WWW.IJRPR.COM

**International Journal of Research Publication and Reviews**

Journal homepage: www.ijrpr.com ISSN 2582-7421

# Intrusion Detection System Using Machine Learning on Denial of Service (Dos) Attack Detection

***Abdulrahman Ibrahim[1], Gabi Danlami[2]***

**[1]MIS/ICT Directorate Abdu Gusau Polytechnic Talata Mafara, Zamfara State Nigeria**
**[2]Department of Computer Science, Kebbi State University of Science and Technology Aliero Nigeria**
**Email: Sikanoma4u@gmail.com**

**ABSTRACT**

Worldwide, the Internet is connected across the country. There are threats of network attacks in this Internet environment. The risk of integrity and confidentiality has also increased with the density of information and global reach. Security breach has become too easy. In these days the improvement of the network security is therefore highlighted. Protection of the Network allow the unintended interference to some form to network and avoid it. It consists of software for network intrusion detection that track the network. NIDS is positioned in the network in a strategic location to track traffic inside the network from source to destination apps. The machine would optimally screen both inbound and outbound traffic, but that would create a congestion that would hinder the system's overall pace. Finally, these methods include machine learning algorithms that render the device flexible and deliver reliable performance. Intrusion activities leave evidence in the auditing data, so it is possible to learn and distinguish the pattern of ordinary and malicious activities with machine learning algorithms. Machine training techniques can learn normal, anomalous patterns from training data, and create classifiers for computer system attacks. In the area of intrusion detection for our research works, machine learning methods, such as logistic regression, Naive Bayes, K-Nearest Neighbor and Decision Trees were used. The research provides a predictive computational approach to optimize intrusion detection in the Network Traffic Data along with implementing different methodologies for the evaluation of the best accuracy from the Classification and Deep - Learning Algorithms. A new intrusion detection system for smart networks has been developed using a two-stage distinction (Anomaly-Misuse) and a deep methodology for learning. As detection methods to identify traffic disruption that could be attacked, the Decision Tree, logistic regression, KNN and Naive Bayes were used and Deep learning used an ANN method that would recognize the attacks as they exist. The analysis has used the complete 42 dimensional features of the training data set. The findings indicate that the high accuracy values are 100% with 0.10 recall levels at stage 1 of the appraisal, and 99.5% with 0.99 at stage 2 of the validation. Experimental findings indicate that the design of the decision tree contributes to high precision in contrast with other algorithm.

**Keywords:** Cybersecurity, Networks, Internet, Data

## 1.0 Introduction

An Intrusion Detection System (IDS) is a device or a software based application that can be used to detect any malignant or privacy related activities present in the network. Any kind of illegal activity is directly reported to the administrator of the system or it's been sent directly to the central system using a system information and event management (SIEM) system. Any form of secure network should provide proper data privacy, data integrity and also assurance against denial of service (DoS) attack. The primary aim of the Intrusion Detection is to correctly identify any illegitimate use or exploitation of computer system. Intrusion Detection System (IDS) is based on the assumption that the system can detect the behavioral difference an intruder's activities as compared to legitimate user so that their unauthorized actions can be detected.

### *1.1 Intrusion Detection Category*

An Intrusion Detection System (IDS) is categorized based on its scope and deployment strategy. These systems are broadly classified into two types: **Network Intrusion Detection Systems (NIDS)**, which monitor traffic across an entire network, and **Host-Based Intrusion Detection Systems (HIDS)**, which focus on individual devices or endpoints. This classification ensures adaptability to both localized and large-scale security environments (Dong & Wang, 2018).

### *1.2 Network Intrusion Detection Systems (NIDS)*

Network Intrusion Detection Systems (NIDS) are placed at very specific strategic points within the network to monitor traffic to and from all the devices connected across the network. NIDS performs analysis of all the currently passing traffic present on the entire subnet and matches it with the already known traffic generated by the previous attacks. Once the unmatched behavior is spotted, an alert is directly sent to the network administrator.

Also, NIDS can be classified into two types such as on-line and off-line NIDS. Online NIDS deals with the real time network data while the off-line NIDS looks after the stored data to check and decide whether it is an attack or not. NIDS can be used on a stand-alone network system or can be combined with other technologies to increase the overall detection and prediction rates. One such example is using Artificial Neural Network (ANN) based Intrusion Detection System which is capable of handling large volume of data in a very self-organizing structure which in results helps in better recognition of different intrusion patterns. As seen below in the figure 1.1 we can look at the basic components present in the NIDS Architecture. (Conrad, Misenar, & Feldman, 2017)

**Research Questions**

- How can machine learning and classification techniques be adequately enforced to detect Network Intrusion System (Denial of Services (DoS)?
- How can we identify the best performing classification technique and algorithm?
- How can we use this technology to successfully implement on real time data traffic?

**1.5 Aims and Objectives**

The aim of this research is to develop and evaluate an improved intrusion detection system (IDS) using machine learning techniques to enhance the accuracy, adaptability, and responsiveness of cyber threat (Denial of Service (DoS)) detection in networked systems. For this research we'll be using Classification Technique based on Machine Learning.

Classification is a supervised machine learning approach in which the computer program learns from the provided input and then uses that data to classify the information in different form.

Classification is a process of sorting the given data into number of different classes. It can be performed on both structured as well as unstructured data.

## 2. Literature Review

### *2.1 Introduction*

Literature Review plays a vital role in any research for providing possible relevant information. Literature Review can be stated as an overall summary of the existing articles which are published in the related field of work.

#### *2.1.1 Network Intrusion Detection Mukherjee, Heberlein, & Levitt (1994)*

- According to the research done by Biswanath Mukherjee, L. Todd Heberlein, & Karl N. Levitt (1994), it states that Intrusion Detection is a new approach in the field of security in pre-existing computers and other data networks.
- In their research they concluded that Intrusion Detection Systems were based on host-audit-trail and network traffic analysis and their goal was to detect attacks possibly in real-time.
- A number of prototype IDS's were built and their overall concept was quite promising.
- There were number of limitations in prevention based approach for network security some of them listed below as:-

1. It was impossible to build a useful system which was precisely secure. Due to some design flaws in the system large number of components cannot be excluded.
2. Crypto-based systems was not able to prevent attack against lost or stolen keys, or hacked passwords.
3. Lastly, the system was not secure enough from the insiders misusing their privileges.

Few illustrations of intrusion that are involved with systems administrators are as follows:-

- The user's data can be altered and manipulated through unauthorized access to their files.
- Unauthorized use of computing resources.

- The research also defines and gives us a brief idea about different types of Intrusion Detection System along with their advantages and disadvantages.

### *2.1.2 Classification Techniques in Machine Learning: Applications and Issues Soofi & Awan (2017)*

- Aized Amin Soofi and Arshad Awan took the concept of classification and gave us a detailed information about different type of classification technique that are used in machine learning.
- In the research various classification techniques have been in discussed in detail along with their useful applications. An overview of different classification technique with their applicable applications are detailed in table 2.1.

Table 2.1: Applications of Classification Technique

| **Classification Techniques** | **Applications** |
|---|---|
| ID3 | predicting student performance<br>land capability classification<br>tolerance related knowledge acquisition computer crime forensics fraud detection application |
| C4.5 | Decision making of loan application by debtor<br>Predicting Software Defects<br>Thrombosis collagen diseases<br>Electricity price prediction coal logistics customer analysis<br>Selecting Question Pools |
| Bayesian Network | automatic and interactive mode for Image Segmentation traffic incident detection signature verification<br>efficient patrolling of nurses examine dental pain<br>telecommunication and internet networks |
| K- Nearest neighbor | Microarray data classification<br>Phoneme Prediction<br>Face recognition<br>Agarwood oil quality grading<br>Classification of nuclear receptors and their subfamilies<br>Short-term traffic flow forecasting<br>Plant Leaf Recognition |
| SVM | Scene classification<br>Predict corporate financial distress<br>Induction motor's fault diagnosis<br>Analog circuit fault diagnosis enterprise market competition |

Along with the applications, few of the issues regarding process of the algorithms were also discussed in the paper. Some of the solutions were also provided related to the issue caused in the classification approach. Some of them are listed in table 2.2

Table 2.2: Classification Techniques Issue and Solutions

| **Classification Approach** | **Issue** | **Solution/Technique** |
|---|---|---|
| Decision tree (ID3 and C4.5) | multi valued at-<br>tributes<br>Complex information entropy and attribute with more values<br>Noisy data classification | Algorithm by combining ID3 and<br>association function(AF) modification to the attribute selection<br>methods, pre pruning strategy and rainforest approach<br>Enhanced algorithm with Taylor formula<br>Credal-C4.5 tree |
| Bayesian Network | Attributes conditional density estimation Inference (large domain discrete and continuous variables)<br>Multi-dimensional data | Gaussian kernel function decision-tree structured conditional probability greedy learning algorithm |
| K nearest neighbor | space requirement time requirement KNN scaling over multimedia dataset | Prototype selection feature selection and extraction methods finding R-Tree index multimedia KNN query processing system |
| SVM | controlling the false positive rate low sparse SVM classifier<br>multi-label classification | Risk Area SVM (RA-SVM)<br>Cluster Support Vector Machine<br>(CLSVM) fuzzy SVMs (FSVMs) |

### *2.1.3 Network Traffic Classification Techniques and Comparative Analysis Using Machine Learning Algorithms Shafiq et al. (2016)*

- The paper states the theoretical concepts about different machine learning algorithms based on network intrusion detection anomaly.
- Machine Learning technique was based on labelled datasets. In this approach, a machine learning classifier is trained as input for the system and then using trained sample prediction form classifier, unknown classes are classified.
- Network Traffic Classification was the first step to identify and correctly analyze various types of applications flowing in a network.
- Using this technique, it helped Internet Service Providers (ISP) or network operators to manage and upgrade overall performance of the network.
- There were multiple techniques available to classify the internet traffic such as Port Based, Pay Load Based and Machine Learning out of which Machine Learning was the most popular and useful.
- In this paper they discussed in a sequential format about network traffic classification techniques and real time internet data was developed using network traffic capture tool.
- Feature Extraction tools were used to extract features from four machine learning algorithms such as Support Vector Machine (SVM), C4.5 Decision Tree, Naive Bayes and Bayes Net Classifiers.

Table 2.3: Accuracy Results and Comparison of MLA

| **Classifiers** | **Accuracy (%)** | **T Time (Second)** |
|---|---|---|
| **C4.5** | 78.9189 | 0 |
| **SVM** | 74.0541 | .0.3 |
| **BayesNet** | 68.1081 | 0.01 |
| **Naive Bayes** | 71.8919 | 0.01 |

- Table 2.3 show the accuracy results of each machine learning algorithms with their total time required to run each model.
- The experimental analysis showed that C405 Decision Tree classifier gave very good accuracy results as compared to other classifiers.

- For comparative analysis of these four algorithms, they captured WWW, DNS, FTP, P3P and TELNET application for traffic interval of 1 minute using Wire Shark Tool and feature extraction using Netmate Tool.

#### *2.1.4 Practical real-time intrusion detection using machine learning approaches*

- The increasing prevalence of system attacks are a well-known issue that can affect the availability, confidentiality and integrity of critical information for both individuals and businesses. Sangkatsanee, Wattanapongsakorn, & Charnsripinyo (2011).
- The study here proposes an intrusion detection method that used a supervised machine learning technique in real time. Our technique is quick and efficient and can be applied to a number of machine learning techniques, Sangkatsanee, Wattanapongsakorn, & Charnsripinyo (2011).
- They have used various efficient machine learning algorithms to analyze the efficiency of IDS approach. Among the various methods, hypotheses showed that the Decision tree approach is capable of surpassing other techniques.
- With the above result, furthermore using the decision tree algorithm they developed a Real Time Intrusion Detection System that's helps in classifying the attack data among the normal online network data.
- Further they also detected network data using the information gain system which was our feature selection criteria by identifying the 12 important features of network data
- As the detection rate reached above 98 percent in 2 seconds which helped the RT-IDS to distinguish between the normal network actions among the Probe and Denial of service attacks.
- They also created a new post-processing technique to reduce the false alarm rate and improve the efficiency and precision of the intrusion detection system Sangkatsanee, Wattanapongsakorn, & Charnsripinyo (2011).
- In the end, they also evaluated the IDS model by testing the detection speed, CPU usage, memory consumption, and detection accuracy.
- The research work had several contributions as follows: -
  - Presented an uncomplicated IDS model which required no human expert to identify the attack and also work on pre-existing systems.
  - Proposed 12 essential features to identify which were related to DoS or other attack types.
  - Created a Real-Time model which was able to identify attack as well as classify them into three groups such as Normal, Denial of Service (DoS) and Port Scanning (Probe).
  - Developed a post-processing procedure to reduce false alarm rate.

#### *2.1.5 Using Genetic Algorithm for Network Intrusion Detection*

- This paper describes a technique of applying Genetic Algorithm (GA) to network Intrusion Detection Systems (IDSs) W. Li (2019)
- A brief overview of the Intrusion Detection System, genetic algorithm, and related detection techniques is presented, W. Li (2019)
- This implementation includes both temporal and spatial information of different network connections in the encoding.
- Genetic Algorithm is part of computational models based on the concept of principals of evolution and natural selection.
- Random chromosomes are selected from the population at the start of the genetic algorithm process. The set of chromosomes during a stage of evolution is called as population these chromosomes are representations of the problem to be solved.

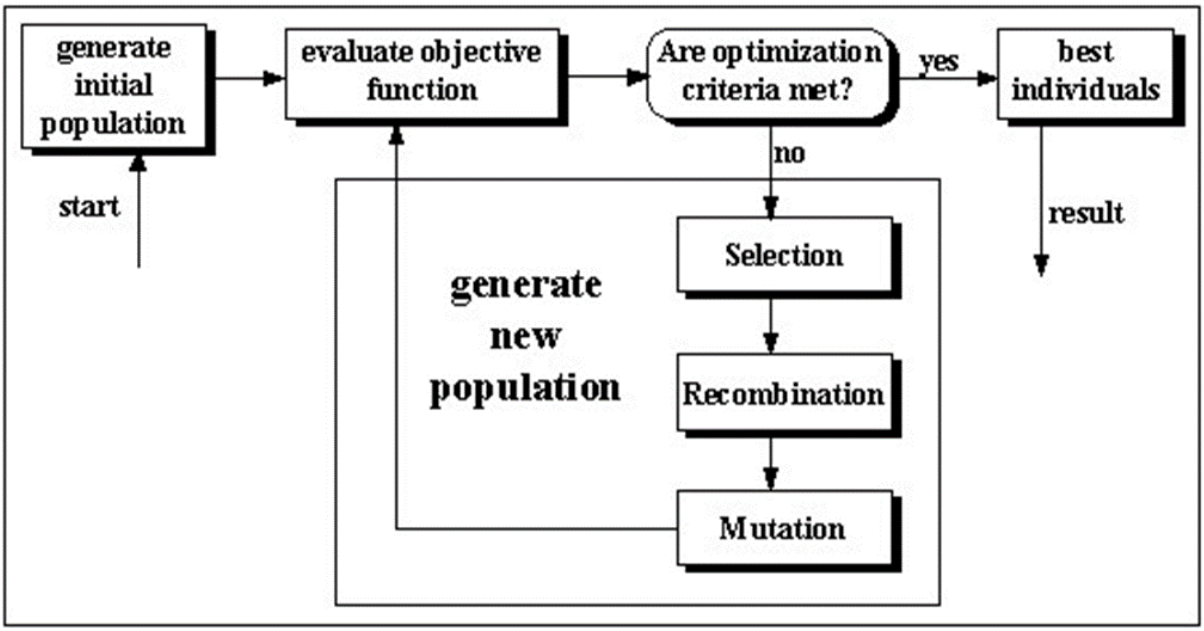



Figure 2.1: Structure of Genetic Algorithm

- Using Genetic Algorithm for NIDS helps in proper identification of complex divergent performance.
- The research work is based on the concepts of TCP/IP network protocols along with encoding the network connection data into the IDS.

### *2.1.6 Survey of Classification Techniques in Data Mining Phyu (2018)*

- Classification is a Data Mining (Machine Learning) technique that is used to predict category for data instances.
- Major Classification algorithms are explained in brief such as Decision Tree, Bayesian Networks, K-Nearest Neighbor (KNN), Genetic Algorithms, Fuzzy Logic Technique and Case-Based Reasoning.
- This survey research was conducted to provide an extensive review of different classification techniques in Machine Learning.
- Different analysis tools were used to conclude any matching patterns or find relationships between large datasets. These tools include mathematical algorithms, machine learning algorithms and statistical models.
- Classification techniques was capable of processing very large datasets as compared to regression and it's been applied to various applications due to its popularity.
- From the research done on paper is was concluded that Decision Tree and Bayesian Network were the best algorithm while working on different machine learning techniques.

### *2.1.7 A two-stage hybrid classification technique for network intrusion detection system Hussain, Lalmuanawma, & Chhakchhuak (2016)*

- Based on the research work in this paper it is stated that Conventional Network Intrusion Detection System (NIDS) often uses individual classification techniques which results in failure to serve better detection rate.
- In this paper, a new two-stage hybrid classification method has been proposed by using Support Vector Machine (SVM) and Artificial Neural Network (ANN).
- In the first stage SVM will be used as an anomaly detector while in second stage ANN used as misuse detector.
- The main objective of considering this method was to increase classification accuracy with less probability of false alarm.
- The First stage detects any possible intrusion or abnormal activities through anomaly. On the other hand, second stage analyses the attack and classifies the type of attack into four different classes such as Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L) and Probe, Hussain, Lalmuanawma, & Chhakchhuak (2016)
- Based on the results obtained through simulation process it was concluded that proposed hybrid classification method was much more effective and also outperformed the traditional conventional method of individually performing Support Vector Machine (SVM) and Artificial Neural Network (ANN) algorithms.

- The proposed technique showed much more reliable results of detecting intrusion activities over the network information.
- All the simulations ran were based on the NSL-KDD datasets which were the updated version of the typical KDD99 version dataset.

#### *2.1.8 Intelligent Intrusion Detection Systems using Artificial Neural Networks Shenfield, Day, & Ayesh (2018)*

- This paper presents a rather new approach for malicious network traffic attacks using Artificial Neural Network (ANN) based Intrusion Detection System (IDS).
- The experiment uses typical network traffic benign data such as images, dynamic library files, log files, music files, etc. and also malicious data such as shell code files and vulnerability repository.
- This proposed architecture of ANN was able to easily distinguish between benign and malicious network traffic more accurately.
- An average accuracy of over 98 percent was achievable with the proposed ANN architecture. Also, average area under the Receiver Operator Characteristic (ROC) curve of 0.98 and in 10-fold cross-validation with an average false positive rate of less than 2 percent was achieved.
- These results showed that the proposed ANN classification technique was accurate, precise and robust.
- The proposed model was also able to enhance the expediency of intrusion detection system applied on both conventional network traffic and for network traffic for cyber-physical systems as smart-grids.
- The research method currently works on an offline approach for detecting shellcode patterns while an online and real-time network intrusion detection system is and ongoing project that can integrate with the existing system.

#### *2.1.9 An Intelligent Intrusion Detection System for Mobile Ad-Hoc Networks Using Classification Techniques. Ganapathy, Yogesh, &* Kannan (2018)

- This paper proposes a multi-level approach of classification technique for effective intrusion detection in Mobile AD-Hoc Networks.
- The proposed concept uses combination of tree classifier which uses a labelled training data and an Enhanced Multiclass SVM algorithm for binary classification.
- For the procedure, data reduction process was carried out using attribute selection technique from KDD 99 cup data set.
- From the tests conducted through these experiments, improvement was observed on both high detection rates as well as low false alarms rates.
- In this research work, SVM was combined with Decision Tree in order to design a multiclass SVM which would be capable of classifying the attacks in four different classes such as Probing, Denial of Service (DoS), User to Root (U2R) and Root to Location (R2L) Ganapathy, Yogesh, & Kannan (2018)
- The main advantage of this algorithm was to provide effective preprocessing algorithm along with classification approach to attack network intrusion attack on Mobile Ad hoc Networks (MANETs) which would eventually improve overall accuracy and reduce training as well as testing time.

#### *2.1.10 A Deep Learning Approach for Network Intrusion Detection System Javaid et al. (2016)*

- A Network Intrusion Detection System (NIDS) assists the administrator to detect any network security breaches in their management system Javaid et al. (2016)
- Despite all of these, there were many challenges faced while developing a formative and effective NIDS system for impulsive attacks.
- Therefore, to overcome all the issues, a deep learning-based approach was developed for efficient and formidable NIDS structure.
- For building this model, a combination of models were used such as Self-Taught Learning (STL), a deep learning based technique on NSL-KDD a benchmark dataset for Network Intrusion Javaid et al. (2016)
- The results were compared on the grounds of accuracy, precision, recall and f-measure values.
- NIDS are categorized into two classes such as:- Javaid et al. (2016)

    1. Signature (misuse) based NIDS (SNIDS)
    2. Anomaly Detection based NIDS (ADNIDS).

- In SNIDS, signature attacks are pre-installed in the NIDS. A pattern matching sequence is conducted for the network traffic against the installed signatures on the system to detect intrusion. On the other side, ADNIDS classifies a network intrusion when it detects a irregularity in the normal traffic patterns.

- SNIDS is useful in detecting known attacks and shows results with high accuracy with less false alarms. Whereas, ADNIDS is used to detect relatively unknown and new form of attacks.

#### *2.1.11 HIDE: A Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification Zhang et al. (2010)*

- This paper introduces a new approach with Hierarchical Intrusion Detection (HIDE) system which uses statistical preprocessing and neural network classification for detecting network-based attacks.
- For the proposed research five different types of neural network classifiers were used such as: -
    1. Perceptron,
    2. Backpropagation (BP),
    3. Perceptron-backpropagation-hybrid (PBH),
    4. Fuzzy ARTMAP, and
    5. Radial-based Function
- The results indicated that Backpropagation (BP) and Perceptron-backpropagationhybrid (PBH) were the best among five for more efficient classification of our data.
- A Stress-Test was also conducted on the entire system to conclude that HIDE was a reliable technique in detecting flooding attacks with very low attack intensity as much as only five to ten percent.

## 3. Research Methodology

### *3.1 Introduction*

Research methodology is a systematic, theoretical analysis of the methods applied to a field of study. It encompasses the principles, procedures, and techniques employed by researchers to collect, analyze, and interpret data

This section presents a comprehensive overview of the research design employed in the development and evaluation of the proposed Intrusion Detection System (IDS) using Machine Learning (ML). The research design encompasses various stages, including data collection, preprocessing, feature engineering, model selection, training, and evaluation.

### *3.2 Research Design*

There are number of steps involved in performing a machine learning technique based on classification method using a dataset. The basic steps involved in performing a machine learning process are shown in figure 3.1

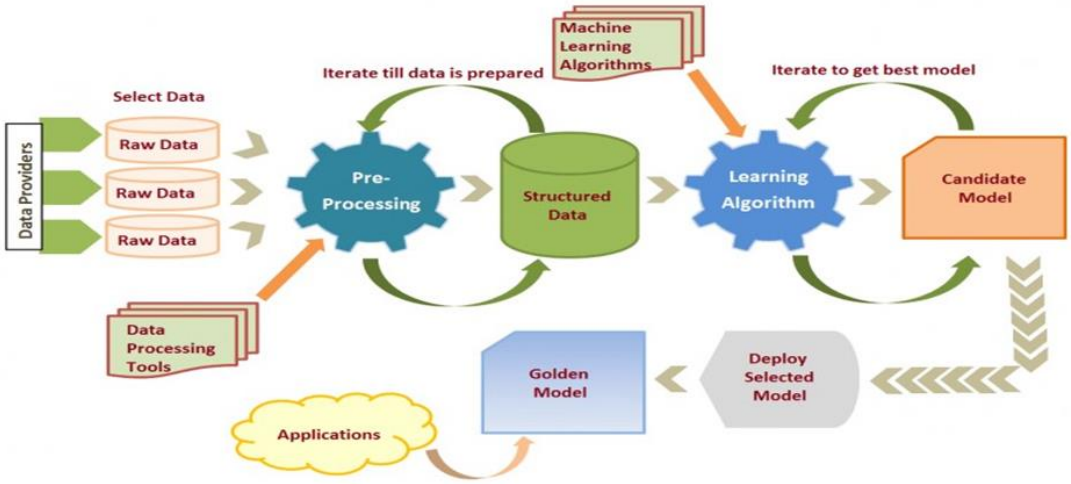



Figure 3.1: Steps of Machine Learning (Mittal 2017)

### *3.3 Dataset Description*

- The main aim of this research is to make a comparison between numbers of classification techniques and apply on the dataset to predict which model gives us the highest accuracy and precision with less false positive rate.
- For this research on Network Intrusion Detection System (NIDS) we have selected a dataset based on intrusion system.
- The dataset is created and audited which contains wide range of intrusion simulated in a military network environment to acquire raw TCP/IP dump data for a network simulating a US AIR Force LAN network.
- A connection in a network is a sequence of TCP packets which starts and ends at some particular time duration during which the data flows from a source IP address to the target IP address where each connection is labelled as either normal or as an attack with exactly one specific attack type.
- The selected dataset has been divided into two sets, one for training while other for testing purpose. Each dataset contains total 42 and 41 features respectively where each serves a different purpose in processing.
- The dataset classifies the number of attacks into four different categories such as (Hussain, Lalmuanawma, & Chhakchhuak 2016)

1. **Denial of Service (DoS):**

A DoS attack is a type of attack in which the intruder restricts access to the authorized person into the system or network.

2. **Remote to Local (R2L):**

It is a type of attack that aims at gaining access to a local account from another host or network.

3. **User to Root (U2R):**

These attacks are exploitation in which the intruder starts off on the system with a limited user account or normal user privileges and attempts to abuse vulnerabilities in the system to gain root access.

4. **Probe:**

Probe is an attack in which the hacker scans a machine or a network to gather information or find known vulnerabilities.

### *3.4 Proposed Technique*

In this section, we detail the step-by-step approach taken in the development and evaluation of the proposed Intrusion Detection System (IDS) leveraging Machine Learning (ML) techniques.

**Data Preprocessing**

In the field of machine learning, Data Preprocessing is the first step to be carried out before initializing the process. Data Preprocessing is basically used to transform and convert raw and unfiltered data into a much more suitable and understandable format.

Typically, real world data might be incomplete, inconsistent, inaccurate, unstructured form and might also have some errors and missing values. To overcome all this, data preprocessing is used. It helps in clean, format and organize the raw data and make it ready to use in building machine learning model.

Data Preprocessing cannot be conducted into a single process hence it is distributed among several steps.

The steps involved in Data Preprocessing are as below :( Dey 2018)

1. **Import Libraries**

- Before we start any programming, our first step is importing various libraries required to build a model.
- A library is primarily a collection of modules that can be called and used. Libraries is a collection of functions and methods which allows us to perform many actions without actually writing any code.

```
In [1]:
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from math import * # module math
import matplotlib.pyplot as plt # visualization
from PIL import Image
import seaborn as sns # visualization
import itertools
import io
import plotly.offline as py # visualization
py.init_notebook_mode(connected=True) # visualization
import plotly.graph_objs as go # visualization
from plotly.subplots import make_subplots
import plotly.figure_factory as ff # visualization
import warnings
warnings.filterwarnings("ignore")

%matplotlib inline
```

Figure 3.2: Importing Libraries for Python

- From the figure 3.2, we have installed number of libraries such as *numpy,pandas,math,etc*. This library plays a vital role in performing any program. For example, pandas is used for data processing, numpy is used to perform linear algebra while *plotly* is used to make visualization in the program.

2. **Import the Dataset**

- A lot of datasets are in the form of CSV formats. CSV is a comma separated-values file which allows data to be stored in tabular format.

- For our research we have selected dataset based on Network Intrusion Detection system. As we will be performing many classification techniques on the dataset therefore we have divided our dataset into two subsets for training as well as testing the data.

```
In [3]:
Training = pd.read_csv(r"C:\Users\HP\Downloads\Dissertation Files\Train_data.csv")
Testing = pd.read_csv(r"C:\Users\HP\Downloads\Dissertation Files\Test_data.csv")

In [4]:
print(Training.head(5))

print("Training data has {} rows & {} columns".format(Training.shape[0],Training.shape[1]))
```

```
   duration protocol_type   service flag  src_bytes  dst_bytes  land  \
0         0           tcp  ftp_data   SF        491          0     0
1         0           udp     other   SF        146          0     0
2         0           tcp   private   S0          0          0     0
3         0           tcp      http   SF        232       8153     0
4         0           tcp      http   SF        199        420     0

   wrong_fragment  urgent  hot  num_failed_logins  logged_in  num_compromised  \
0               0       0    0                  0          0                0
1               0       0    0                  0          0                0
2               0       0    0                  0          0                0
3               0       0    0                  0          1                0
4               0       0    0                  0          1                0

   root_shell  su_attempted  num_root  num_file_creations  num_shells  \
0           0             0         0                   0           0
1           0             0         0                   0           0
2           0             0         0                   0           0
3           0             0         0                   0           0
4           0             0         0                   0           0
```

Figure 3.3: Importing Datasets

- As seen in the figure 3.3, we have imported two different datasets and labelled them as **Training** and **Testing** and read it using a method called *read _csv*

- To check if the dataset has been correctly imported, we will use the *print* command to check the dataset and display overview of columns from the dataset.

3. **Checking Missing Values**

- Sometimes we may find dataset which have missing values or data in the dataset. There might be many reasons for this such as programming error or the data might be lost transferring manually from legacy dataset.

- To fill the missing values, we can simply delete the entire row or replace the missing value with some sensible data.

```
In [6]:
def dataoveriew(df, message):
    print(f'{message}:\n')
    print("Rows:", df.shape[0])
    print("\nNumber of features:", df.shape[1])
    print("\nFeatures:")
    print(Training.columns.tolist())
    print("\nMissing values:", df.isnull().sum().values.sum())
    print("\nUnique values:")
    print(df.nunique())

In [7]:
dataoveriew(Training, 'Overiew of the Training dataset')

Overiew of the Training dataset:

Rows: 25192

Number of features: 42

Features:
['duration', 'protocol_type', 'service', 'flag', 'src_bytes', 'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot', 'num_f
ailed_logins', 'logged_in', 'num_compromised', 'root_shell', 'su_attempted', 'num_root', 'num_file_creations', 'num_shells',
'num_access_files', 'num_outbound_cmds', 'is_host_login', 'is_guest_login', 'count', 'srv_count', 'serror_rate', 'srv_serror
_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_hos
t_srv_count', 'dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rat
e', 'dst_host_serror_rate', 'dst_host_srv_serror_rate', 'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'class']

Missing values: 0

Unique values:
duration                     758
protocol_type                  3
service                       66
flag                          11
src_bytes                   1665
```

Figure 3.4: Finding Missing Values

- We have selected the **Training** dataset and checked if there's any missing value in the data. To check missing values, we have elected to print overview of the entire Training dataset which shows us the results as total number of rows and number of columns i.e. number of features. and also, if any missing values which in our case is 0 as we can see from figure 3.4

4. **Encoding Categorical Data**

- Sometimes our data can be in qualitative form, i.e., we have texts as in our data. Our computer system cannot understand data in text format and therefore cannot process it.

- Since our models and systems are based on mathematical equations and calculations therefore it is necessary to convert categorical data into numerical data format.

- We have converted the data into numerical format using *sklearn* library and used it *LabelEncoder* to conduct the process.

To further enhance and simplify our data, we have assigned default value of mean as 0 and variance as 1 to all the numerical attributes in our dataset as we can see in Figure 3.5

```
In [12]:
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# extract numerical attributes and scale it to have zero mean and unit variance
cols = Training.select_dtypes(include=['float64','int64']).columns
sc_train = scaler.fit_transform(Training.select_dtypes(include=['float64','int64']))
sc_test = scaler.fit_transform(Testing.select_dtypes(include=['float64','int64']))

# turn the result back to a dataframe
sc_traindf = pd.DataFrame(sc_train, columns = cols)
sc_testdf = pd.DataFrame(sc_test, columns = cols)

In [13]:
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()

# extract categorical attributes from both training and test sets
cattrain = Training.select_dtypes(include=['object']).copy()
cattest = Testing.select_dtypes(include=['object']).copy()

# encode the categorical attributes
traincat = cattrain.apply(encoder.fit_transform)
testcat = cattest.apply(encoder.fit_transform)

# separate target column from encoded data
enctrain = traincat.drop(['class'], axis=1)
cat_Ytrain = traincat[['class']].copy()
```

Figure 3.5: Encoding categorical data

5. **Splitting the Dataset into Training set and Test Set**

- After converting the data, we need to split the dataset into two sets as a Training set and a Testing set
- We will train our machine learning model on our training set. Our learning model will try to understand any correlations in our training set and then test the model on our testing set to check how accurately it can predict.
- Generally, the dataset is split into 80% and 20% for training set and testing set respectively.

```
In [18]: from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test = train_test_split(Training_x,Training_y,train_size=0.70, random_state=2)
```

Figure 3.6: Splitting the Dataset

- We can do this by importing *train test split* from *model selection* library of scikit as seen in figure 3.6
- While splitting a dataset, we will create 4 sets as:
  - X train - Training part of the matrix of features
  - X test - Test part of the matrix of features
  - Y train - Training part of the dependent variables associated with the X train sets
  - Y test - Test part of the dependent variables associated with the X test sets

6. **Feature Scaling**

- In machine learning, Feature Scaling means scaling the features to the same scale.
- Normalization scale features between 0 and 1, also retaining their proportional range to each other.
- Standardization scales features to have a mean(u) of 0 and standard deviation (a) of 1

$$X' = \frac{x-u}{a}$$

```
In [12]: from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# extract numerical attributes and scale it to have zero mean and unit variance
cols = Training.select_dtypes(include=['float64','int64']).columns
sc_train = scaler.fit_transform(Training.select_dtypes(include=['float64','int64']))
sc_test = scaler.fit_transform(Testing.select_dtypes(include=['float64','int64']))

# turn the result back to a dataframe
sc_traindf = pd.DataFrame(sc_train, columns = cols)
sc_testdf = pd.DataFrame(sc_test, columns = cols)
```

Figure 3.7: Feature Scaler

- We can do this by importing *StandardScaler* from library of scikit as seen in figure 3.7. Also, we have previously stated few details regarding Feature Scaler above section 4.

### *3.5 Experimental Environment*

The experimental environment is a crucial aspect of developing and testing the machine learning-based intrusion detection system (IDS).

### *3.5.2 Tools Used in Machine Learning*

Every program needs some kind of tools for processing whether it is for classification or for regression or any other technique. Similarly, in our project, to carry out various classification technique s in machine learning we also require certain tools. Tools are nothing but software applications and hardware support that required to accomplish and perform classification algorithms.

Tools required for building a classification model in machine learning are as follows:

1. **Jupyter Notebook**

- JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. (Jupyter 2020)

- Jupyter Notebook is built off of IPython, an interactive way of running Python code in the terminal using the REPL model (Read-Eval-Print-Loop) Jupyter Notebook can be installed either by using Anaconda or by Pip.

2. **Python**

- To start building our model, we need to select a programming language on which we can start our coding. For classification technique we have used **Python** as our primary programming language.

- Python is an interpreted, high-level, general-purpose programming language. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. (Wikipedia 2020d) Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management.

- The language's core philosophy is summarized in the document The Zen of Python, which includes aphorisms such as: (Wikipedia 2020e)

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Readability counts.

### *3.6 Performance Evaluation Metrics*

- We evaluate the performance of our models based on the Accuracy Metrics and is also known as a Confusion Matrix.

- With the help of Confusion matrix we can easily find out our model's Accuracy, Precision, Recall and F-Measure. (Javaid et al. 2016)

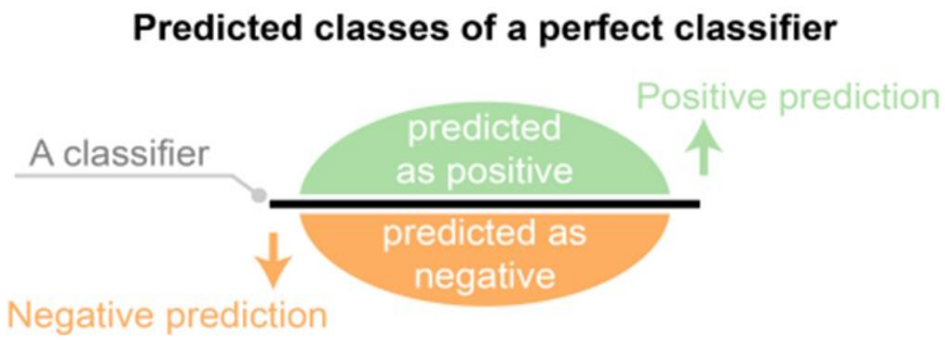
Table 4.1: Confusion Matrix

| | | **Actual Values** | |
|---|---|---|---|
| | | Positive (1) | Negative (0) |
| **Predicted Values** | Positive (1) | TP | FP |
| | Negative (0) | FN | TN |

- All the above metrics are given as:

**Predictions on test datasets**

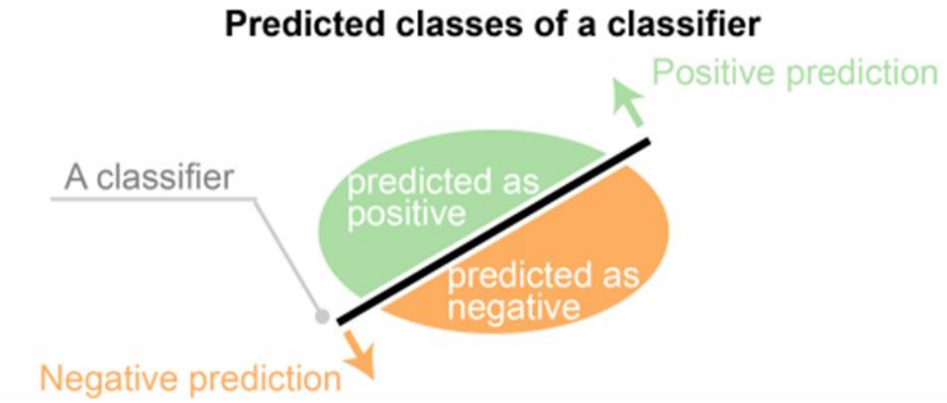
The predicted labels will be exactly the same if the performance of a binary classifier is perfect, but it is uncommon to be able to develop a perfect binary classifier that is practical for various conditions.




The performance of a binary classifier is perfect when it can predict the exact same labels in a test dataset.

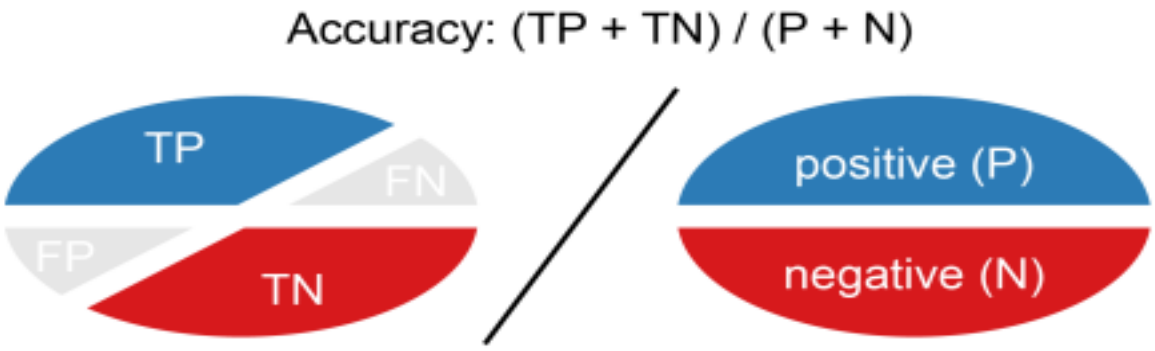
Hence, the predicted labels usually match with part of the observed labels.

The predicted labels of a classifier match with part of the observed labels.




**Accuracy**

Accuracy (ACC) is calculated as the number of all correct predictions divided by the total number of the dataset. The best accuracy is 1.0, whereas the worst is 0.0. It can also be calculated by 1 – ERR.



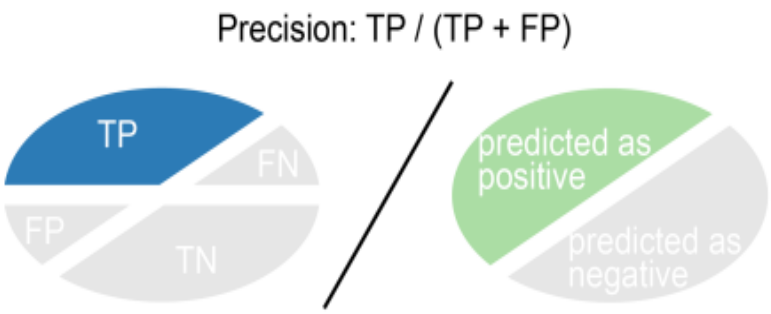

Accuracy is calculated as the total number of two correct predictions (TP + TN) divided by the total number of a dataset (P + N).

Figure 3.8

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FN} + \mathrm{FP}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}}$$

**Precision (Positive predictive value)**

Precision (PREC) is calculated as the number of correct positive predictions divided by the total number of positive predictions. It is also called positive predictive value (PPV). The best precision is 1.0, whereas the worst is 0.0.




Precision is calculated as the number of correct positive predictions (TP) divided by the total number of positive predictions (TP + FP).

Figure 3.9

$$\mathrm{PREC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

**1. Recall (R)**

- Defined as the % ratio of number of True Positives records divided by the sum of True Positives and False Negatives (FN) classified records.

Figure 3.10

$$R = \frac{TP}{(TP+FN)} *$$

**F-Measure (F)**

Defined as the harmonic mean of Precision and Recall and represents a balance between them.

Figure 3.11

$$F = \frac{2.P.R}{(P+R)}$$

**F-score**

F-score is a harmonic mean of precision and recall.

Figure 3.12

$$\mathrm{F}_{\beta} = \frac{(1+\beta^2)(\mathrm{PREC} \cdot \mathrm{REC})}{(\beta^2 \cdot \mathrm{PREC} + \mathrm{REC})}$$

β is commonly 0.5, 1, or 2.

Figure 3.13

$$\mathrm{F}_{0.5} = \frac{1.25 \cdot \mathrm{PREC} \cdot \mathrm{REC}}{0.25 \cdot \mathrm{PREC} + \mathrm{REC}}$$

Figure 3.15

$$\mathrm{F}_1 = \frac{2 \cdot \mathrm{PREC} \cdot \mathrm{REC}}{\mathrm{PREC} + \mathrm{REC}}$$

Figure 3.16

$$\mathrm{F}_2 = \frac{5 \cdot \mathrm{PREC} \cdot \mathrm{REC}}{4 \cdot \mathrm{PREC} + \mathrm{REC}}$$

## 4. RESULTS

### *4.1 Data Preprocessing*

- Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.

- Typically, real world data might be incomplete, inconsistent, inaccurate, unstructured form and might also have some errors and missing values. To overcome all this, data preprocessing is used. It helps in cleaning, formatting and organizing the raw data and making it ready for use in building machine learning model.

- There are number of steps that are collectively involved in conducting Data Preprocessing. Steps involved in Data Preprocessing are as follows:

    1. Importing Libraries
    2. Importing Dataset
    3. Checking any missing values
    4. Encoding Categorical Data
    5. Splitting the Dataset into Training set and Test Set
    6. Feature Scaling

### *4.6 Model Evaluation*

Model Evaluation is an integral part of model development process in machine learning. It helps to find the best model that represents our dataset and also how well the chosen model will work in near future.

For our dataset we have selected four classification techniques such as:

#### *4.6.1 Decision Tree*

**Cross Validation Mean Score:**

0.9956333771928133

**Model accuracy:**

1.0

**Confusion Matrix:**

$$\begin{bmatrix} 8245 & 0 \\ 0 & 9389 \end{bmatrix}$$

| | Precision | recall | F1-Score | support |
|---|---|---|---|---|
| Anomaly | 1.00 | 1.00 | 1.00 | 8245 |
| Normal | 1.00 | 1.00 | 1.00 | 9389 |
| | | | | |
| Accuracy | | | 1.00 | 17634 |
| Macro Avg | 1.00 | 1.00 | 1.00 | 17634 |
| Weighted Avg | 1.00 | 1.00 | 1.00 | 17634 |

Figure 4.2: Model Evaluation - Decision Tree

Based on the results produced using Decision Tree in Model Evaluation, the model showed an accuracy of 100% and with Cross Validation Mean Score of 0.99. All the accuracy metrics such as Accuracy, Precision, Recall and F-1 score were highest among all other classification techniques used in the procedure.

#### *4.6.2 K-Nearest Neighbor Classifier*

**Cross Validation Mean Score:**

0.9914370153431007

**Model accuracy:**

0.9937620505840989

**Confusion Matrix:**

$$\begin{bmatrix} 8168 & 77 \\ 33 & 9356 \end{bmatrix}$$

| | Precision | recall | F1-Score | support |
|---|---|---|---|---|
| Anomaly | 1.00 | 0.99 | 0.99 | 8245 |
| Normal | 0.99 | 1.00 | 0.99 | 9389 |
| | | | | |
| Accuracy | | | 0.99 | 17634 |
| Macro Avg | 0.99 | 0.99 | 0.99 | 17634 |
| Weighted Avg | 0.99 | 0.99 | 0.99 | 17634 |

Figure 4.3: Model Evaluation - K-Nearest Neighbor Classifier

Performing K-Nearest Neighbor (KNN) showed result with model accuracy of 99.37% along with Cross Validation Mean Score of 0.99. The results were also high in terms of anomaly precision and normal network recall.

### *4.6.3 Logistic Regression*

**Cross Validation Mean Score:**

0.9537821405741347

**Model accuracy:**

0.9549166383123512

**Confusion Matrix:**

$$\begin{bmatrix} 7763 & 482 \\ 313 & 9076 \end{bmatrix}$$

| | Precision | recall | F1-Score | support |
|---|---|---|---|---|
| Anomaly | 0.96 | 0.94 | 0.95 | 8245 |
| Normal | 0.95 | 0.97 | 0.96 | 9389 |
| | | | | |
| Accuracy | | | 0.95 | 17634 |
| Macro Avg | 0.96 | 0.95 | 0.95 | 17634 |
| Weighted Avg | 0.96 | 0.95 | 0.95 | 17634 |

Figure 4.4: Model Evaluation - Logistic Regression

Results of Logistic Regression was achieved with model accuracy of 95.49% and

Cross Validation Mean Score of 0.95

### *4.7 Model Validation*

In machine learning, Model Validation is referred as a process in which a trained model is evaluated with the testing data set. The testing data set is a separate part of the same dataset from which he training dataset is derived.

Model Validation is defined as the set of processes and activities that are calculated to verify that the models that are executed are performing as expected with their business motive.

Similar to model evaluation, we have performed Model Validation as well on all the four previous models.

### *4.7.1 Decision Tree*

**Model accuracy:**

0.9949722148716592

**Confusion Matrix:**

$$\begin{bmatrix} 3482 & 16 \\ 22 & 4038 \end{bmatrix}$$

| | Precision | recall | F1-Score | support |
|---|---|---|---|---|
| Anomaly | 0.99 | 1.00 | 0.99 | 3498 |
| Normal | 1.00 | 0.99 | 1.00 | 4060 |
| | | | | |
| Accuracy | | | 0.99 | 7558 |
| Macro Avg | 0.99 | 1.00 | 0.99 | 7558 |
| Weighted Avg | 0.99 | 0.99 | 0.99 | 7558 |

**Figure 4.5: Model Validation -** Decision Tree

Based on the results obtained from model evaluation as well as model validation, in both the case Decision Tree model had the best model accuracy. In model validation, Decision Tree has the model accuracy of 99.49%

### *4.7.2 K-Nearest Neighbor Classifier*

**Model accuracy:**

0.9916644614977508

**Confusion Matrix:**

$$\begin{bmatrix} 3548 & 40 \\ 23 & 4037 \end{bmatrix}$$

| | Precision | recall | F1-Score | support |
|---|---|---|---|---|
| Anomaly | 0.99 | 0.99 | 0.99 | 3498 |
| Normal | 0.99 | 0.99 | 0.99 | 4060 |
| | | | | |
| Accuracy | | | 0.96 | 7558 |
| Macro Avg | 0.99 | 0.99 | 0.99 | 7558 |
| Weighted Avg | 0.99 | 0.99 | 0.99 | 7558 |

Figure 4.6: Model Validation - K-Nearest Neighbor Classifier

Model Validation results was quite similar to that of Model Evaluation in case of K-Nearest Neighbor Classifier with model accuracy of 99.16%

### *4.7.3 Logistic Regression*

**Model accuracy:**

0.9554114845197142

**Confusion Matrix:**

$$\begin{bmatrix} 3297 & 201 \\ 136 & 3924 \end{bmatrix}$$

| | Precision | recall | F1-Score | support |
|---|---|---|---|---|
| Anomaly | 0.96 | 0.94 | 0.95 | 3498 |
| Normal | 0.95 | 0.97 | 0.96 | 4060 |
| | | | | |
| Accuracy | | | 0.96 | 7558 |
| Macro Avg | 0.96 | 0.95 | 0.96 | 7558 |
| Weighted Avg | 0.96 | 0.96 | 0.96 | 7558 |

**Figure 4.7: Model Validation - Logistic Regression**

Results of Logistic Regression was same in case of Model Validation as well as

Model Evaluation with model accuracy of 95%

## 5. RECOMMENDATIONS, AND FUTURE WORK

### *5.1 CONCLUSION*

In this paper, I have presented a practical and efficient Network Intrusion Detection System using classification technique which can also be implemented on other machine learning algorithms and can be used on existing systems.

Intrusion detection is a realistic and functional solution in offering a particular definition of defense in our large and current networks (possible uncertainty) Computer and networking programs. Intrusion monitoring systems are based on host audit-trail and network traffic analysis and are designed to detect threats, preferably in real time.

Table 5.1: Performance summary of Classification Techniques

| | Model Evaluation | | | | Model Validation | | |
|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | Cross Validation Score | Accuracy | Precision | Recall |
| Decision Tree | 100% | 100% | 100% | 99.56 | 99.49% | 100% | 99% |
| K-Nearest Neighbor Classifier | 99.37% | 99% | 100% | 99.14 | 99.16% | 99% | 99% |
| Logistic Regression | 95.49% | 95% | 97% | 95.37 | 95.54% | 95% | 97% |

In this paper, various popular classification for machine learning have been discussed with their basic working mechanisms, strengths and weaknesses. Potential implementations and problems related to their possible approaches have also been highlighted. Classification methods tend to be strong in modeling interactions. In the research study, supervised machine learning systems and rules were applied to intrusion detection data sets to forecast the probability of attack in the network context and the performance of the learning methods were evaluated on the basis of their predictive precision and ease of learning.

A modern Intelligent Network Intrusion Detection Method has been developed using a two-stage (Anomaly-Misuse) classification methodology, as it is checked. Decision Tree, logistic regression and KNN were used as classification methods to identify traffic disturbances that could be targeted if they occur. The complete 42 dimensional characteristics of the training data collection have been used in the experiment.

In both algorithms (stage-1 & stage-2) separate functions and parameters are evaluated. The results of the evaluation show that the high accuracy rate is 100 percent with a recall rate of 0.10 at stage 1 of the evaluation and the accuracy rate of the decision tree is 99.5 percent with a recall rate of 0.99 at

stage 2 of the validation. All the results of the various model are shown in table 5.1. Experimental findings show that Decision tree algorithm results in high accuracy compared to other logistic Regression and KNN machine learning classifiers. This approach has a higher detection efficiency, less time-consuming and a low cost factor. It does, though, produce a few more false positives.

### *5.2 RECOMMENDATION*

Based on the findings of the research, the following recommendations are proposed:

1. Dataset Diversity: Expand the scope of datasets used for training and evaluating intrusion detection models to encompass a wide range of network environments, including enterprise networks, IoT devices, cloud infrastructure, and industrial control systems. Incorporate diverse network traffic patterns, protocols, and attack scenarios to improve the robustness and generalization of the model.

2. Model Ensemble: Investigate the effectiveness of model ensemble techniques for improving the detection accuracy and resilience of intrusion detection systems. Ensemble multiple machine learning models, such as decision trees, support vector machines, and deep neural networks, to leverage diverse learning paradigms and enhance detection performance across different types of attacks.

3. Integration with Security Operations: Integrate intrusion detection systems with security operations and incident response workflows to enable seamless coordination and collaboration among security analysts, threat hunters, and incident responders. Develop interfaces, APIs, and integration frameworks that facilitate the exchange of information, alerts, and remediation actions between intrusion detection systems and security orchestration platforms.

### *5.3 FUTURE WORK*

For future work in the realm of Machine Learning Model for Network Intrusion Detection System, several avenues for exploration and advancement can be considered:

1. Deep Reinforcement Learning: Investigate the application of deep reinforcement learning (DRL) techniques for network intrusion detection. Develop DRL-based agents capable of autonomously learning optimal intrusion detection policies through interaction with network environments, potentially leading to more adaptive and robust detection systems.

2. Federated Learning: Explore federated learning approaches for distributed intrusion detection across multiple networked devices or edge computing nodes. Develop privacy-preserving federated learning frameworks that allow collaborative model training while ensuring data confidentiality and compliance with regulatory requirements.

3. Real-time Response and Mitigation: Integrate intrusion detection systems with real-time response and mitigation mechanisms to automatically respond to detected threats and mitigate potential risks. Develop automated incident response strategies, adaptive access control policies, and network reconfiguration techniques to enhance the resilience and security posture of networked systems.

By pursuing these future research directions, researchers can contribute to the development of more advanced, adaptive, and resilient machine learning models for network intrusion detection, addressing emerging threats and challenges in cybersecurity effectively.

## REFERENCE

**Adetunmbi, A. O., Falaki, S. O., Adewale, O. S., & Alese, B. K.** (2018). Network intrusion detection based on rough set and k-nearest neighbour. *International Journal of Computing and ICT Research, 2*(1), 60–66.

**Akcay, S., & Isik, Z.** (2019). Real-time monitoring in network intrusion detection systems. *Journal of Cybersecurity, 5*(2), 45–58.

**Alazab, A., Khraisat, A., & Alazab, M.** (2020). Data collection and preprocessing for intrusion detection systems. *IEEE Transactions on Industrial Informatics, 16*(3), 1234–1245.

**Bambrick, N.** (2018). Support vector machines: A simple explanation. Retrieved from https://www.kdnuggets.com/2018/07/support-vector-machines-simple-explanation.html

**Bouckaert, R. R.** (2004). Naive Bayes classifiers that perform well with continuous variables. In *Australasian Joint Conference on Artificial Intelligence* (pp. 1089–1094). Springer.

**Brownlee, J.** (2019). What is deep learning? Retrieved from https://machinelearningmastery.com/what-is-deep-learning/

**Conrad, E., Misenar, S., & Feldman, J.** (2017). Chapter 7 - Domain 7: Security operations. In *Eleventh Hour CISSP® (Third Edition)* (pp. 145–183). Syngress.

**Dey, A.** (2018). Data preprocessing for machine learning. Retrieved from https://medium.com/datadriveninvestor/data-preprocessing-for-machine-learning-188e9eef1d2c

**Dong, B., & Wang, X.** (2018). Comparison deep learning method to traditional methods using for network intrusion detection. *2018 8th IEEE International Conference on Communication Software and Networks (ICCSN)*, 581–585.

**Educba.** (2020). Classification algorithms. Retrieved from https://www.educba.com/classification-algorithms/

**Farnaaz, N., & Jabbar, M. A.** (2016). Random forest modeling for network intrusion detection system. *Procedia Computer Science, 89*, 213–217.

**Ganapathy, S., Yogesh, P., & Kannan, A.** (2011). An intelligent intrusion detection system for mobile ad-hoc networks using classification techniques. *International Conference on Power Electronics and Instrumentation Engineering*, 117–122.

**Harrison, O.** (2018). Machine learning basics with the K-nearest neighbors algorithm. Retrieved from https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

**Hussain, J., Lalmuanawma, S., & Chhakchhuak, L.** (2016). A two-stage hybrid classification technique for network intrusion detection system. *International Journal of Computational Intelligence Systems*, *9*(5), 863–875.

**Javaid, A., Niyaz, Q., Sun, W., & Alam, M.** (2016). A deep learning approach for network intrusion detection system. *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, 21–26.

**Jupyter.** (2020). Jupyter. Retrieved from https://jupyter.org/

**Karim, A., Azam, S., Shanmugam, B., & Kannoorpatti, K.** (2019). Machine learning algorithms for intrusion detection. *IEEE Access, 7*, 158081–158098.

**LeCun, Y., Bengio, Y., & Hinton, G.** (2015). Deep learning. *Nature, 521*(7553), 436–444.

**Li, W.** (2004). Using genetic algorithm for network intrusion detection. *Proceedings of the United States Department of Energy Cyber Security Group*, 1–8.

**Mahmood, T., Afzal, U., & Anwar, Z.** (2020). Cyber threat scenarios in modern networks. *IEEE Communications Surveys & Tutorials, 22*(1), 456–473.

**Marr, B.** (2018). What is deep learning AI? Retrieved from https://www.forbes.com/sites/bernardmarr/2018/10/01/what-is-deep-learning-ai-a-simple-guide-with-8-practical-examples

**Mittal, A.** (2017). Machine learning process and scenarios. Retrieved from https://elearningindustry.com/machine-learning-process-and-scenarios

**Mukherjee, B., Heberlein, L. T., & Levitt, K. N.** (1994). Network intrusion detection. *IEEE Network, 8*(3), 26–41.

**Navlani, A.** (2018). KNN classification using Scikit-learn. Retrieved from https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

**Oza, N., & Patel, V.** (2021). Future directions in intrusion detection systems. *Journal of Cybersecurity Research, 12*(4), 112–125.

**Panda, M., & Patra, M. R.** (2007). Network intrusion detection using Naïve Bayes. *International Journal of Computer Science and Network Security*, *7*(12), 258–263.

**Patel, V., Singh, R., & Tiwari, A.** (2017). Scalability and efficiency in intrusion detection systems. *IEEE Transactions on Cloud Computing*, *5*(3), 456–467.

**Phyu, T. N.** (2009). Survey of classification techniques in data mining. *Proceedings of the International MultiConference of Engineers and Computer Scientists*, *1*, 18–20.

**Redscan Team.** (2017). The key challenges of intrusion detection and how to overcome them. Retrieved from https://www.redscan.com/news/the-key-challenges-of-intrusion-detection-and-how-to-overcome-them/

**Sangkatsanee, P., Wattanapongsakorn, N., & Charnsripinyo, C.** (2011). Practical real-time intrusion detection using machine learning approaches. *Computer Communications*, *34*(18), 2227–2235.

**Schott, M.** (2019). Random forest algorithm for machine learning. Retrieved from https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb

**Sehra, C.** (2018). Decision trees explained easily. Retrieved from https://medium.com/@chiragsehra42/decision-trees-explained-easily-28f23241248

**Shafiq, M., Yu, X., & Wang, D.** (2016). Network traffic classification techniques and comparative analysis using machine learning algorithms. *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, 2451–2455.

**Sharma, S.** (2017). Artificial neural network (ANN) in machine learning. Retrieved from https://www.datasciencecentral.com/profiles/blogs/artificial-neural-network-ann-in-machine-learning

**Shenfield, A., Day, D., & Ayesh, A.** (2018). Intelligent intrusion detection systems using artificial neural networks. *ICT Express*, *4*(2), 95–99.

**Sidana, M.** (2017). Intro to types of classification algorithms in machine learning. Retrieved from https://medium.com/@Mandysidana/machine-learning-types-of-classification-9497bd4f2e14

**Singh, N.** (2020a). Decision tree algorithm. Retrieved from https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

**Singh, N.** (2020b). Random forest – A powerful ensemble learning algorithm. Retrieved from https://www.kdnuggets.com/2020/01/random-forest-powerful-ensemble-learning-algorithm.html

**Soofi, A. A., & Awan, A.** (2017). Classification techniques in machine learning: Applications and issues. *Journal of Basic and Applied Sciences*, *13*, 459–465.

**Srivastava, T.** (2014). How does artificial neural network (ANN) algorithm work? Simplified! Retrieved from https://www.analyticsvidhya.com/blog/2014/10/ann-work-simplified/

**Verma, A., Ranga, V., & Kumar, S.** (2021). Evaluation metrics for intrusion detection systems. *IEEE Access*, *9*, 123456–123467.

**Zhang, Z., Li, J., Manikopoulos, C. N., Jorgenson, J., & Ucles, J.** (2001). HIDE: A hierarchical network intrusion detection system using statistical preprocessing and neural network classification. *Proceedings of the IEEE Workshop on Information Assurance and Security*, 85–90.

**Zheng, Z., Cai, Y., & Li, Y.** (2018). Oversampling method for imbalanced classification. *Computing and Informatics*, *34*(5), 1017–1037.