle attack. Using ML in intrusion detection systems, 95 per cent of attacks are recognized and where rules-based systems come into play, 85 per cent get caught, according to Gartner.

*Analyzing security data*, Trend analysis can be used with these security events using ML algorithms for processing of high-volume datasets. This allows

security analysts to find and react to threats in quick time. According to IDC, using (ML) Security analytics (solution) you can shorten the identification of an attack up to 1/2 (half). (Ballou, 2022).

*Automating security tasks,* Automation using ML algorithms is also possible for numerous other security functions (e.g., patching vulnerabilities, setting up firewall filters, etc.). And this can allow security analysts to be more productive working on more advanced things. According to a report from McKinsey, this is what machine learning (ML)-based cybersecurity can do for automating cybersecurity operations, reducing up to 20% in costs (Greis& Sorel, 2024).

Cybersecurity can benefit a lot from using the power of ML. As ML tech progresses, we expect to witness more exciting techniques using ML for safeguarding our systems and their data against malicious cyber-attacks.

**Intrusion detection system (IDS)**

An Intrusion Detection System (IDS) is a monitoring, alert analysis and response, policy-based application in software or hardware form that searches patterns of illicit activity or rule breaking on network traffic or host. An IDS can detect various types of intrusions including malware infection, DoS, and attempted unauthorized accesses (NIST, 2001).

The main functionality of an IDS depends on analyzing System Logs data (logs), as well as Network Traffic data (packets) to detect oddities representing intrusion signatures (threats). (NIST, 2001) It could detect anomalies which would be things like abrupt changes in network usage or strange activity or suspicious traffic or repeated attempted logins into systems that probably shouldn't be messed around with.

When an IDS identifies something unusual, it creates an alert or stops the traffic or does what is required to handle the threat. (NIST, 2001). They collaborate easily with other secure apparatuses (as an example firewall, Network IPS).

There are two main types of IDSs: such as network intrusion detection systems (NIDSs) and host intrusion detection systems (HIDSs) (NIST, 2001).

Most NIDS nodes are placed in important places within a network; typically, near the firewalls or edge nodes. Generally, HIDSs run on single servers/workstations.

There is a plethora of ways that IDS can detect suspicious behavior; here,

however, I will focus on the most common methods that are used. Some common techniques include,

*Signature - based detection:* A signature-based detection involves comparing network traffic or system logs against a predefined list of characteristics or footprints of well-known attacks. If the hash from this block of data shows a match, IDS considers that a hacker attack is happening.

*Anomaly-based detection:* Anomaly detection looks for deviation from the standard behavior of network traffic and system activity. Otherwise, if anything unusual traffic is found, the IDS will interpret this as a cyberattack taking place. An IDS can also assist in stopping attacks from progressing if suspicious behavior is identified/alerted upon.

Some cases of how IDSs can be used to improve security: *Detect malware infections,* An IDS can identify malware by monitoring the network to uncover anomalous traffic or system activity. For instance, an IDS may search for out of character network activity which is directed towards a famous malware command and control server.

*Detect denial-of-service attacks,* some of the tools to be used in detecting DoS attacks using IDS includes observing abnormal traffic patterns.

*Detect unauthorized access attempts,* An IDS has the ability to identify any attempted accesses to or from unauthorized applications and services. For instance, an intrusion detection system can search traffic being sent to a server that the user is not allowed to reach, or to traffic from a network whose presence on the server is not allowed to be.

Inherently, as with any security solution, though, there are limitations. IDS easily circumvent for attackers which IDS is in place. For comprehensive protection, IDSs should be used together with other security measures, for example, Firewall and Access Control Lists etc.

**Machine learning algorithms**

Logistic regression (Geetharamania et al., 2021) is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc, but instead of giving the exact value as 0

and 1, it gives the probabilistic values which lie between 0 and 1.

In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

The curve from the logistic function indicates the likelihood of something.

Logistic Regression (Geetharamania et al., 2021) can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. Logistic Regression use Logistic Function (Sigmoid Function), and the equation below show the logistic function:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Sigmoid function is a mathematical function used to map the predicted values to probabilities. It maps any real value into another value within a range of ] 0, 1 [. (Guna sree et al.,2021)

The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the sigmoid function or the logistic function.

In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

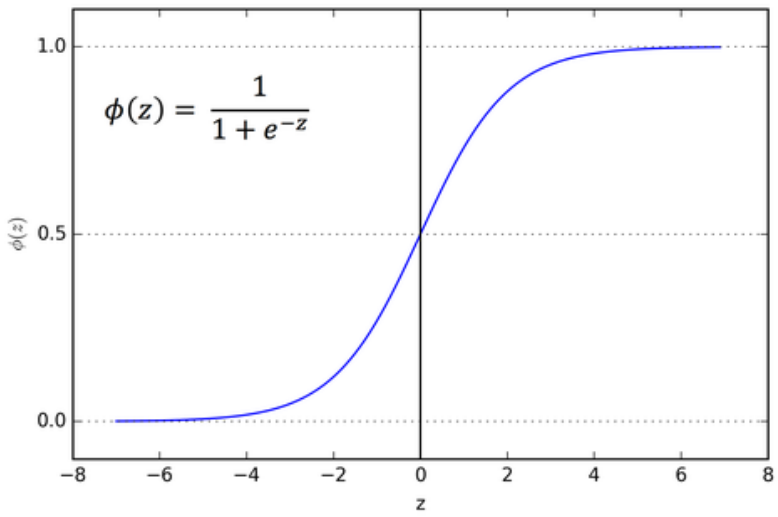

**Figure 1.** Graphic and formula for sigmoid function. Source: Image Medium.com

The Logistic regression equation can be obtained from the Linear Regression equation. (Guna sree et al., 2021)

## Application of various functions and methods on the dataset

As mentioned in the introduction, dataset "KDDTrain+.txt" was used to carry out this study.

The relevant commands were created to import the necessary Python libraries to apply the methods on the dataset, the dataset contains 125972 rows and 43 columns according to the picture below.

**Table 1.** Dataset View

| | 0 | tcp | ftp_data | SF | 491 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 0.10 | 0.11 | 0.12 | 0.13 | 0.14 | 0.15 | 0.16 | 0.18 | 2 | 2.1 | 0.00 | 0.00.1 | 0.00.2 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | udp | other | SF | 146 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 1 | 0.0 | 0.0 | 0.0 | |
| 1 | 0 | tcp | private | S0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 123 | 6 | 1.0 | 1.0 | 0.0 | |
| 2 | 0 | tcp | http | SF | 232 | 8153 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 0.2 | 0.2 | 0.0 | |
| 3 | 0 | tcp | http | SF | 199 | 420 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 30 | 32 | 0.0 | 0.0 | 0.0 | |
| 4 | 0 | tcp | private | REJ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 121 | 19 | 0.0 | 0.0 | 1.0 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 125967 | 0 | tcp | private | S0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 184 | 25 | 1.0 | 1.0 | 0.0 | |
| 125968 | 8 | udp | private | SF | 105 | 145 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 0.0 | 0.0 | 0.0 | |
| 125969 | 0 | tcp | smtp | SF | 2231 | 384 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0.0 | 0.0 | 0.0 | |
| 125970 | 0 | tcp | klogin | S0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 144 | 8 | 1.0 | 1.0 | 0.0 | |
| 125971 | 0 | tcp | ftp_data | SF | 151 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0.0 | 0.0 | 0.0 | |

125972 rows × 43 columns

The entire dataset is checked for normal values and values of different attacks separated by type in the attack column. After that the count of the repetition of each value in the 'attack' column was performed through the function

frequency_attack = dataset['attack'].value_counts()

and the result was printed as below.

**Table 2.** Cyberattack Types and Their Frequencies

| | |
|---|---|
| normal | 67342 |
| neptune | 41214 |
| satan | 3633 |
| ipsweep | 3599 |
| portsweep | 2931 |
| smurf | 2646 |
| nmap | 1493 |
| back | 956 |
| teardrop | 892 |
| warezclient | 890 |
| pod | 201 |
| guess_passwd | 53 |
| buffer_overflow | 30 |
| warezmaster | 20 |
| land | 18 |
| imap | 11 |
| rootkit | 10 |
| loadmodule | 9 |
| ftp_write | 8 |
| multihop | 7 |
| phf | 4 |
| perl | 3 |
| spy | 2 |

Name: attack, dtype: int64

These results are also presented in the graph, where it is noted that the highest number is for normal values, and then come the types of attacks

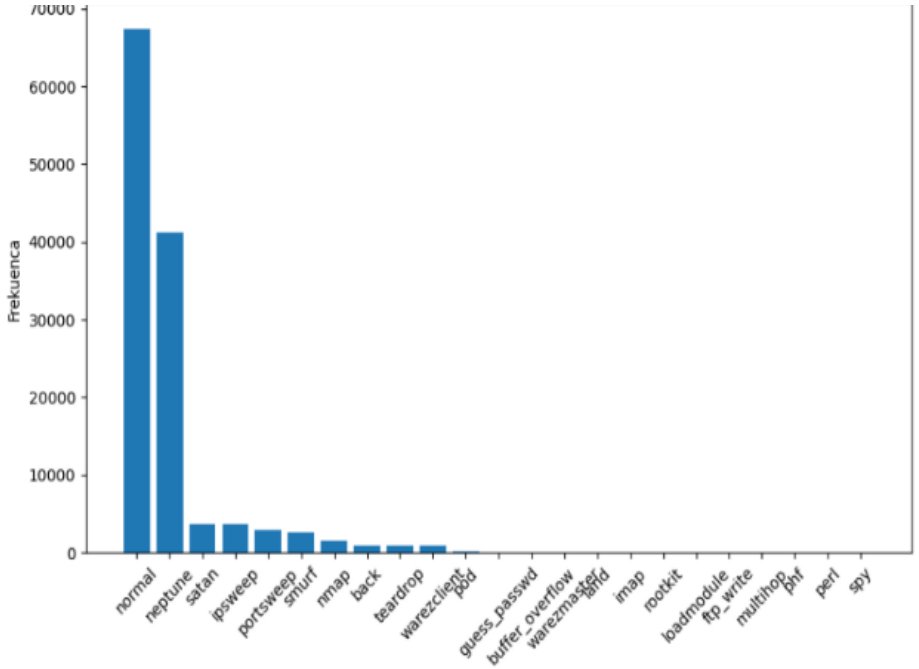


**Figure 2**. Graph of Cyberattack Types and Their Frequencies

In the graph below we make a big separation between attacks and normal traffic cases. The figure shows the graph where normal traffic is marked with 0 and traffic when there are attacks is marked with 1

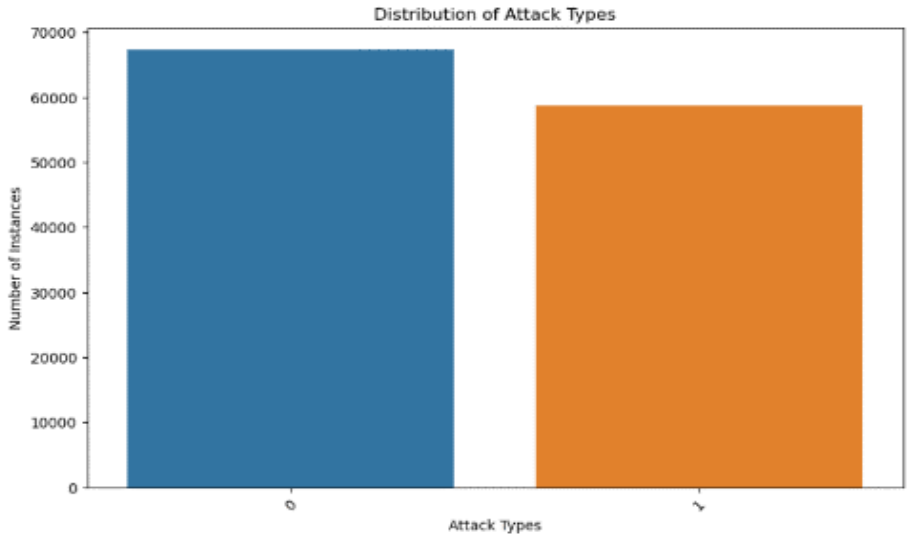


**Figure 3**. Distribution of Attack Types

Also, a graph was created for the distribution of attacks referring to duration and the distribution is clearly seen in cases where there are attacks or not.

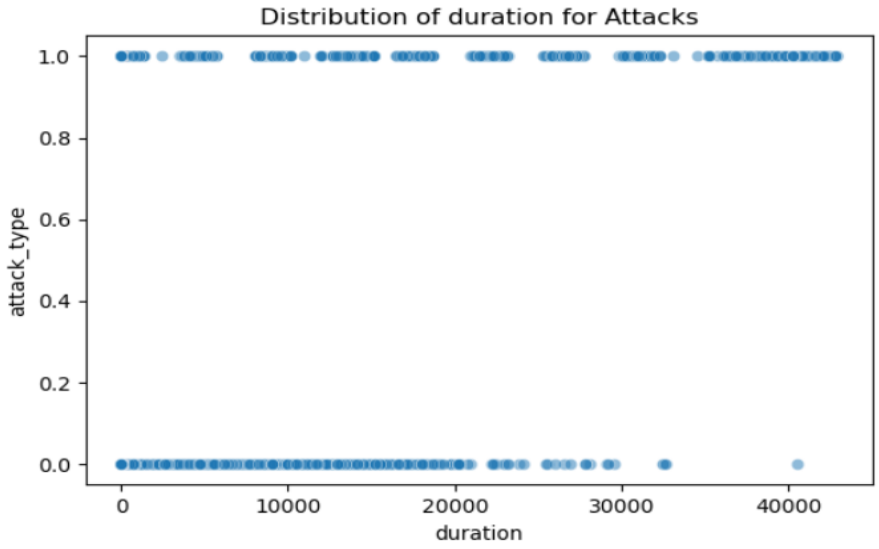


**Figure 4.** Graph of distribution of duration attacks

And to a graph was created for the distribution of attacks, referring to

src_bytes, which also shows the extent of the attacks according to the figure below.

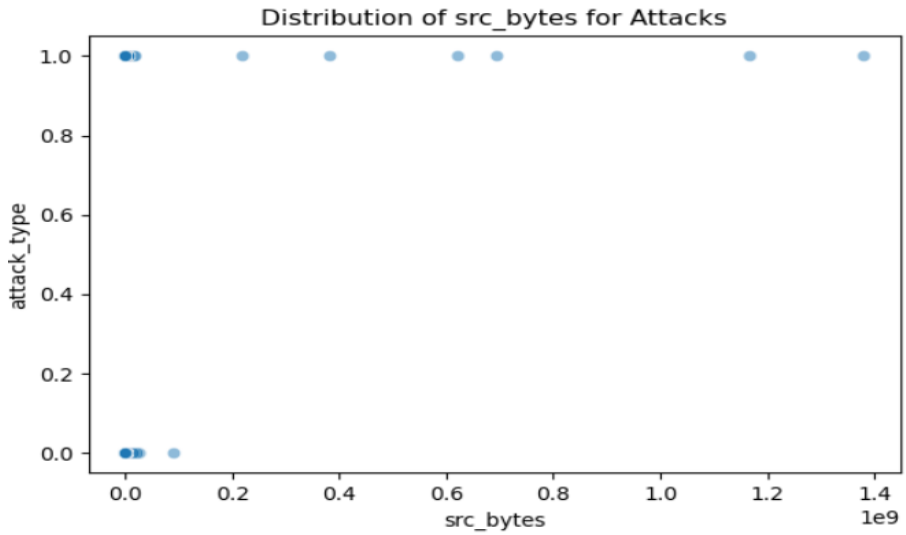


**Figure 5.** Graph of distribution src_bytes for attacks

These graphs show us an overview of our data as well as the classification of data into attacks or normal data.

**Correlation Matrix**

We retrieve the data from dataset, obtain correlate matrix for the numeric columns with dataset, then generate and display the heatmap show the correlation between them. It's a standard thing to know how different variables in a dataset are correlated with one another.

The correlation matrix will display the Pearson's correlations between each column of your dataset. Each entry of the matrix represents the Pearson correlation coefficient for two features. The corr() function calculates these correlation coefficients for all pairs of numerical columns in the DataFrame.

We generated a heatmap of the correlation matrix with seaborn. Heatmap is a graphical presentation of the data representing the elements of the matrix using colors. Here, the correlation coefficients are depicted in colors. Here we'll plot a heatmap using matplotlib's sns.heatmap() command.

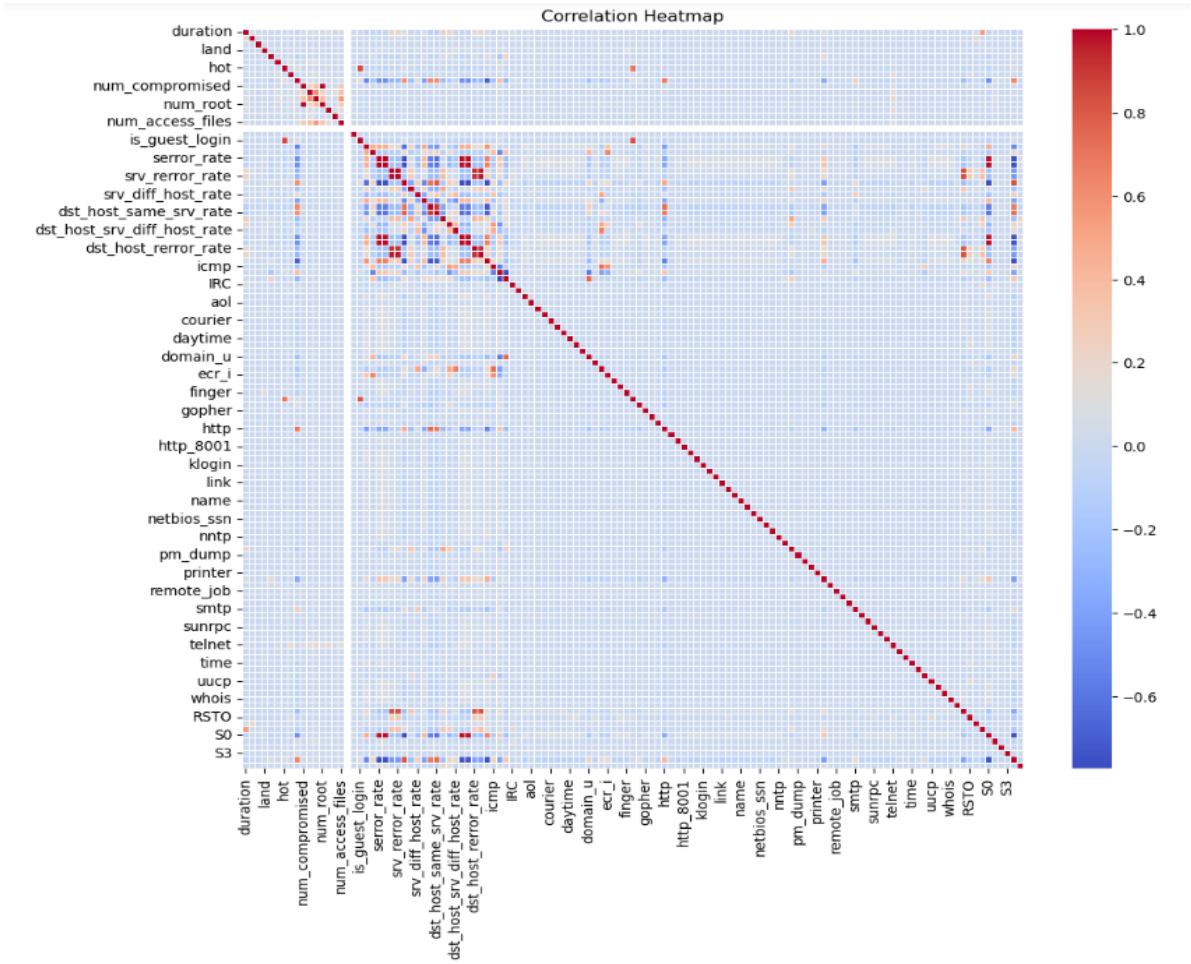


**Figure 6.** Correlation Matrix Heatmap

This heatmap shows the correlation coefficients of the numbers present in the dataset using Pearson correlation measures. The diagonal line portrays the correlations with value one when the features being measured are same.

Positive correlations appear as red areas and the negative correlations as blue areas.

Equally distributed non diagonal elements represent low correlation between most of the variables while high values in certain areas represent strong correlation. Therefore, the heatmap can be helpful in understanding how features are associated, that helps in data analysis and feature engineering.

**Data training**

The actual development of the results can only proceed with training data.

We will build a code which will split the data into the training dataset and the test dataset; we will normalize the data and then apply binary classification to the training dataset using the logistic regression classifier. The effectiveness of the logistic regression model classifier can be evaluated using the model evaluation results. The output is as follows after running the code.

Accuracy: 0.88

Precision: 0.87

Recall: 0.86

F1 Score: 0.87

A comprehensive evaluation was conducted for models Random Forest Classifier, Naive Bayes, Logistic Regression Classifier, and Support Vector Machines (SVM) Classifier. The results were observed as following:

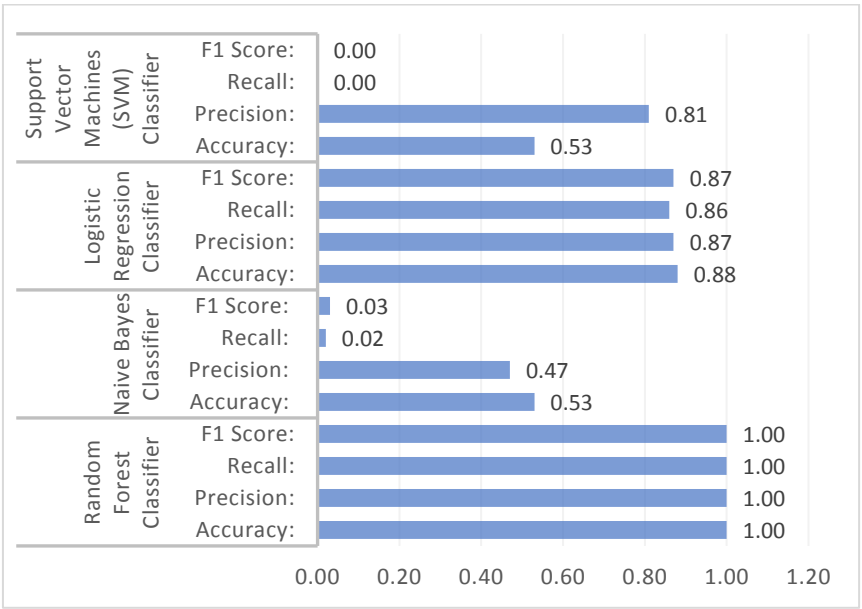


**Figure 7**. Performance Comparison of Machine Learning Classifiers

Analysing the results, we can observe that:

The Random Forest Classifier performed the best in all metrics (Accuracy, Precision, Recall, and F1 Scores) with perfect accuracy. That's because it attains accurate results with great precision in predicting/classifying attacks.

The Naive Bayes Classifier had relatively good accuracy but was terrible on both recall and precision (low F1 Score).

The Logistic Regression Classifier got better overall results (good trade off between precision and recall) resulting in decent F1 Score.

Unfortunately, the support vector machines (SVM), classify has performed very bad in particular for recall and f1 score which means that it is not capable to properly label attacks.

These comparative results show us important clues on the merits and deficiencies of each model which can be useful while deciding on choosing an ideal predictor for particular classification work in the future studies.

**XGBoost Algorithm**

To improve the performance more we will use the mode XGBoost (Chen & Guestrin, 2016), which is one of the most popular and efficient implementations of the Gradient Boosted Trees algorithm, a supervised learning method that is based on function approximation by optimizing specific loss functions as well as applying several regularization techniques. XGBoost has a strong mathematical background

**Performance improvement with XGBoost**

The XGBoost model was used to improve the results displayed by the logistic regression model.
Results after applying XGBoost in our dataset are very interesting because they tend to go to 1.0 which is the best result.

The results are as below

XGBoost Model Results:

Accuracy: 1.00

Precision: 1.00

Recall: 1.00

F1 Score: 1.00

To retrieve the optimal hyperparameters for a machine learning model that has undergone hyperparameter tuning using techniques like Grid Search or Random Search we use function

best_params = grid_search.best_params.

In our dataset, it helps in identifying the hyperparameters that yield the best performance for a specific model. Once these optimal hyperparameters are determined, they can be used to train the final model with the best configuration, ensuring that the model performs optimally for our classification task.

The subsequent line, best_params = grid_search.best_params_,

stores the best hyperparameters in the best_params variable, which can then be printed using print("Best Parameters:", best_params) to display the specific hyperparameter values that resulted in the highest model performance.

Running the following function

*best_params = grid_search.best_params_*

*print("*Best Params*:", best_params)*

we have output results as below:

Best Params:

{'learning_rate':0.2,'max_depth':5,'n_estimators': 300,'subsample': 1.0}

This output shows us,

Learning rate: This parameter controls the step size in updating model weights during training. A lower value like 0.2 leads to slower weight updates, which can help the model converge to a better solution.

max_depth: It determines the maximum depth of each tree in the XGBoost model. A depth of 5 means that the trees are relatively shallow, which can help prevent overfitting.

n_estimators: This parameter specifies the number of trees (estimators) in the XGBoost model. Having more trees can improve model performance, up to a point.

subsample: This parameter controls the fraction of the training data that is randomly sampled and used in each boosting round. A value of 1.0 means that all the data is used, ensuring that the model sees the full training dataset in each iteration

**Verification of the accuracy of the results of the XGBoost model**

Because XGBoost is our best performer so far, we will be checking to confirm that it works on the larger datapoints as well (we could do more validation and model evaluation).

To test the accuracy of the XGBoost results, was runed another code to perform the following steps:

Cross-Validation Assessment: The code uses k-fold cross validation to measure the model performance on various parts of data. The process here is to slice the X-train into some chunks (folds) and use different combinations of them as test and train set in an 80\–20% ratio. As it is an unseen dataset (test set), it provides a better benchmark of how much the model can generalize.

Training-Test Data Split: The dataset is then divided into 2 sets: one for train and another for testing after using the above-mentioned Cross-Validation in scikit-learn. This separation enables one to assess the model’s performance on

data it hasn't seen before. With these measures in place, the code verifies whether your XGBoost model is accurate, robust and able to generalize on the unseen data. This thorough check provides insights in terms of possible problems such as overfitting, and it ensures if the model can be used in practical scenarios.

Once we run this code to check if the model will "overfit" on this data as well as check how good the model might be fit on the whole set of data, We made the conclusion in the result as follows., The cross-validation results are as follows:

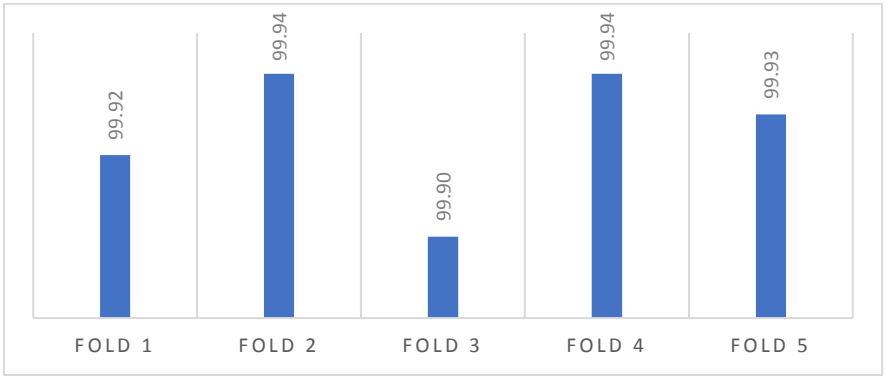


**Figure 8:** Fold Results

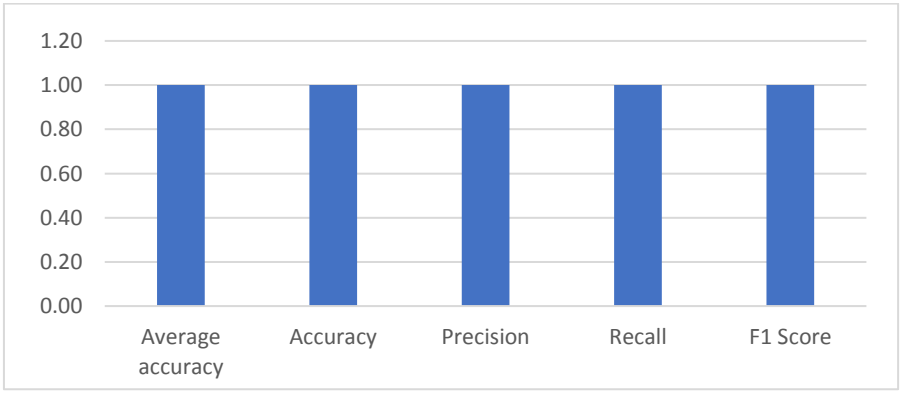


**Figure 9.** Test Set Results

The overall Accuracy across all folds = 99 .92 % & shows that the model performance is high across different sets of data in Cross Validation. Following cross-validation, the dataset is divided into two sets, we have the training set along with a test set. It involves splitting the data into two separate

datasets so that the model can be evaluated on out-of-sample (OOS) data.

The code utilizes the optimized XGBoost model for training on the training set.

Once trained, the code applies the learnt model on the test set and calculates performance using different metrics:

Accuracy: The accuracy of the model on the test set is 100%, indicating that it correctly classifies all samples.

Precision: The precision of the model is 100%, showing that it has no false positives.

Recall: The recall is 100%, suggesting that the model correctly identifies all positive instances.

F1 Score: The F1 Score is also 100%, indicating a perfect balance between precision and recall.

The results above demonstrate that the optimized Results with cross-validation and one independent test dataset below show how the optimized XGBoost model is working very well. The high accuracy, precision, recall, and F1 Score indicate that the model is not overfitting and generalizes effectively to new data, making it suitable for real-world applications.

## Conclusions

The primary goal of this analysis was to explore and model cybersecurity attack data using various machine learning techniques. We aimed to build classification models to detect and classify cyberattacks based on a dataset that contained multiple features related to network traffic and attack types.

To achieve the goal, we started by exploring the dataset, visualizing the distribution of attack types, and converting them into binary labels (normal vs. attack). This step helped us understand the dataset's composition and the class imbalance between normal and attack instances.

We also performed a correlation analysis to explore the relationships between different features. While not discussed in detail, this analysis can provide insights into which features are most relevant for classification

The dataset was preprocessed by removing unnecessary columns, dividing it into training and testing sets, and normalizing the data using the RobustScaler to handle outliers effectively.

We use Logistic Regression Model, model was trained, and its performance was evaluated. The results indicated an accuracy of 88%, demonstrating the model's ability to classify network traffic effectively. Precision, recall, and F1-score were also high, indicating a well-rounded performance.

We replaced the logistic regression model with an XGBoost classifier, which further improved the classification performance. The XGBoost model achieved an accuracy of 100% with perfect precision, recall, and F1-score, suggesting excellent predictive capabilities.

We also made an Additional Model Comparison Beyond logistic regression and XGBoost. We experimented with other machine learning models, including Random Forest, Naive Bayes, and Support Vector Machines (SVM). These models provided varying levels of accuracy and performance. Random Forest achieved perfect results similar to XGBoost, while Naive Bayes and SVM had lower accuracy due to the dataset's complexity.

In conclusion, this analysis demonstrated the effectiveness of machine learning models, particularly XGBoost and Random Forest, in classifying cybersecurity attacks based on network traffic data. These models exhibited high accuracy, precision, recall, and F1-scores, indicating their potential utility in real-world cybersecurity applications. The choice of the most suitable model may depend on specific requirements and the importance of minimizing false positives or false negatives. Further research and feature engineering could enhance the models' performance and robustness in identifying and mitigating cyber threats.

**References**

A. A. Al-Shammari, (2019): Machine Learning for Cyber Security: A Review of Methods and Applications, Security and Privacy, IEEE (Volume: 17, Issue: 1).

Cybersecurity and Infrastructure Security Agency (CISA), (2021): Avoiding Social Engineering and Phishing Attacks

D. Wang, Y. Zhang, and W. Sun, (2019): Real-Time Cyberattack Detection Using Machine Learning: A Case Study, ACM Transactions on Information and System Security (TISSEC), 22(3), 1-35,.

E. Gelenbe and M. Nakip, (2023): Real-Time Cyberattack Detection with Offline and Online Learning, 2023 IEEE 29th International Symposium on Local and Metropolitan Area Networks (LANMAN), London, United Kingdom, 2023, pp. 1-6, doi: 10.1109/LANMAN58293.2023.10189812.

E. Tekiner, A. Acar, A. S. Uluagac, E. Kirda, and A. A. SoK, (2021): Cryptojacking Malware Selcuar" Xiv:2103.03851v2 [cs.CR] .

F. Abbas and A Taeihagh, (2024): Unmasking deepfakes: A systematic review of deepfake detection and generation techniques using artificial intelligence, Expert Systems with Applications"

Forrester Consulting, (2011): A Study on The Quantifiable Benefits Of Service Automation.

G. Geetharamania, K. Dhinakaranb, J. Selvarajb, S. Christopher and E. Singhc. (2021): "Sport-utility vehicle prediction based on machine learning approach. Journal of Applied Research and Technology" https://www.scielo.org.mx/pdf/jart/v19n3/2448-6736-jart-19-03-184.pdf

Gartner, (2023): Top 10 Security and Risk Management Trends for 2023.

Justin Greis & Marc Sorel, (2024): McKinsey, The cybersecurity provider's next opportunity: Making AI safer

Kaspersky, (2022): Cybercriminals attack users with 400,000 new malicious files daily – that is 5% more than in 2021,

https://www.kaspersky.com/about/press-releases/2022_cybercriminals-attack-users-with-400000-new-malicious-files-daily---that-is-5-more-than-in-2021.

M. Ballou, (2022): IDC MarketScape: Worldwide Application Security Testing, Code Analytics, and Software Composition Analysis 2022 Vendor Assessment — Coordinating Security and Quality for Resilience and DevSecOps,.

M. Tavallaee, E. Bagheri, W. Lu, & A. A. Ghorbani. (2009): A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA) (pp. 369-373). IEEE

N.Guna sree, M.Divya, N.Reshma, Mr.K.Rajendra Prasad, (2021): Loan Approval Prediction Using Machine Learning Algorithm- Decision tree, UGC Care Group I Journal Vol-11, ISSN : 2347-7180

National Institute of Standards and Technology (NIST), NIST Cybersecurity Framework, Intrusion Detection Systems (IDSs)

R. Mangal, A.V. Nori, A. Orso, (2019): Robustness of Neural Networks: A Probabilistic and Practical Approach. https://arxiv.org/abs/1902.05983,

S. Hettich and S.D. Bay, (1999): The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science,.

S. Rawat and J. Abraham, (2021): Deep Learning for Cyberattack Detection: A Review, arXiv preprint arXiv: 2102.02385.

Steve Morgan, (2017): Cybercrime damages will cost the world $6 trillion annually by 2021, Cybersecurity Ventures, Cybercrime Report.

T. Chen and C. Guestrin, (2016) XGBoost: A Scalable Tree Boosting System, https://arxiv.org/abs/1603.02754, https://doi.org/10.48550/arXiv.1603.02754.

Verizon, (2022): Data Breach Investigations Report. Y. Dong, R. Wang and J. He, (2019): Real-Time Network Intrusion Detection System Based on Deep Learning, IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, pp. 1-4, doi: 10.1109/ICSESS47205.2019.9040718,.