COVER PAGE

AN ENHANCED THREAT CLASSIFICATION FRAMEWORK FOR ELECTRONIC HEALTH SYSTEMS IN NIGERIA

EDITH ABENGOWE

B.Sc. (HONS.) (NOUN), M.Sc. (NOUN)

ACE22250014

AUGUST, 2024

TITLE PAGE

AFRICA CENTRE OF EXCELLENCE FOR TECHNOLOGY ENHANCED LEARNING (ACETEL)

AN ENHANCED THREAT CLASSIFICATION FRAMEWORK FOR ELECTRONIC HEALTH SYSTEMS IN NIGERIA

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN

CYBERSECURITY

BY

EDITH ABENGOWE

ACE22250014

SUPERVISORS:

1. **Dr. Uyinomen Ekong**
2. **Dr. Saheed Kayode**
3. **Mr. Felix Rishamma**

SEPTEMBER, 2024

DECLARATION

I, Edith Abengowe, the author of this dissertation entitled "An Enhanced Threat Classification Framework for Electronic Health Systems in Nigeria", affirm that it is my original work and that it has not been submitted, in whole or in part, in any previous application for a degree and I will not present it, or cause it to be presented, for a degree in another institution. All sources of information have been appropriately acknowledged using references and other acceptable methods.

NAME: EDITH ABENGOWE

SIGNATURE: _____

DATE: _____

CERTIFICATION

This is to certify that this dissertation: "An Enhanced Threat Classification Framework for Electronic Health Systems in Nigeria" submitted by Edith Abengowe (ACE22250014), in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Cybersecurity, meets the regulations governing the award of PhD degree of the Africa Centre of Excellence and Technology Enhanced Learning (ACETEL). The work has made original contribution to knowledge. It has been examined and approved by the following Supervisors:

(a) Supervisor

Signature……. Date: ……

Name:…….. Rank:……

(b) Co-Supervisor

Signature……. Date: ……

Name:…….. Rank:……

(c) Co-Supervisor

Signature……. Date: ……

Name:…….. Rank:……

DEDICATION

This dissertation is dedicated to my spouse, Emmanuel Abengowe for believing in me and standing by me throughout the writing of this dissertation and to my children, Barbara and Pearl, for their support and understanding.

ACKNOWLEDGEMENTS

My profound gratitude goes to Almighty God for His faithfulness and grace, and for granting me the wisdom, strength, and perseverance to complete this research.

I would like to express my deepest gratitude to Dr. Uyinomen Ekong, my Supervisor, for her guidance, support, and invaluable feedback throughout this research. I am also grateful to Mr. Felix Rishamma, my Co-Supervisior, for his insightful comments and suggestions.

My sincere appreciation also goes to the staff of ACETEL for their academic and administrative support during my study. I would also like to thank my HOD, Dr. Adeyinka Abiodun for her valuable comments and support that greatly enhanced the research process. Each time I call, she responds.

My heartfelt thanks go to my family and friends, especially Aisha Adamu, Charity Nuhu (my HOU), Aisha Ali, Faith Azemobor, and Esther Lasisi for their unwavering support and encouragement. I would not fail to mention my amazing boss (Dr. Olayemi Nwankwo) for being my backbone during the course of this research.

To everyone who has supported me along this journey, thank you. This achievement would not have been possible without you.

Finally, I extend my appreciation to all those who assisted in getting materials for this study, particularly Dr. Oyong, you came through for me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACL	Access Mitigate List
AES	Advanced Encryption Standard
AP	Access Point
AP	Aptoide
API	Application Programming Interface
APK	Android Application Package
ARP	Address Resolution Protocol
ART	Android Run Time
BLE	Bluetooth Low Energy Protocol
BMP	Bit-map Image Format
BSS	Basic Service Set
CRC	Cyclic Redundancy Check
DCT	Discrete Cosine Transformation
DDK	Knowledge Discovery and Datamining
DES	Data Encryption Standard
DEX	Delvik Executable Format
DFT	Discrete Fourier Transform Technique
D-H	Diffie-Helman algorithm
DPI	Deep Packet Inspection
DS	Distributed system
DWT	Discrete Wavelet Transformation
EAP	Extensible Authentication Protocol
ESS	Extended service set
FIPS	Federal Information Processing Standard

FTP	File Transfer Protocol
HAS	Human Auditory System
HMAC	Hash Message Authentication Code
HVS	Human Visual System
IANA	Internet Assigned Network Authority
IBSS	Independent basic service set
ICMP	Internet Mitigate Message Protocol
ICV	Integrity Checksum value
IdP	Identity Provider
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronic Engineering
IoT	Insecurity of Things
IoT	Internet of Things
IP	Internet Protocol
IV	Initialization vector
JPEG	Joint Photographic Experts Group
LSB	Least Significant Bits
MAC	Media Access Control
MD2	Message Digest 2
MIC	Message Integrity Check
MITM	Man-In-The-Middle attack
MPD	Multi-pixel Differencing
MPEG	Moving Picture Experts Group
MSE	Mean Square Error
NIC	Network Interface Card

NLP	Natural Language Processing
OTN	Optical Transport Network
OTP	One Time Password
Ping	Packet Internetwork reachability to the target
PRNG	Pseudorandom Number Generator
PSNR	Peak Signal to Noise Ratio
PSTN	Public Switched Telephone Network
PVD	Pixel Value Differencing
RFP	Request for proposal
RGB	Red, Green, and Blue values
RMSE	Root Mean Square Error
RSA	Rivest-Shamir-Adleman algorithm
RTU	Remote Terminal Unit
SaaS	Software as a Service
SAML	Security Assertion Markup Language
SAT	Site Acceptance Test
SHA	Secure Hash Algorithm
SMS	Short Message Services
SNMP	Simple Network Monitoring Protocol
SSID	Service Set Identifier
SSIM	Structural Similarity Index
SSL	Secure Socket Layer
SSO	Single sign on
STK	Short Term Key
TDR	Time Domain Reflectometer

TK Temporary Key

TKIP Temporary key integrity protocol

VPN Virtual Private Network

VS Virus Share

VSM Vector Space Model

WAP Wireless Access Protocol

WEP Wired Equivalent Privacy protocol

Wi-Fi Wireless Fidelity

WPA Wi-Fi Protected Access

WS Web service

WSN Wireless Sensor Network

XACML Extensible Access Control Markup Language

XML Extensible Markup Language

LIST OF APPENDICES

PUBLISHED WORK

ABSTRACT

The security of healthcare systems and patients' records privacy are constantly threatened by malware through attacks on them and expose them to ridicule, embarrassment and possible litigation on hospital management for breach of ethics on the handling of patients' records. Attack agents include viruses, worms, Trojans such as ransomware, key loggers and rootkits; while attack types include denial of service (DOS), probe, remote to local (R2L) and user to root (U2R). To curb the menace of malware threats, this research work developed a Framework using machine learning (ML) tools, combined them using ensemble learning technique such as bagging. The dataset used is Network Security laboratory – knowledge discovery in databases (NSL-KDD), obtained from Kaggle, a public data repository. It is categorized into normal class and attack types. This dataset is imbalanced with normal features being much more than attack types, especially PROBE, R2L and U2R types. For effective performance of the trained models, the dataset had to be balanced using Synthetic minority oversampling technique (SMOTE) and extracted using Pearson correlation and information gain, then normalized and selected using principal component analysis (PCA). The preprocessed data is then split into training dataset (80%) with 125,973 records and test dataset (20%) with 22,544 records using train_test_split() function. ML tools used include Random Forest algorithm that trains base classifiers such as k_Nearest_Neighbors (KNN), Naïve Bayes (NB), Decision tree (DT), and Logistic regression (LR). The predictions of trained classifiers are aggregated using majority voting scheme into soft vote classifier, which is used to classify test dataset into normal or malware labels. The actual and predicted results are compared using confusion matrix. The models are then evaluated for performance using cross validation parameters such as accuracy, precision, recall, f1-score and area under the Receiver operating characteristic curve (AUC-ROC). The comparative analysis of different models reveals significant differences in their performance metrics. The Random Forest Classifier outperforms other models with the highest accuracy of 0.99932 and an exceptional ROC-AUC score of 1.0, indicating its superior ability to distinguish between classes. The KNN model also shows strong performance, with an accuracy of 0.98816 and a ROC-AUC of 0.99740. In contrast, the Naive Bayes classifier demonstrates much lower performance across all metrics, with an accuracy of 0.39624 and a ROC-AUC of 0.7419, suggesting that it struggles with the dataset's complexity. The Decision Tree Classifier performs nearly as well as the Random Forest with an accuracy of 0.99804 and a ROC-AUC of 0.9988, though its precision score appears to be incorrectly reported, matching that of the Naive Bayes Classifier. Overall, the Random Forest Classifier and Decision Tree Classifier exhibit the best performance, making them the preferred choices for this classification task. In addition, the results of this research work are compared with results of other works in literature that used multiple datasets to train and evaluate a set of models. Their results were outperformed by this research work, even when they used more modern datasets.

CHAPTER ONE
INTRODUCTION

1.1 Background of the Study

In recent times, challenges confronting the world include infectious diseases (including COVID-19 Pandemic), environmental health hazards, hunger, and crime prevention (Sardi *et al,* 2020). One of the main challenges in the health sector is cyber threat. A threat could be defined as the combined probability of an unwanted event and its impact on the target object (Sardi *et al,* 2020). Another school of thought defined threat as potential event, intentional or otherwise, that compromises security attributes such as confidentiality, integrity, availability, authentication, access control and non-repudiation. (Tatam *et al,* 2021; HIPAA, 2024). Threat can also be defined as different kinds of likely actions that are caused by either stilt or natural means against a functional system (Idris *et al*, N.D).

A Framework is a graphical representation of the system being developed that shows the input, logical internal processes, and output (Tatam *et al,* 2021). While an electronic health system (EHS) is defined as a digital version of the paper-based system, which replaces the manual recording process with electronic healthcare records (EHR) (Hathaliya and Tanwar, 2020). EHS offers economic benefits such as efficiency, data management and administration of patients care system. It also enables the sharing of information between patients and their health care providers. This improves diagnosis, increases patient education and promotes personalized healthcare (Fatima and Colomo-Palacious, 2018). Also, EHS provides collaboration on patient care and emergency care. In addition, it gathers patients' data and stores in EHR.

However, EHS has become common targets for ransomware attack, crypto mining, data theft, phishing, spoofing and insider threats. Cyber-attacks are becoming sophisticated by the day with cyber threat actors exploiting the vulnerability that should have been patched before to avoid intrusions. A cyber threat actor is a person or entity that undermines security attributes. The neglect is not unconnected with non-adherence to security best practices, due to budgetary pressure, few skilled information technology (IT) professionals in the health sector, and ignorance or indecision of the most effective way to provide resilience to cyber threats (HIPAA, 2024). What is more worrisome is the scale of these data breaches, necessitating an urgent action to curb the menace.

Security is an important sector in information processing, especially in critical settings like EHS. Cloud Computing Technology (CCT) provides the enabling environment for the classification of threats at lower cost (Yeng *et al,* 2020). CCT is very useful to health care organizations, for it helps to focus on their therapeutic services. The benefits of using CCT for EHS include:

i. Easy collaboration and data sharing
ii. Mobility
iii. Cost reduction on information and communication technology (ICT) services
iv. "Pay for what you use", instead of owning structures and equipment
v. Scalability
vi. Business continuity
vii. Flexibility, and
viii. Strategic values.

These benefits not-with-standing, CCT does have shortcomings. The security challenges include (Fan *et al.,* 2024):

i. Data loss or leakage
ii. Transaction hijacking
iii. Insecure user interface
iv. Denial of service (DOS) attacks
v. Abuse of cloud services
vi. Malicious insider attack
vii. Data breaches
viii. Virtual machine (VM) escape
ix. Unauthorized access to management interface
x. Metering and billing systems
xi. Vendor lock-in

In vendor lock-in, a client can move its data and services to a cloud service prodder (CSP) but cannot easily change the vendor without severe cost, legal constraint or technical incompatibility (Yeng *et al.,* 2020). The menace of malicious programs threatening the healthcare system needs to be curbed, otherwise healthcare infrastructure may lose its relevance. For instance, healthcare and public health institutions had 23% ransomware attacks in 2021 (FBI internet crime report, 2021). In addition, healthcare is among the four organizations susceptible to multi-vector attacks (CBD report, 2021). Similarly, Verizon DBIR (2022) reported web application attacks and system intrusions as top cyber-attack patterns in healthcare industries. In 2023, the healthcare system suffered eleven (11) breaches in the United States of America (USA). These incidences took place in hospitals, insurance companies working with the

hospitals, and vendors working with medical facilities; up to 70 million people's medical records were affected (HIPAA, 2024).

The health sector depends on inter connected information systems, as such new security vulnerabilities are opened, making the industry a prime target for cybercrime. This puts patient data, underlying infrastructure and lives at risk. Traditional security measures often fail to keep pace with evolving threats. Some threats include phishing, spoofing, malicious insider, ransomware, data breaches, and more. These breaches, among others, are generally grouped into four attack types:

i. Denial of service (DOS) attacks
ii. Probe or surveillance attack
iii. User to root (U2R) attack
iv. Remote to local (R2L) attacks

These attack types can shatter the trust patients have for medical or healthcare institutions, if not controlled. Modern medical devices operate in networked clinical environment, and can attract data breaches, disruption to clinical operations, and invasion of patients' privacy and safety (Aijaz *et al.,* 2023). This research work intends to develop a framework that will train models, which will classify security threats in electronic health care systems into normal and abnormal types.

Machine learning (ML) tools such as Decision Tree (J48), Naïve Bayes (NB), Logistic Regression (LR), and K-Nearest Neighbor (KNN) will be trained by Random Forest, the algorithm, using National Science Laboratory-Knowledge Discovery in Databases (NSL-KDD) dataset. The trained models will learn the characteristics of normal applications and use that knowledge to classify malware from normal applications

(Abushark *et al.,* 2022). Abnormal applications also called malware include viruses, worms, Trojan horses, spyware, adware, ransomware, key loggers, among others.

Viruses are computer programs that attach themselves to executable programs (.EXE, .BAT, and .COM). They depend on the intervention of the user to spread. A worm, unlike the virus, is a computer program that does not depend on the user to spread. It replicates itself as it moves from one network to another. Trojan horse is a fake computer program that masquerades as a legitimate program but contains a malicious code that activates and runs when the need arises to attack the host system. Spy-ware or probe program appears to be dormant, and does not directly cause any harm on the host system. It rather stealthily gathers information, personal data, and vulnerable points of the affected host system or network and send back to the command and control (C&C) server. The data gathered usually include credit/debit card numbers, personal data like name, address, password, and so on. Adware are more or less inconvenient applications that pop-up on the screen unannounced, indeed, they are unwanted adverts. They are difficult to do away with.

The dataset to use in this research work is NSL-KDD dataset. It consists of normal and abnormal applications, grouped into DOS, PROBE, U2R and R2L. It is divided into training dataset (80%) and test dataset (20%) with forty-one attributes, including labels. Python programming language, in collaboration with its external modules like scikit learn, pandas, and so on, will be used to manipulate the dataset to produce results.

1.2 Statement of the Problem

In critical healthcare systems, network servers provide storage, communication channels and other services for the improvement of patient care using patient health records (PHR). Actors such as physicians, nurses, pharmacists, laboratory technicians

and patients access these assets in PHR using cloud computing technology However, the main challenges of PHR are security and patient privacy (Akram *et al.,* 2021). EHS security and privacy issues include (Singh and Chatterjee, 2019):

i. Unauthorized Access
ii. Impersonation attack through spoofing
iii. Malicious insider

1.3 Aim and Objectives

1.3.1 The Aim of the Study

The aim of this research work is to develop a framework using machine learning tools, which will classify malware threats from normal applications, in CCT platform.

1.3.2 Specific Objectives of the Study

The specific objectives of the study are to:

i. Build a framework using Random Forest algorithm to train base models such as Decision Tree (J48), Naïve Bayes (NB), Logistic regression (LR) and K-Nearest Neighbor (KNN) to learn the characteristics of normal applications and use that knowledge to predict labels of normal applications and malware attack types.
ii. Aggregate the models' predictions using VotingClassifier() and determine a consensus classifier called Hard vote classifier, which will be used to detect intrusions in test dataset.
iii. Use anomaly intrusion detection system (IDS) also called dynamic IDS, to analyze a specific point in the network or host system and detect threats; and set aside privileged users such as system administrators, computer operators or contractors.

These objectives are achieved in methodology.

1.4 Methodology

Electronic healthcare system and Patients Health Records (PHR) are central to patient care system. They are used to reduce morbidity and mortality of patients in hospital management. To curb the menace of security and privacy threats on them, the following steps are put in place:

i. Download and use NSL-KDD dataset to train and test the models for generalization. The dataset is gotten from Kaggle, a public data repository.

ii. Preprocess the dataset using selection techniques such as SMOTE, Pearson correlation (PC) and Information gain (IG); and then extract impurities using min-max normalization to reduce the features' dimensions to a range, say [0,1]; and remove redundant, noisy and missing data using principal component analysis (PCA)

iii. Convert symbolic (text) features such as protocol type, Services and flags to crisp values using One-hot-encoder() and label-encoder().

iv. Split the dataset into training dataset (80%) with 125,973 records; and test dataset (20%) with 22,544 records using train-test-split() function in a ratio of 80:20

v. Train the models such as DT, NB, LR in parallel using RandomForest(); and compute the shortest distance from k (number of nearest neighbors) in KNN to assign the target object.

vi. Predict the labels using each trained classifier with model.predict() function, then aggregate the predictions of the models and that of KNN into a consensus classifier, soft vote.

vii. Predict the labels using soft vote classifier

viii. Compare the predictions of the soft vote with the actual labels using confusion matrix (being a supervised learning problem)

ix. Generate results

1.5 Significance of the Study

The timeliness of this research work cannot be over emphasized, given the fact that electronic healthcare system (EHS) and its subsidiary, electronic health records (EHR) are very crucial in promoting health care, security and patients' privacy. EHR contains sensitive patient's data such as medical history, treatment records, and personal identification details such as name, address, password, credit card number, debit card number, and more. EHS and EHR deserve to be protected from malicious programs and hackers to avoid privacy breach, identity theft, and security compliance penalty.

Unfortunately, security and privacy attacks are constantly being launched against hospital management. Some of these challenges include

i. Unauthorized access, which could result from neglect and carelessness of physicians and computer operators

ii. Vulnerabilities of both hardware and software resulting from misconfiguration, system bugs and spyware attacks

iii. Data integrity, which could result from dynamic code injection, hijacking of processes, and phishing.

iv. Data collaboration between systems, organizations and governments, has the good side and bad side. Data collaboration has the advantage of sharing patients' records between authorized persons; however, the more data

moves between organizations so also are vulnerable points not only increasing but exploited by threat actors.

The solution to this problem is to adopt standardized protocols and ensure that data exchange formats are adhered to.

For instance, in 2018, OptimumHealth experienced a data breach caused by phishing, thereby exposing personal information of 10 million patients. Such information includes patient's name, address, date of birth, credit/debit card numbers and insurance information. (Singh and Chatterjee, 2019). Also, in 2015, Anthem, one of the health insurance companies in the US, experienced a hacking attack that affected over 78 million people. The hackers stole patients' information such as names, addresses, social insurance and medical information. These breaches highlight the need to protect EHR data from unauthorized access, disclosure, alteration or destruction. Some solutions to these breaches can be provided by the user, while this research work handles cyber security and patient's privacy to safe guard lives and property for the interest of stake holders, include:

i. Adoption of robust user authentication protocol such as two-factor authentication to reduce the risk of unauthorized access. Traditionally, password and user name are used, which are vulnerable and easy to break using brute force technique. Biometric identification is much harder to break than password.
ii. Update anti-virus programs to the latest version, this will include the latest vulnerability patches supplied by Google, for those using Android platform.
iii. Backup your data to reduce the risk of unforeseen data lose or host system crashing.

iv. Training both clinical and non-clinical staff of the healthcare organization to know what to do in the event of an attack; the significance of data privacy and the implication of data breaches.

v. Encryption protects the integrity and confidentiality of information stored in EHR.

In this research work, these challenges are addressed using machine learning ensemble classification techniques. The solution to these breaches will provide confidence and integrity in the patients, hospital management, insurance community and the general public. These are indeed, the stake holders in hospital management, with the objective of saving lives.

1.6 Scope and Limitations

This research work is delimited to the following intents:

i. Electronic Health System (EHS) has many security issues, but only unauthorised access, vulnerability, data integrity and confidentiality are considered; also considered is patient's privacy threat. Other security threats such as probe, non-repudiation, encryption and ransomware are not considered

ii. Security attributes covered in this research work include Confidentiality, Integrity, Availability, Other attributes such as confidence Index, Client App., Source IP, Date/Time, Object of Attack type, Object Name, and more, are not considered.

iii. The models are trained to learn the behaviour and characteristics of 67,343 normal applications, and use that knowledge to classify applications that

deviate from the norm as malware. Malware family signatures and their databases are not considered.

iv. Bagging technique, through Random Forest Algorithm, is used to train and combine multiple classifiers (models) such as DT, NB, SVM and KNN to solve the problem of malware threats. Other techniques such as hybridization, boosting and stacking are not considered.

1.6.1 Limitations

The framework in this research work cannot directly be deployed on desktop, PCs or mobile devices because of the following that will not be part of this research work.

i. The framework will not be integrated to application programming interface (API) for uploading to the cloud, and the front-end interface will not be designed and developed to enable linking with the backend.

ii. Other limitations include:

- Undocumented APIs and Applications from third party stores may still find their way to the device because Android devices accept them (70% of which are malware or faked applications).
- Dynamic analysis monitors one device component at a time in real-time for deviations from the norm by applications manipulating it and report as malware; other components are not considered at the time.

1.7 Definition of Terms

To flow with the current write-up, it is normal to define some of the terms that will be encountered in the course of reading the thesis.

Attack: In the context of computer and network security, an attack is an attempt to access resources on a computer or network without the necessary authorization or to bypass security measures put in place.

Artificial Intelligence: This is the ability of a computer to learn human expertise or cognitive function, builds that into a device to function or assist experts or users in this field. For example, some cars are capable of providing security for themselves devoid of human intervention.

Audit: This is to track security related events, such as logging onto the system or network, accessing objects or exercising user or group rights or privileges.

Backdoor: This is an opening or break in the software, hardware or network usually for debugging purposes. Malware code (Rootkit) is a malware that installs itself on the system and assumes the privileges of the system administrator, and paves the way for more malicious codes to be installed on the said system.

Botnet: This is a collection of internet connected devices and systems that have been compromised by malware infections. They constitute peer-to-peer (P2P) system with one of them serving as a proxy to send stolen information to the botmaster (or command and control server).

Breach: Successfully defeating security measures to gain access to data or resources illegally or illegitimately. Also, making data or resources available to unauthorized persons or delete or manipulate computer files.

Brute force attack: This is an attempt to crack passwords by sequentially trying all possible combinations of characters until the right combination grants access.

Buffer overflow: This is an attempt to crash the system by putting more data than the buffer can accommodate.

Buffer: A holding area for data

CIA triad: This stands for Confidentiality, Integration, and Availability of data and services.

Classifiers: These are trained models used to classify a binary data into normal or anomalous types

Confidentiality data: This ensures that the content of data is kept secret

Counter measures: Steps taken to respond to an attack or anomaly.

Cracker: A hacker who specializes in cracking or revealing or discovering system passwords in order to gain access illegally to a computer or network system.

Crash: Sudden failure of a computer system, rendering it unstable.

Data availability: This has to do with reliable and timely access to data (24/7), anytime, anywhere.

Defense in dept.: The practice of implementing multiple layered security.

Denial of service attack: A deliberate way of bashing the target system or resource with request, so much such that legitimate users would be starved of its services.

Exposure: A measure of the extent to which a computer or network is open to attack, based on its particular vulnerability, how well known it is to the hackers, and the time of operation.

Hacker: An expert computer programmer, who spends his or her time writing programs that do negative intents of his or hers.

Integrity of data: This is a security characteristic which ensures that the data sent or received is not modified.

Intelligent: The intelligence in this case is referring to systems or devices learning ability of human expertise, and using that knowledge to assist experts in that domain.

Intrusion Detection: This is the process of monitoring host systems, mobile devices or network for abnormal behavior of applications manipulating the device component. Intrusion is an unlawful entry into the device or unlawful manipulation of a device.

Least privilege: The job schedule of an employee, the level of access to the system resources and information granted to such an employee.

Machine learning: This is the process whereby computers learn features of data or human expertise and uses that knowledge to recognize similar features that may not be labeled.

Malicious code: A computer program or script that performs actions inimical to the system, but satisfies the master's evil intent.

Mobile Device: This is a handheld device used to communicate, and do other things like buying and selling online, pay bills, make transfers, using the internet.

Penetration testing: Evaluating a system by attempting to circumvent the computer's or network's security measures.

Reliability: The probability of a computer system or network continuing to perform satisfactorily under normal operating circumstances.

Risk management: The process of identifying, controlling, and mitigating the problem militating against data confidentiality, data integrity, and system availability.

Risk: The probability that a certain security threat will exploit the system's vulnerability (weakness) resulting in damage, loss of data or other undesired effects.

Sniffer: Packet sniffer is a program that captures data in transition across the network, for nefarious intents, in most cases.

Social engineering: This is a process whereby malicious programs persuade users to either click an infectious site, download or install a program, the consequences in most cases, he or she may not know.

Threat: This is a potential danger to data or system or even the network. Threat agents are employed to cause the threat, they include: Trojan, virus, worms, spyware, adware, etc.

Vulnerability: This is a flaw or bug or weakness in the software or hardware component of a system that can be exploited by the "bad boys" to their advantage.

Zero-day vulnerability: This is an undisclosed defect or flaw, which attackers can exploit. It is called zero-day because it is not publicly reported before becoming active.

Zombie: This is a compromised computer, which is usually connected to the network, the malware agents that infect such system could be a virus, worm, Trojan, etc. It can be used to perform malicious tasks, and in most cases made to join the botnet family.

1.8 Organization of the Thesis

In this research work, chapter one introduces electronic health system (EHS) and one of its subsections, electronic health records (EHR) among other sub types of EHS. Their cyber problems, and in particular security and patients' records privacy motivated the researcher to design and implement a framework that will classify the threats ravaging the health sector. Chapter two presents the detailed literature review, especially, specific words (terms or phrases) in the thesis title and other related topics in literature. The chapter also reviews related literature and compares their strengths and weaknesses in relation to the current work being developed. In chapter three, the method used to develop the framework, which is ensemble learning technique, is discussed. Five classifiers are trained and aggregated to produce a consensus model, called soft vote, which is used to classify the test dataset for generalization and performance. Random forest (RF), decision tree (DT), K-nearest neighbor (KNN), Naïve Bayes (NB) and logistic regression (LR) are the classifiers used in the development of the framework. Chapter four presents the results of the threat detection and the performances of the

trained models, and that of the soft vote classifier. Confusion matrix is used to compare the expected labels and the predicted labels to ascertain the true positive rate, the true negative rate, the false positive and false negative values of each model. A discussion was also put in place that compares the relationship of the models in terms of accuracy, precision, recall, f1 score and area under the curve (AUC) of ROC values. In chapter five, the work is summarized, concluded and our contribution to knowledge duly stated. The problems encountered during the framework development are reduced to future h works.

CHAPTER TWO
LITERATURE REVIEW

2.1 Preamble

Electronic Health System (EHS) is the digital use of information and communications (ICT) to support health and health related fields. It includes health information techniques such as (Alobo *et al.,* 2020):

i. Patient health information and data
ii. Clinical decision support system
iii. Results management and central data repository
iv. Computerized physician order entry (CPOE)

These health systems are used to support the organization in the care and delivery of health services (Alobo *et al.,* 2020). For instance, they improve the use of physician time, improve patients' safety, improve efficiency in health management, enable the exchange of information, patient data and medical images electronically, and assist physicians in reducing patients' morbidity and mortality rates.

EHS is applied in the following areas:

i. National Health Insurance Scheme (NHIS) management
ii. Cashless payment integration
iii. Electronic identification of patients
iv. Bed assignment
v. Laboratory orders, reporting and consumables
vi. Easy and quick retrieval of patient's records

The challenges of EHS include:

i. Threat to patient's privacy
ii. Information overload
iii. Power outages, among others.

In addition, electronic health records (EHR) is the digital collection of patients records in a central database, which can be shared across different health settings to provide collaboration, real-time decision support, and permanent patient health records. Although these capabilities improve health care quality, their vulnerability to malicious attacks (threats) has become worrisome. If something is not done to curb these threats, health care infrastructure will lose its relevance, patients' trust and patronage (Aijaz *et al.,* 2023).

The importance of security and privacy of patients in health care and the need to preserve life, reduce morbidity and mortality rates has motivated this researcher to design and implement a framework that will classify threats in critical lifesaving electronic healthcare system.

2.2 Theoretical Framework

Traditionally, in Africa, and Nigeria in particular, healthcare is paper-based, patients records are paper based, referral of patients from one hospital to another is paper based, even communication between units within a hospital is paper based. In the 21st century, where there are many means of automated systems like smartphones with smarter means of communication such as SMS, MMS, electronic mailing, and social media, there should be a change in medical health system. The current paper-based system is fraught with problems such as:

i. Scalability is not tenable
ii. The paper-based system is prone to error and damage by pests
iii. The files are degraded with time
iv. The system is highly unreliable in a critical setup like the medical healthcare system
v. Retrieval is time consuming
vi. There is no visible audit trial
vii. The file cabinets and shelves occupy too much space.

The researcher is motivated by these shortcomings to develop and implement a framework that will monitor and report critical security threats. The framework will be automated devoid of human intervention.

The dataset to be used in this implementation is national science laboratory – knowledge discovery in databases (NSL-KDD) dataset. This dataset will be split into two parts, training dataset (80%) and testing dataset (20%) and used to train and test the classifiers respectively. Machine learning technique such as ensemble learning (bagging) will be used to implement the framework, with cloud computing technology (CCT) as the platform.

Algorithms to train using the dataset include Logistic regression (LR), decision tree (DT), K-nearest neighbor (KNN), random forest (RF), and naïve Bayes (NB).

The trained models will be used to predict labels. Their predictions will be aggregated using soft vote classifier to produce a consensus model that will be used to predict labels using test dataset, which they did not train with. The predicted labels will be compared with expected labels using confusion matrix, being a supervised learning problem.

The trained models' performances will be evaluated using the following parameters, accuracy, precision, recall, f1-score and area under the curve (AUC). Python programming language and its external modules such as scikit learn, pandas and NumPy will be used to train, test and evaluate the models.

The significance of the research work is to prepare the minds of hospital management on electronic healthcare system, which is more useable, accessible, accommodating and versatile, given the modern-day communication networks, including social media.

2.3 Electronic Health System (EHS)

EHS is the use of Information and Communications Technology (ICT) to support health and health related fields. It encompasses a wide range of uses such as telemedicine, smartphone applications, sensors and wearable; from mobile health (m-Health) to telehealth, and increasingly underpins all health-related activities (Singhal and Cowie, 2020). Other e-health subsidiaries include Electronic Health Records (EHRs), Sensors, non-invasive sensors, wearable, invasive sensors, tele-health, self-care and personalized care.

EHR offers varying levels of functionality such as basic documentation, data entry, real-time display of clinical signs and observations, communication and interoperability with other healthcare professionals. Indeed, there is no unique EHR within any country; individual healthcare organizations purchase their software and use, even find it difficult to interchange records for fear of loss, theft and mutilation of records (Singhal and Cowie, 2020).

m-Health is the use of mobile wireless technology like smart phones for remote access to healthcare information services. Sensors are integrated to m-health systems. They measure signals and collect data that are transmitted or recorded for analysis. Sensors

are subdivided into evasive and non-evasive sensors. Wearable and non-wearable are examples of non-evasive sensors. Telehealth is the use of telecommunications and virtual technology to deliver health care outside of the traditional healthcare settings.

Nykanen (2017) defined e-health as a socio-technical system composed of healthcare organizations, service providers, professionals, customers, citizens and patients; two-sided market, industrial companies providing their products and services, technology-mediated platforms for communication, and infrastructures that in collaboration provide value and services. Infrastructure and services are needed for knowledge sharing, management, and exchange of information and data.

An e-health ecosystem in two-sided market means that products and services are developed and delivered to meet the customers' needs. Customers are health professionals, patients, citizens, service providers, healthcare organizations and suppliers.

However, Eysenbach (2001) defined e-health as an emerging field in the intersection of medical informatics, public health and business, referring to health services and information delivered or enhanced through the internet and related technologies. In a broader sense, the term e-health involves many things; technical development, state-of-mind, a way of thinking, an altitude, and a commitment for networked global thinking to improve healthcare locally, regionally and globally using ICT. Indeed, e-health encompasses more than internet and medicine, as such the "e" in e-health does not only stand for 'electronic" but implies a number of other "e's" such as depicted in Table 2.1 (Eysenbach, 2001:

Table 2.1: The Meaning of "e's" in e-Health System

S/N	"e's"	What the "e" stands for
1	**Efficiency**	e-health increases efficiency in healthcare by reducing cost, avoid duplication of diagnostics or therapeutic interventions between healthcare establishments, and patient's participation.
2	**Enhancing** quality of care	Increasing efficiency does not only reduce cost but enhances quality of service and healthcare.
3	**Evidence** based	e-health should provide scientific analysis in the laboratories to test for suspected ailments especially in diseases with confusable symptoms.
4	**Empowerment** of consumers and Patients	e-health makes the knowledge bases of medicine and personal electronic records accessible to consumers and patients over the internet hence providing patient centered medicine.
5	**Encouragement**	Patients and healthcare professionals' relationship is encouraged towards true partnership in decision making.
6	**Education**	Physicians, patients and other health workers should be educated in the use of ICT gadgets and medical terms used, diagnostics, which are tailored towards preventive and personalize healthcare.
7	**Enabling**	Information and record exchange between health care workers, government agencies and insurance companies in healthcare be enabled through safe ICT transmitting media.
8	**Extending**	The scope of health care is extended beyond the conventional boundaries, through global health providers online.
9	**Ethics**	The patient-physician interaction poses new challenges and threats to ethical issues such as online professional practice, informed consent, privacy and equity.
10	**Equity**	e-healthcare should be equitable between the "haves and have-nots". Political measures and legislation should ensure that the digital divide running between the rural vs. the urban populace, rich vs. the poor, young vs. the old is bridged.

Source: Eysenbach (2001)

In addition to the 10 “e’s”, e-health should also be easy to use, entertaining exciting and, indeed exist. In EHS, the patient medical data are stored remotely with cloud service providers (CSP) in servers and huge data bases as depicted in Figure 2.1

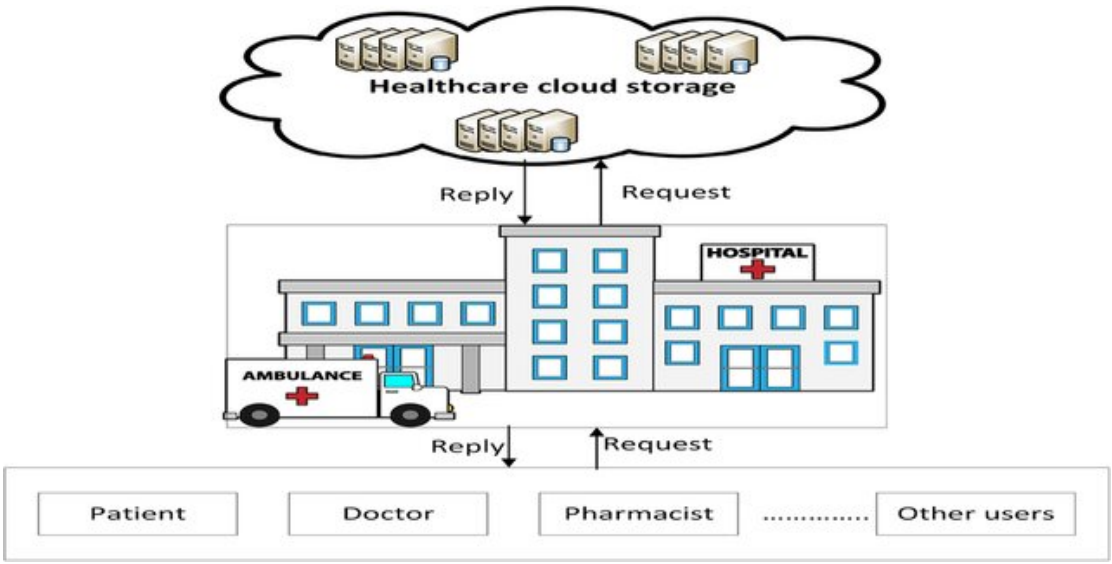


Figure 2.1: Electronic Health System with Patients’ Records Stored in the Cloud
Source: Singh and Chatterjee (2019)

Some CSPs include:

i. Amazon AWL

ii. Microsoft Azure

iii. Google Cloud

iv. IBM

v. NetSuite.

From Figure 1, the CPS collaborates with the hospital, which is responsible for managing the access rules, authentication mechanism and providing security and privacy to the patients’ data (Singh and Chatterjee, 2019). To treat a patient, the physician and other healthcare workers request for the patient’s record through the hospital using the healthcare application. The patient can also access his or her personal medical records. Delegate users can also obtain permission to access patient’s medical

records or issue permission to the healthcare practitioners to access their data. Other healthcare users such as researchers, insurers or government agencies can also access the patients' medical records through the hospital. In case of emergency, some access rules are relaxed to accelerate privilege treatment. However, EHS and its subsidiary HER are not fully adopted by patients, citizens and medical practitioners.

Health workers and patients have varying reasons for not wholly accepting EHS/HER. Some of the reasons include:

i. Lack of awareness of, and confidence in healthcare solutions among patients, citizens and healthcare professionals.
ii. Lack of interoperability between e-health solutions.
iii. Limited evidence on the cost effectiveness of e-health tools and services
iv. Lack of legal clarity for health, and wellbeing of mobile applications
v. Lack of transparency regarding the utilization of data collected by such applications.
vi. Fragmented legal frameworks, including the lack of reimbursement schemes for e-health systems.
vii. High start-up cost in setting up e-health systems or business
viii. Regional disparity in accessing ICT services, especially deprived areas

On the whole, education will go a long way to disabuse the minds of patients and e-healthcare practitioners to enable patients, young or old to access e-health interventions.

2.4 Electronic Medical Records (EMR) in Healthcare Systems

An Electronic Medical Record (EMR), also called Electronic Health Records (EHR), is one of the subsidiaries of EHS. EMR is an electronically stored digital version of medical data, where patients' examination results, prescribed medication, as well as

treatment history records are stored (Wesolowski *et al.,* 2016). EHR are real-time records, which are patient-centred and accessible to all authorized personnel including health care providers and organizations (Wesolowski *et al.,* 2016). By patient-centred healthcare we mean that the patient is given his or her dignity when treated, respected and involved in decision taken on his or her healthcare needs It empowered the patient to become active participant in his or her care.

For example, physicians, nurses, laboratory technicians, pharmacist and patients are actors manipulating the asset (EHR). Although EHR and EMR are seemingly used interchangeably, there are slide differences. EMR allows for electronic data entry, storage and maintenance of digital medical data, while EHR contains the patient's records from doctors (physicians), which include demographics, test results, medical history, History of Present Illness (HPI) and medications. Therefore, EMRs are part of EHRs and may contain the following (HHS, 2022).

i. Patient registration, billing, preventive screening or check-ups.
ii. Patient's appointments and schedules
iii. Tracking patient's data over time
iv. Monitoring and improving overall quality of care

However, EHRs are threatened by malware and the threats could be categorized into

i. Human factors: employees, contractors or hackers.
ii. Natural disasters, such as fire, earthquakes and flood
iii. Technical failures such as system crash, vulnerabilities, malware attacks, and hardware misconfigurations.

Human threats could be further categorized into internal or external; intentional or accidental. For instance, in 2015 up to 113 million breaches were recorded

(Wesolowski *et al.,* 2016). The ubiquity of computers and mobile devices has increased the risk of unauthorized access to EHR data. These risks are based on a range of factors, which include:

i. User related issues

ii. Financial issues

iii. Design flaws

These risk factors prevent EMR/EHR from being used as effective tools to deliver healthcare services. In recent times, it has become a common target to health care breaches. In 2020, at least 2,354 US government healthcare facilities and schools were breached by ransomware (HHS, 2022). This attack caused significant disruption across the healthcare industry. Also, in 2021, Health and Human Services (HHS) received reports of data breaches from 578 healthcare organizations, impacting more than 41.45 million individuals. Figure 2.2 depicts top threats against EMR/EHR systems.

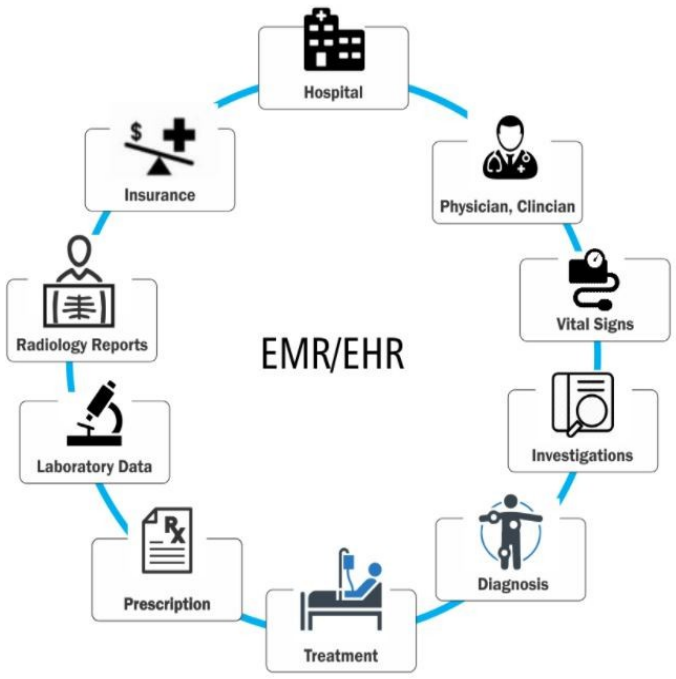


Figure 2.2: Top Threats Against Electronic Medical and Health Records
Source: HHS 2022

The threats impacting Figure 2 include:

i. Phishing attack

ii. Malware and Ransomware attacks

iii. Encription Blind spots

iv. Cloud threats

v. Employees (staff, contractors, supliers)

A phishing attack is a social engineering scam, where threats actors pretend to be trusted sources and lure innocuous user to open the email or click a link. A social engineering attack depends on the user to propergate. Figure 2.3 depicts a phishing attack.

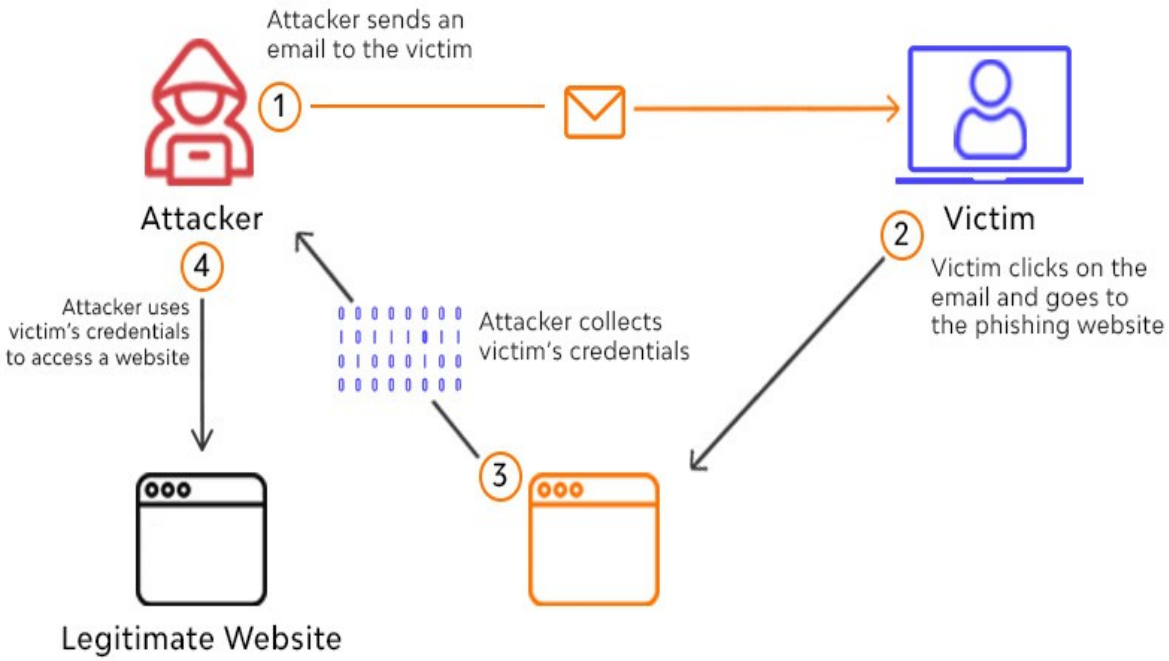


Figure 2.3: Phishing Attack on Electronic Health Records
Source: HHS (2022)

Ransomware is a type of Trojan malware (malicious software) that locks out the user from their host system or network and demands for a ransom, usually paid in bit coin. For hospitals, this could result in death as it disrupts patients' care. Figure 2.4 depicts stages of ransomware attack

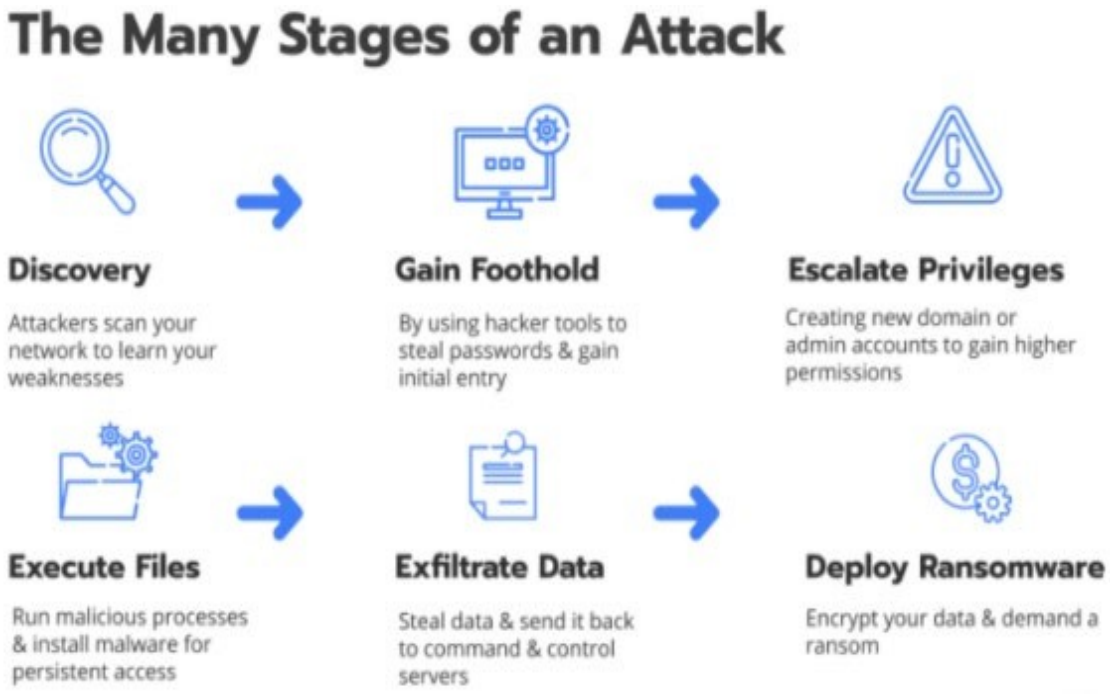


Figure 2.4: Stages of Ransomware Attack on EHR
Source: HHS (2022)

Encrypted blind spots are exploited by hackers to hide and avoid detection until they execute their payload on the target object.

The breaches in healthcare systems include:

i. Security and privacy issues
ii. Vulnerability
iii. Data can be lost or stolen or destroyed
iv. Inaccurate paper to computer transcription
v. Cause of treatment error

These breaches are propagated by hackers because EHR/EMR data have economic value in the black market (HHS, 2022). The data components or features include:

i. Name
ii. Geographic data
iii. Email address
iv. Health plan beneficiary numbers
v. Web URL
vi. Full face photo and comparable images
vii. Date
viii. Telephone numbers
ix. Fax number
x. Medical Record number
xi. Device identity and serial number
xii. Biometric identity such as retina scan or fingerprint
xiii. Social security number
xiv. Account number
xv. Internet Protocol (IP) address

From the foregoing, it is important to educate healthcare professionals to verify all EHR requests before sharing files with any remote request. The fight against malware threats is carried out in the cloud, which provide several services in many platforms. Given its advantages, many industry owners are now hoisting their data and services in the cloud. Figure 2.5 depicts the services of cloud computing technology.

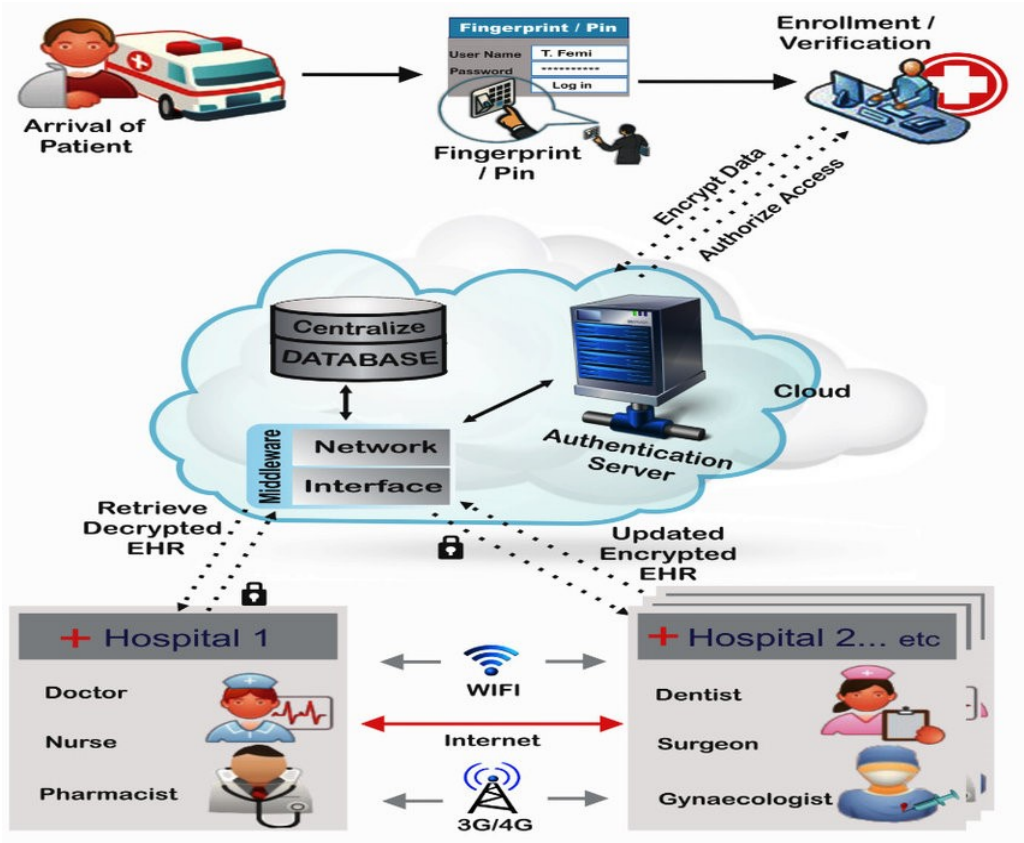


Figure 2.5: Cloud Computing Services in Medical Health Care System
Source: HHS (2022)

From Figure 2.5, the patient arrives in the hospital and undergoes the process of registration, identification, verification if he or she is regular patient, and activities of the various healthcare personnel as they manipulate the patient data in the cloud.

However, to reduce internal threats in healthcare organizations, the cyber security strategy and policies must be adhered to, such as:

i. Educate all healthcare patients and staff
ii. Enhance administrative control
iii. Monitor physical equipment and systems across the organization,
iv. Create workstations policies such as auditing and monitoring system users, employ device media control, and apply data encryption.

2.5 Security

Security is a process that protects systems, data, and transmission routes from unauthorized access (Pawar and Anuradha, 2015; Yesilyurt and Yalman, 2016; GSMA, 2019). Security has the following attributes: confidentiality, integrity, authenticity, non-repudiation, availability and access control.

i. Confidentiality prevents unauthorized access to data. (Babita and Kaur, 2017).

ii. Integrity prevents unauthorized modification or manipulation of data in transition.

iii. Authenticity verifies the identities of communicating parties in a transaction.

iv. Non-repudiation ensures that communicating parties do not disown their involvement or commitment to the transaction.

v. Availability ensures that required resources and services are available to authorised parties. It prevents denial of service or denial of access to legitimate users.

vi. Access control regulates who uses resources in a computing environment. It ensures that only legitimate communicating parties have access to data, through user name and password, digital signature, and finger printing.

There are many types of computer security such as physical security, network security, and cyber security. In this research work, physical security and network security are discussed. Figure 2.6 illustrates the security attributes as modified from Babita and Kaur (2017):

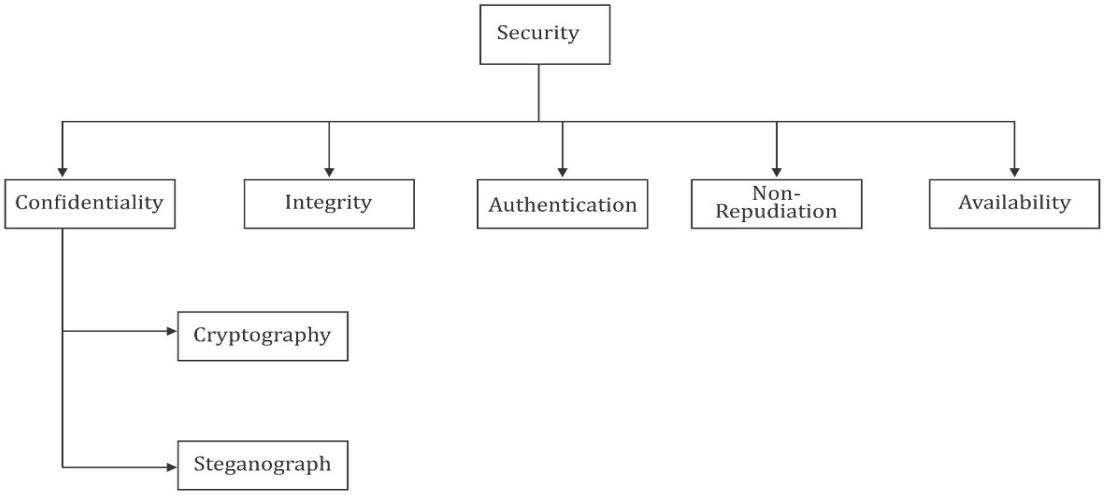


Figure 2.6: Security Attributes

2.5.1 Physical Security

Pphysical security of network environment is designed to protect networked computers, servers, routers, switches, hubs and gateways that house important data and system files. Unfortunately, this aspect of network security is often ignored by system administrators, and more often than not exposed to attack (Pawar and **Anuradha** 2015; Stein, 2020). Figure 2.7 depicts a vulnerable hub in a network system. For instance, Ethernet hub broadcasts data through every port; if a spare port is left unused, it becomes vulnerable as an intruder can plug his or her laptop in such a port and as the hub sends out data to legitimate systems, the same is sent to the intruder. Packet sniffing can also be plugged in such a port for the same purpose; an example is address resolution protocol (ARP) spoofing (faking the ownership of a particular address). Internet control message protocol (ICMP) uses routers to broadcast messages that contain their IP (internet protocol) addresses, which could be intercepted if left unattended to. These examples serve to demonstrate the fact that physical security measures should not be ignored by company management.

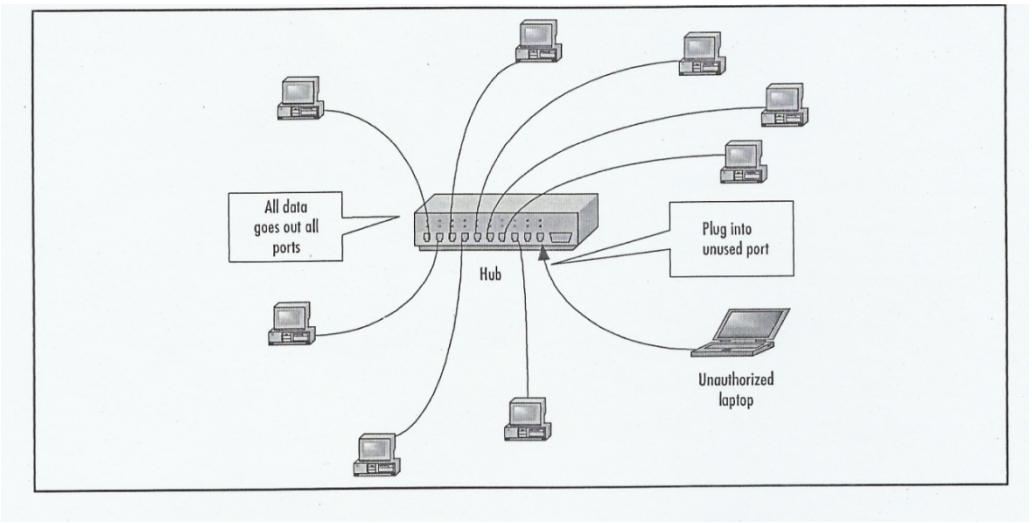


Figure 2.7: A Vulnerable Hub in a Network System
Source: Stein (2020)

Ethernet unshielded twisted pair (UTP) and coaxial cables are also vulnerable to data capture by tapping into them using appropriate devices (tools) if not attended to. Fibre optic cables that use laser light pulses to transmit binary data (1s and 0s) are not free from attack either. Optical splitters are used to tap into optical fibre cables for unauthorized access to data (Pawar and Anuradha 2015; Stein, 2020). Although physical security (wired medium) has its shortcomings, it complements other network security types.

2.5.2 Network Security

Network security consists of the six attributes: confidentiality, integrity, availability, authenticity, non-repudiation, and access control using firewall, encryption, steganography, intrusion detection system (IDS), intrusion response system (IRS) and more (Pawar and Anuradha, 2015). These security techniques are applied to systems or data features using network transmission media, which are either wired or wireless.

Three varieties of wireless media abound, such as radio (narrow band or spread spectrum), satellite (microwave), and laser (infrared). For instance, radio-based media operate according to IEEE 802.11i standards. However, wireless media have many

drawbacks in network security. They are susceptible to eavesdropping and man-in-the-middle (MITM) attacks that hijack (swindle) legitimate transactions from their owners. Laser signal may not be that susceptible, but it is a line-of-sight technology and sensitive to ambient factors like weather, flood, and earthquake (Pawar and Anuradha 2015; Stein, 2020).

For instance, in 2018, there were 53,308 security incidents, 2,216 data breaches, and 65 countries involved in these data breaches (Verizon, 2018). From these security breaches: 76% attacks were financially motivated, 73% of cyber-attacks were perpetrated by members of organized crime (outsiders), while 28% of such attacks were perpetrated by insiders (Verizon, 2018; Cheng *et al.,* 2017; Sharma *et al.,* 2020). In the public sector, 304 confirmed breaches were reported in 2018, they include cyber espionage, privilege misuse, web applications and lost or stolen assets (Verizon, 2018; Vemprala and Dietrich, 2019; GSMA, 2019).

Every organization, and individuals alike, needs a secured network as unstructured data is generated from there, stored there, and processed there. Unstructured data do not have a pre-defined data model nor organized in a pre-defined manner; examples include social media comments, images, emails, and rich media. Unstructured data is generated and transmitted using communication networks which include wired and wireless technologies. Wired technology is the communication media that involves Ethernet unshielded twisted pair (UTP) cables, coaxial cables and fibre optic cables. These cables are used to transmit data from source to destination (Ibrahim, Danladi, and Aderinola, 2017).

Wireless technology uses electromagnetic or acoustic waves such as radio (narrow band or spread spectrum), satellite (using micro waves), and laser (propagating infrared) to transmit data over short and long distances between sender and receiver (Ibrahim *et al.*,

2017; GSMA, 2019). The advantages of this type of transmission include reduction to cable distance, dynamic network formation, low cost and ease of deployment. The areas of application include Bluetooth, Wi-Fi (an IEEE 802.11 protocol), global positioning system (GPS), general packet radio service (GPRS), two-way radio, television remote control, mobile devices, wireless modem, computers, wireless keyboard and wireless mice (Yoti and Saini, 2017; Ibrahim *et al.*, 2017). Table 2.2 depicts the comparison between wired and wireless networks.

Table 2.2: Comparison of Wired and Wireless Networks
Source: Yoti and Saini (2017)

S/N	Characteristics	Wired networks	Wireless networks
1	Installation	Difficult to moderate: more components to connect using cables.	Easy installation
2	Visibility: Node to node	All nodes on a wired network can hear each other	Many nodes on a wireless network cannot hear each other on the same network
3	Visibility: Network to Network	Wired networks are invisible to other wired networks.	Wireless networks are often visible to other wireless networks. They do affect the performance of each other.
4	Cost	Low cost	High cost
5	User connectivity	Limited	Connectivity is wide
6	Mobility	Limited	Outstanding
7	Reliability	High	Reasonably high
8	Efficiency	High	Very high
9	Speed and bandwidth	High, up to 100mbps	Low, up to 54 mbps
10	Type	LAN, WAN, MAN	WLAN, WMAN, WWAN, WPAN; GSM, TDMA, CDMA; Wi-Fi (802.11) network, Bluetooth, Infrared network
11	Signal loss and fading	Less interference	More interference due to absorption, refraction and reflection
12	Interference	Less	High (radio, weather, other wireless devices and obstructions like walls
13	Quality of service	Better	Poor- delay and longer connection time.

2.5.3 Wireless Network Protocols and Their Vulnerabilities

There are many wireless protocols controlling networks. Protocols are sets of rules that govern the relationship between computers and other equipment within the network. They, however, are vulnerable and permeable to attacks. Some of them include:

i. **Access Point (AP**) protocol

This is a hardware device that enables the connection of wireless communication devices such as smart phones, tablets, PDAs, and laptops to a wireless network. Usually, an AP connects to a wired network and provides a bridge for data communication between wired and wireless devices.

ii. **Open System Authentication (OSA) protocol**

OSA is an IEEE 802.11 standard. It enables a handshake between sender and receiver. The client sends authentication request (in plain text) to the server, which responds in cipher text. However, authentication management frames are sent unprotected (in plain text). Upon a successful handshake, both stations (source and sink) are considered mutually authenticated. For a better communication security, OSA can be hybridized with wired equivalent privacy (WEP) protocol. WEP is used to encrypt the plain text, only after the "hand shake". This communication between sender and receiver can be tapped by hackers since it is done in plain text. However, steganography is the solution (or a hybridized form with cryptography).

iii. **Shared Key Authentication (SKA) protocol**

SKA uses WEP (wired equivalent privacy) and a shared secret key to provide authentication. Upon encryption, the authentication client will return the cipher text to the AP for verification. Authentication is a success if the AP decrypts the cipher text back to same plain text.

iv. **Service Set Identifier (SSID) protocol**

SSID allows wireless clients to communicate with appropriate AP. With proper configuration, only clients with correct SSID can communicate with the AP. SSID acts as a single shared password between the AP and the clients.

v. **Wired Equivalent Privacy (WEP) protocol**

WEP, an IEEE 802.11 standard, is used to provide confidentiality to data transmission over a wireless network, by encrypting the said transmitted data or information. Hackers have cracked WEP, and made it insecure, using automated tools, which are freely available on the net for download.

vi. **Wi-Fi Protected Access (WPA 1 and 2)**

WPA 1 is a wireless security protocol designed to fix known security flaws in WEP and improve authentication. However, WEP 2, which is based on IEEE 802.11i, is a revised protocol which allows only authorized users to access a wireless device (Ibrahim *et al.,* 2017; Yoti and Saini, 2017).

The vulnerabilities of these protocols are exploited by malware intrusions.

2.4.4 Cyber Security in Electronic Healthcare Systems

Cyber security refers to the measures that are taken to protect data and information stored on all electronic devices that are connected to the internet from malicious software (malware), hacking operations, data theft, sabotage, and transmission disruption (Saeed and Asaad, 2022). To curb the menace of malware threats involves traditionally static analysis and dynamic or anomaly detection methods.

Static analysis methods decompose or decompile the application to learn its characteristics and determine its payload physically, without running the said application. While these techniques are efficient and detect known malware using their family signatures stored in databases, they are however, easily avoided using

obfuscation techniques, polymorphism and metamorphism. They cannot detect new variants of the known malware and zero-day attacks. Zero-day attacks affect vulnerabilities that have not been made public before being exploited by hackers. Static analysis main drawback is that they do not access the source code of the programs they analyze, more so, it is difficult to extract binary code (Kumar and Das, 2023). Unfortunately, majority of security solutions depend on signature families, and are therefore static. To address this pitfall in static analysis, dynamic solutions are embraced (Saeed and Asaad, 2022).

Anomaly detection methods are aimed at identifying deviations from normal system's characteristics or user behavior. However, they are prone to false positive rates (FPR), can be difficult to tune and maintain (Katiyar *et al.,* 2024). Again, anomaly-based detection of malware threats operates in silos (confined environments like sandboxes), focusing on specific aspects of the network or host system, such as endpoints, servers, or applications without a holistic view of the whole security setup. This fragmentation approach leads to blind spots and oversight in detecting threats that span multiple domains (Katiyar *et al.,* 2024).

The limitations of these traditional techniques, coupled with increasing volume, velocity, and variety of cyber threats, necessitated the use of machine learning techniques, on aspect of computational intelligence (soft computing) to solve such problems. Machine learning (ML) approaches have the following abilities:

i. Improved threat detection by analyzing huge data sets (big data) to identify patterns and anomalies and unravel the Mistry of unknown and zero-day threats.
ii. It carries out automated discovery of threats without human intervention (system administrators).

iii. ML algorithms can continuously adapt and learn new threats and are scalable when new datasets are added to train, compared to rule-based approach

iv. ML models learn the characteristics of normal applications and use that knowledge to predict the type of an application presented to it, to enable mitigation.

To effectively use ML tools, the data input features need to be cleaned, normalized and reduced to acceptable principal components using principal component analysis (PCA).

The process of transforming the raw data into a usable format is called feature engineering. Feature engineering involves the extraction of statistical properties, header information or byte sequences form the network traffic data. These parameters are analyzed and the results gotten are used to compute the performance of the models. Common performance evaluation parameters are accuracy, precision, recall, f1-score and area under the curve (AUC), an aspect of receiver operating characteristic (ROC) curve.

To achieve these parameters, ML tools such as Logistic regression (LR), decision tree (DT) (decision tree can be C4.5 or J48 or ID 3), Gaussian naïve Bayes (GNB), K-nearest neighbor (KNN), random forest (RF) and deep-learning neural network (DNN), among others. They are usually trained to classify malware threats and then tested for generalization with new or dataset that it did not train with. In supervised learning problem, the predicted results are compared with the expected (known) values or labels using confusion matrix.

However, these activities are carried out in the cloud using cloud computing technology, with facilities provided by cloud service providers (CSP).

2.5.5 Cloud Computing Technology

The initiative of cloud computing technology (CCT) is based on the principle of reusability of ICT capabilities (Okediran *et al.,* 2022). More so, the emergence of CCT led to the reduction in cost of infrastructure, computation, application hosting, content storage and delivery using the "pay for what you use" policy, rather than owning infrastructure at such high cost. Cloud computing technology (CCT) is a distributed system (DS) aimed at providing unlimited shared resources to registered users. CCT provides physical implementations such as (Azeez and Der Vyver, 2018):

i. Software as a service (SaaS)
ii. Platform as a service (PaaS)
iii. Infrastructure as a service (IaaS)

Also, the implementation aspect of CCT include:

i. Private cloud computing
ii. Public cloud computing
iii. Enhanced cloud computing
iv. Community cloud computing

These implementations are carried out by the following cloud service providers (CSP:

i. Amazon AWS
ii. Microsoft Azure
iii. Google cloud

Cloud computing is very useful to healthcare organizations, for it assists them to focus on their therapeutic services. The benefits in using CCT include (Yeng, *et al.,* 2020):

i. Easy collaboration and data sharing
ii. Mobility

iii. Cost reduction on ICT services

iv. Cost reduction in infrastructure

v. Scalability

vi. Business continuity

vii. Flexibility and

viii. Strategic values

These advantages notwithstanding, cloud computing technology has security challenges such as

i. Data loss or leakage

ii. Transaction hijacking

iii. Insecure user interfaces

iv. Denial of service attack (DOS)

v. Malicious insiders

vi. Data breaches

vii. Abuse of cloud services by CSP

viii. Estimated metering and billing system

ix. Vendor lock-in

x. Virtual machine escape

In vendor lock-in, a client can move its data and services to a CSP but cannot easily change the vendor without severe cost, legal constraints or technical incompatibility. The cyber health community in Nigeria is in its infancy, with healthcare systems and services fragmented, disjointed and heterogeneous with strong local autonomy; it is also not distributed among healthcare givers platforms (Jenyo *et al.,* 2023).

Jenyo *et al* (2023) defined an ideal cyber health system as a condition of cyber systems and networks that are not only free from malware and botnet infections, but also contribute to the overall trust and usability of the cyberspace for the wellbeing of all. This researcher and the entire research community are fighting hard to curb the menace of malware threats and botnet attacks using machine learning tools, to ensure that the cyberspace remains safe.

2.6 Cyber Threats

Cyber security threats and cyber security attacks seem to be used interchangeably. However, there are minor differences between them. A threat is a possible security violation that might exploit the vulnerability of an asset. Threats can result from accidental events, environmental, human negligence or human failure. There are four types of threats (Saeed *et al.,* 2022):

i. Unstructured threats, usually executed by inexperienced people
ii. Structured threats, which involves an organized attempt to breach a specific network or organization.
iii. External threats – These might come from individuals working outside the organization, via the internet.
iv. Internal threats – This occurs due to authorized network access. It could be due to infiltrated server account or physical access

An attack on the other hand, is a deliberate unauthorized action on a system or asset. Attacks can be active or passive, it must have a motive and should follow a preplanned method, when the need arises. Primary attack types are:

i. Reconnaissance (Probe)
ii. Denial of service (DOS)
iii. User to root (U2R)

iv. Remote to local (R2L)

While threats can be intentional or unintentional; malicious or not malicious; it can cause damage or otherwise; it can also be initiated by the system or outsiders; Attacks are deliberate, intentional, and malicious, with an objective to cause damage. It can alter or damage information, it cannot be blocked by controlled mechanism, and it is always initiated by an outsider (Saeed *et al.,* 2022).

However, in this research work, the terms threat and attacks are used interchangeably. It is an act performed by individual or computer program with harmful intent, and with a goal to steal date, cause damage or disrupt operations.

Cyber security threats can imamate from various sources such as:

i. Nation-state – These are hostile countries that can launch cyber-attack on local companies and institutions to cause disorder, interfere with their communication or inflict damage on their equipment.
ii. Terrorist organizations with ideological intent
iii. Criminal groups – to extort money from the user through phishing spamming, spyware and malware.
iv. Hackers – motivated by personal gain, revenge, financial gain and political motivation.
v. Malicious insiders – Employees, contractors and suppliers.

Electronic healthcare system (EHS) was introduced to facilitate health care delivery and health record management. It facilitates improved workflow for health care providers and increased access to patients' records, for better service delivery at lower operational cost (Alhassan *et al.,* 2016). However, EHS has become one of the attack targets by adversaries due to security vulnerabilities. These threats (attacks) reduce the privacy of patients' records and erode the confidence EHS has enjoyed.

A threat is any action or event that may lead to malfunction of the host system or exposes the patients' records to threat actors (Alhassan *et al.,* 2016), thus infringing on security attributes like confidentiality, integrity, availability, authentication, access control and non-repudiation (Tatam *et al.,* 2021; Alder, 2024). A threat could also be defined as the combined probabilities of unwanted events and their impact on the target object (Sardi *et al.,* 2020).

Threat is again defined as different kinds of likely actions that are caused by either stealth or natural means against a functional system (Idris *et al.,* ND; Tatam, *et al.,* 2021). A cyber threat actor is a person or entity that undermines security attributes (Idris *et al., ND).* To identify threats in a host system or network requires the visibility of suspicious events identifiable via indicators. An indicator of compromise (IOC) is an action perpetrated by the adversary. Advanced persistent threats (APT) are attack groups that remain undetected after compromising a host system or network (Tatam *et al.,* 2021).

Given the resistant to detection of attack groups, research community has advocated the combination of classifiers (or models) using machine learning (ML). This is achieved using either hybridization or ensemble learning methods. To model threats in EHS, one needs to identify assets of the EHS, access points, threats and rates the identified threats. Threat modeling identifies and provides visibility to threats affecting an organization (Tatam *et al.,* 2021).

An asset is any valuable component of a system that attracts the attacker, while an attacker is a person or process or program that constitutes a threat to the asset (target object). The attacker can be within the system (inside threat) or outside threat. For EHS, assets constitute the various hardware and software components and **actors** that interact with the assets. Figure 2.8 depicts actors' interaction with an asset.

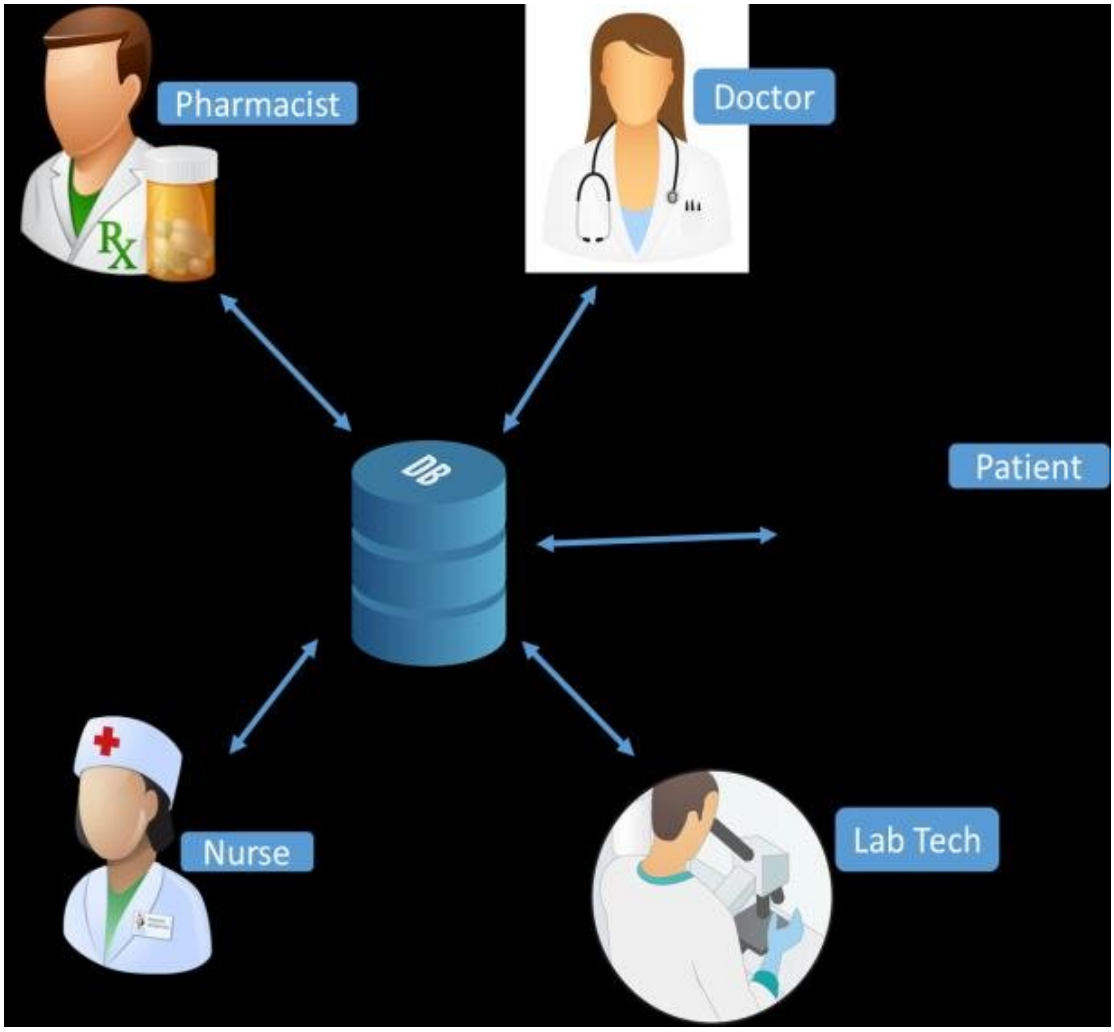


Figure 2.8: Threat actors interacting with the database (An asset)
Source: Alhassan et al. (2021)

The actors in Figure 8 include

i. The patient

ii. The Nurse who creates the patient's profile and generate identity code

iii. The physician, who records his diagnoses of the patient in HER

iv. The Laboratory scientist or technologist, who accesses the data in the EHR and conducts tests, and analysis of the samples provided by the patient and records in the database.

v. The pharmacist, who fills the prescription and provides the recommended drugs.

An access Point – these are the various interfaces used by attackers to obtain unauthorized access to the asset. Access points include:

i. Hardware ports
ii. Login screens
iii. User interfaces
iv. Open sockets
v. Configuration files.

Similarly, threats result from activities of inside attackers (who are privileged to be authenticated and authorized to access the system), and external attackers. Threats are borne from weakness in the design, coding bugs, and configuration of the hardware components. Some threats include:

i. Spoofing –This is a situation where by a person or program masquerades as legitimate through false identity to gain unauthorized access
ii. Tempering –Tempering involves changing the data to escape detection during an attack.
iii. Repudiation – This is the situation where by one of the business partners denies involvement or knowledge in the business
iv. Confidentiality – This is the situation where by information is unlawfully disclosed to unauthorized users
v. Denial of service (DOS) – this attack type bombards the target object with irrelevant requests to wear it down, and deny access to legitimate users
vi. Social Engineering –This is the act of luring the user to disclose confidential information. Some of such practices include shoulder surfing, pretexting, phishing,

man-in-the-middle (MITM) attack and code injection. Figure 2.9 depicts threat actions on the actors who access the EHR in their various duties.

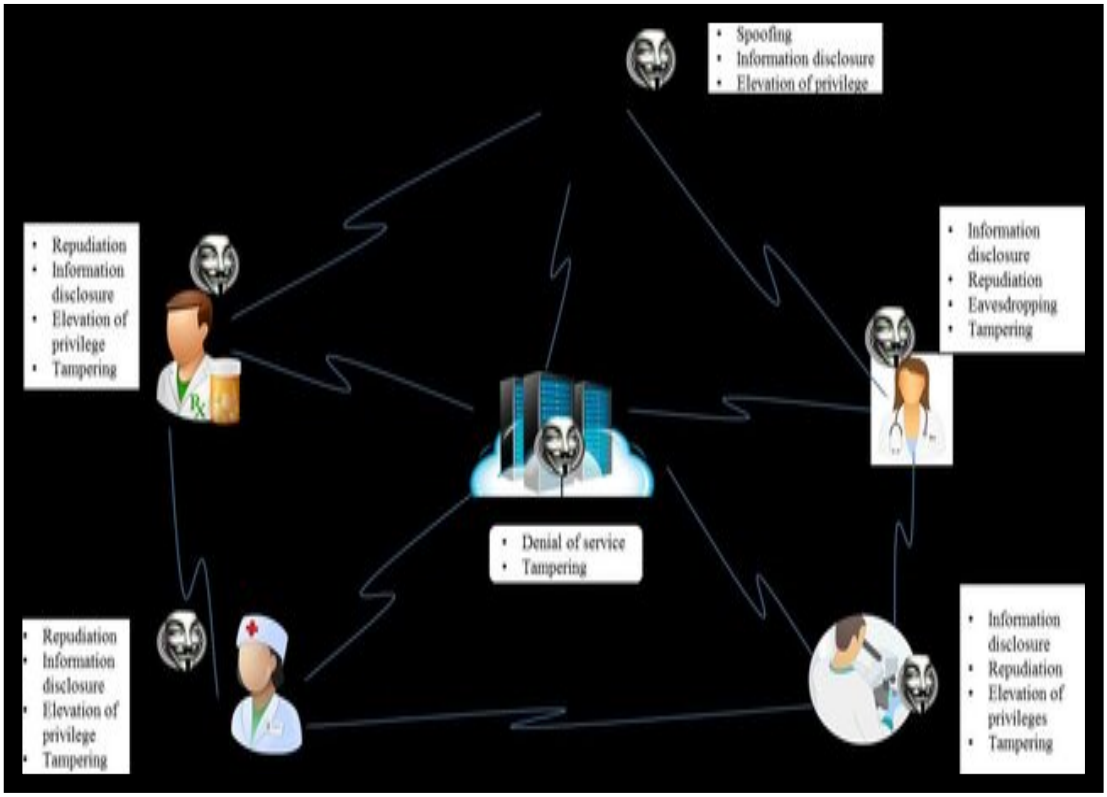


Figure 2.9: Threat Actions on the Various Actors and Asset
Source: Alhassan *et al.* (2021)

Figure 2.9 shows the database as the central asset that all users interact with by using the various interfaces available to them. Threats are ranked in terms of the damage potential, reproducibility, exploitability and discoverability. DOS is one of the attack methods, it can deprive user who connects to the EHS via the browser from accessing it. And to every actor, the related features of the threat are shown.

The most significant cyber threats include:

i. Malware, which are malicious software designed to infiltrate, damage, or gain unauthorized access to computer systems. They include viruses, Trojan horses, worms, ransomware, rootkit, key loggers, and so on.

ii. Phishing – This is a form of social engineering attack in which users are tricked to reveal sensitive information, or they install other malware by masquerading as legitimate entities.

iii. Advanced Persistent threats (APT) – These are stealth and continuous cyber-attacks, often sponsored by nation-state to target specific organizations and steal sensitive data, intellectual property, and disrupt operations.

iv. Insider threat – These are security threats originating from within the organization, either from malicious (aggrieved) insiders or negligent employees who expose systems or critical data to external threats, or contractors who are privileged to work in the system as suppliers or insurers.

v. Distributed Denial of Service (DDOS) attack – They attempt to obstruct the flow of traffic to or from a target server or asset by overwhelming it with unintelligent internet traffic from multiple sources.

Table 2.3 summarizes some of the cyber threats in recent times.

Table 2.3: Cyber incidents in recent years

Source: Katiyar *et al.* (2024)

Year	Incident	Impact
2017	WannaCry ransomware	Infected over 2000,000 across 150 countries
2018	Marriott Data breach	Exposed personal data of 500 million guests
2019	CapitalOne data breach	Compromised data of over 100 million customers and applications
2020	SolarWinds supply chain attack	Affected 18,000 customers including government agencies
2021	Microsoft Exchange Server Vulnerability	Impacted 30,000 US organizations and 250,000 organizations globally.

2.6.1 Malware Penetration Techniques

Malware penetrates computer systems including PCs, laptops and mobile devices through various ways, some of which are as follows (Arshad *et al.,* 2016; Abraham, 2017).

i. **Repackaging** – Malware developers decompile popular applications in Google play store, infuse or embed malicious contents and recompile, then distribute the impregnated (fake) application packages through third party stores. Repackaging is done through reverse engineering. Reverse engineering is a form of recovering or retrieving the source code from machine code (0s and 1s) in order to analyze the program and see what it does, subject to modification as in the case of repackaging. After embedding the code, the signature of the repackaged application is changed to enable it bypass static analysis. According to Arshad *et al.,* (2016), 77% of the top 50 free applications available in Google play store are repackaged.

ii. **Drive by download** – This is an unintentional download, which occurs when a user visits a malicious site unknowingly, and malicious code is injected into the victim's device. It could be an authorized download, without the user understanding the consequences of granting the requested permission (Arshad *et al.*, 2016; Abraham, 2017).

iii. **Homogeneity** – This is a situation whereby all the systems are running on the same platform, and connected to the same network; thus, increasing the chances of a worm in one computer spreading to other systems on the network (Arshad *et al.*, 2016; Abraham, 2017).

iv. **Dynamic payload** – dynamic payload is an encrypted malware, embedded into the device as a resource file in the system's application programming kit (APK), at run time. After installation, the malware decrypts the

encrypted payload and executes the code. Some codes are downloaded dynamically without being noticed by static intrusion detection systems (Arshad *et al.*, 2016; Abraham, 2017).

v. **Stealth malware technique** – Malware developers take advantage of the vulnerability of networked devices (host systems, laptops and mobile devices), which include low battery power, limited memory, and low processor speed; and obfuscate the malicious code to bypass the anti-malware system. Different stealth techniques used to attack victims' devices include: key permutation, dynamic loading, native code execution, code encryption, and java reflection. Vulnerability is a security defect in software and hardware that can be exploited by malware. It could be a design flaw, programming error or neglect of physical network components in the company, etc., (Arshad *et al.,* 2016; Abraham, 2017).

vi. **Zero-day vulnerability** – This is an undisclosed flaw that hackers exploit. It is called zero-day because it was not publicly reported before becoming active.

These malware penetration techniques are summarized into four types of attacks.

2.6.2 Common Malware Types

Malware comes in various forms, powered by various agents, such as (Abraham, 2017):

i. **Computer virus** – A computer virus is a contagious code or program that attaches itself to other software programs; and requires human interaction to propagate. It hides within computer executable files and modifies them in such a way that when a victim's file is executed, the virus is also executed (Yin and Song, 2019; Abraham, 2017). Such executable files have the following file extensions: .EXE, .BAT, .JS, .VB and. SCR. Viruses are the

only malware that infect other files, making them hard to be removed because they attach themselves to legitimate files; as such, most anti-virus programs rather delete the infected files or quarantine them (Abraham, 2017).

ii. **Worms** - A worm is standalone infectious software, which targets operating system files. While viruses attach themselves to existing files, worms carry themselves in their own containers. Worms usually show up through e-mails and instant messages. They use computer network to spread and do not attempt to change the system they pass through. Examples include Melissa, Morris, Mydoom, Sasser, Blaster and Mylife (Abraham, 2017).

iii. **Trojan horse**– A Trojan horse is a malicious program that masquerades as a legitimate or useful program. It spreads in the guise of routine software that persuades its victim to install it on his or her computer system. That is, it must be executed by its victim. Trojan payload is usually a backdoor malware that gives other malware like key loggers, tootkit, and more, unauthorized access to the affected computer or mobile device. Key loggers capture account names, passwords, credit card numbers, and pass all to the command and control (C & C) server. Trojan horses also steal user IP addresses, passwords, and banking details. Indeed, some Trojan horses even masquerade as anti-virus software, when actually they introduce malware to the system. Trojan horses are even developed and used by Government agencies like FBI and NSA. Examples include Magic Lantern, Finfisher, WARRIORS PRIDE (Abraham, 2017).

iv. **Hybrid and exotic forms** – Today, Trojans and Worms combine (hybridize) to form malware; and occasionally combine with a virus to form

a complex malware. The malware appears to the end user as a Trojan, but once executed, it attacks other victims over the network as worms. Examples include rootkits or stealth programs. Rootkits attempt to modify the underlying operating system and take absolute control of the system; and hide from anti-malware programs. To remove such malware, you need to remove the controlling component from memory. Bots are combination of Trojans and worms that attempt to make compromised systems (zombies) part of the larger malicious network (botnet). Botmasters are servers where zombies, the infected systems, take instructions from and send information gathered to, respectively (Abraham, 2017).

v. **Ransomware** – These are malicious programs that encrypt user data and hold them hostage waiting for a ransom to be paid in cryptocurrency before unlocking the system. They are Trojans and spread through social engineering (Yin and Song, 2019). Some examples include Reveton, Cryptolocker, Cryptowall, Wanacry, and Fusob. Fusobs affect computers and network (Abraham, 2017).

vi. **Rootkits** – This is a collection of malicious software that attempts to undermine operating system of the device and take control of the device. They usually work at the background so that users would not notice their presence. Rootkits are difficult to remove, some of which are NTRootkit and Sonny BMG Copy protection rootkit (Abraham, 2017).

vii. **Key loggers** – This malware records all the information typed using the physical keyboard. They cannot use or manipulate the virtual keyboard. The gathered information is sent to (C & C) servers (Abraham, 2017).

viii. **Grayware** – These are unwanted applications and files that though not classified as harmful, but can worsen the performance of computers and lead to security risks. Examples include Adware and Spyware (Abraham, 2017).

ix. **Adware** – Adware is used for advertising. They usually show pop-up adverts in windows that cannot be forced to close. They are nuisances but not harmful (Abraham, 2017).

x. **Spyware** - Spyware constantly spy on their victims. Their main purpose is to keep track of internet activities, including that of organizations, without the knowledge of users (Abraham, 2017).

These malware types have ways of evading detection from anti-virus programs and intrusion detection systems (IDS).

2.6.3 Malware Evasion Techniques

Several techniques exist that malware use to evade detection. These techniques are so sophisticated that anti-malware developers are struggling to catch up with.).

i. **Encryption** – Malware developers encrypt their payload to evade detection through static analysis. Encrypted malware is made up of two parts: the decryption loop, and the main body of the code. The decryption loop is capable of encrypting and decrypting the code of the malware. The main body is encrypted with either XOR or Advanced Encryption Standard (AES) algorithm. Anti-malware solutions must decrypt the main code to get to the valid signature and detect the malicious piece of software.

ii. **Oligomorphism** - The purpose of this technique is to produce a different decryption code on every infection. This technique can be detected but requires more time as the decryptions are many and randomly chosen

iii. **Polymorphism** – In polymorphism, the malicious code (payload) is modified to produce a new variant, using dead code mechanism; while maintaining its original algorithm.

iv. **Metamorphism** – This technique uses mutation engine to change the body (malicious code) on every compilation. Mutation creates slightly different versions of the same object called alleles - it generates new alleles.

v. **Obfuscation** – Hiding information by manipulating strings defeats intrusion detection systems (IDS) that are based on signature recognition. Encrypting malware is the first stage in bypassing signature-based solutions as they do not have readable information to compare. Some of the coding techniques used to evade detection through obfuscation include:

   a. *Dead code insertion* – It simply adds ineffective instructions to a program to change its appearance, but its behavior remains intact.

   b. *Register reassignment* – It simply changes or switches registers from one version to another.

   c. *Subroutine reordering* – A subroutine is reordered in a random way giving multiple variants.

   d. *Code transposition*- It reorders the sequence of the original code using unconditional branching or based on independent instructions.

   e. *Code integration*- A malware joins its code with a valid program by decompiling the original one and rebuilding it with infected code.

vi. *GPU-Assisted Malware* – Graphics Processing Units (GPU) are compressors that display 2D and 3D graphics information on the screen in real time, including videos, visual computing, and displays. It provides real time visual interaction with computed objects via graphics images and videos. GPU assisted malware does not run on virtual machines, as it uses GPU libraries. Also, GPU assisted malware has the ability to access host memory directly, making it to share CPU and GPU.

Hence, malware developers use GPU to perform tasks that might be noticed on CPU, just to go undetected (Vasiliadis, Polychronakis, and Joannidis, 2014).

Whereas malware have many ways of evading detection, they however, have many ways of leaving behind trails of their presence on a device (like smart phone).

2.7 Nigeria

In the global setting, Nigeria is a West African country and occupies an area of 923,768 square kilometers (Attah, 2017). Nigeria has an estimated population of two million people, made up of a plurality of ethnic groups with many languages. However, her official language is English. She gained political independence from Britain in October 1, 1960.

Administratively, Nigeria is divided into 36 states and a federal capital territory, Abuja. The States are further divided into 774 local government areas (LGAs). This implies that Nigeria is divided into Federal, State and Local governments.

Figure 2.10 depicts map of Nigeria showing the 36 states and federal capital territory

.

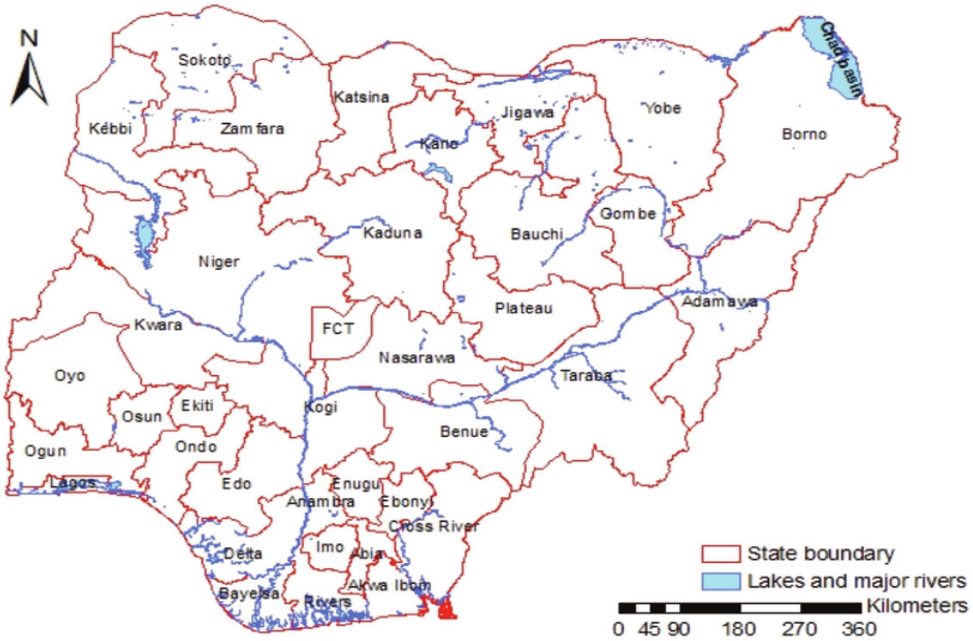


Figure 2.10: Map of Nigeria with Thirty-Six States and FCT
Source: Ishaku (2011)

2.7.1 The Nigeria Healthcare System

In Nigeria, healthcare is subdivided into three tiers

i. The traditional healthcare system

ii. Public healthcare system

iii. Private healthcare system

The public healthcare system is further divided into

i. Tertiary healthcare

ii. Secondary healthcare

iii. Primary healthcare systems

These orthodox healthcare systems are managed by the three tiers of government respectively. Figure 2.11 depicts the Nigerian healthcare system.

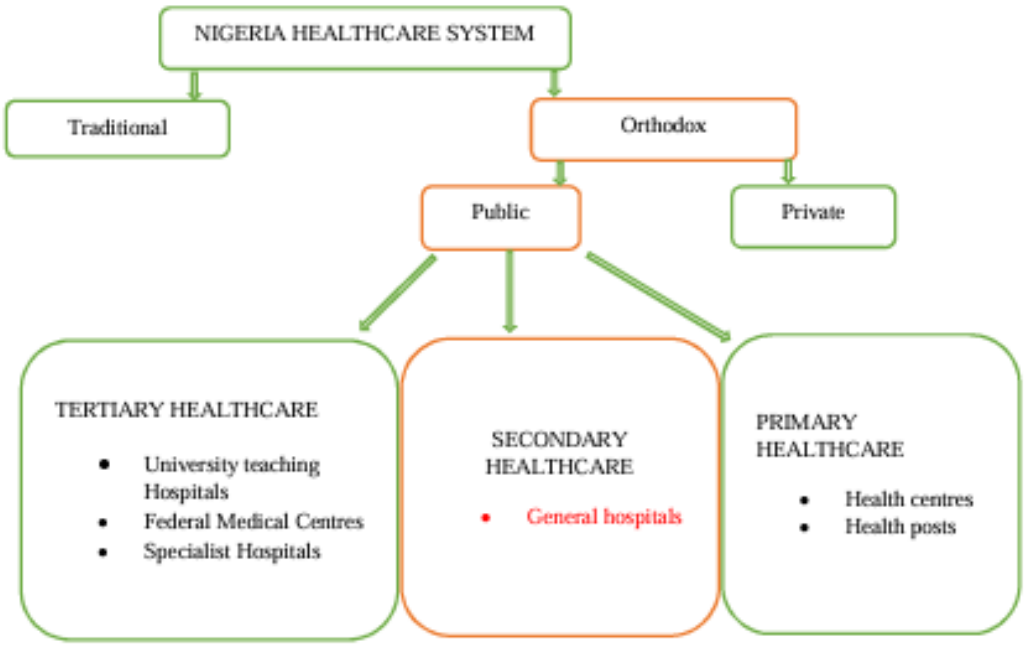


Figure 2.11: Healthcare Delivery System in Nigeria
Source: Attah (2017)

From Figure 2.11, the tertiary healthcare systems are managed by the federal government, and the secondary healthcare system is managed by the state governments; while the primary healthcare systems are managed by the local governments. While the organization of the healthcare systems seems well coordinated, the practical working of the setup does not as plausible. There are duplications of responsibilities and confused roles among the three tires of government, corruption in the system or absence of basic infrastructure. No drugs in the hospitals, no equipment, no beds, and in the primary level, there are no doctors nor nurses.

However, the private sector seems better organized than the government sector, with better facilities, personnel and infrastructure, but costly and near beyond the reach of the common man.

Traditional healthcare systems are seemingly not recognized by the government of Nigeria, rather, emphasis is on orthodox medicine. Nigerian patients, who seem to know what sickness is for orthodox medicine and which is for traditional care, patronize traditional medical care at their own discretion and risk. In developed countries, in particular Asian countries like China, traditional medicine is carried along with orthodox medicine and co-funded by government and private companies or individuals.

2.7.2 Electronic Health Systems in Nigeria

Efforts to develop e-Health in Nigeria started in 1994, but these efforts are piecemeal, uncoordinated and at experimental (pilot) stages (Attah, 2017). Barriers to e-Health implementation in Nigeria include:

i. Lack of national e-health strategy and policy
ii. Lack of e-health legislative framework
iii. Epileptic electricity supply
iv. Poor communication networks and facilities, with very loosely controlled mechanism by Nigerian Communications Commission (NCC).
v. Little or no ICT personnel and infrastructure in the healthcare systems
vi. Inadequate government commitment in terms of policy, legislation, funding and more.
vii. Malicious attacks on the pilot systems by malware and botnets, amidst internal and external aggressions

It is in the light of these problems that the researcher is motivated to join the research community to design and develop a framework that will use machine learning tools to detect the malicious software (malware threats) and suggest remedial option

2.8 Soft Computing

Cyber security is seriously being undermined by threats and this affects individuals, organizations, and society at large. Malicious actors (malware developers and hackers and even programs) and constantly developing variants of known malware signature families to compromise computer networks, host systems and intimidate innocuous users; the threats also steal sensitive or critical data and disrupt operations and services (Katiyer *et al.,* 2024). Cyber security approaches based on signature-family detection, and manually defined security policies could not keep pace with the ever-evolving threats landscape. Hence the use of machine learning tools to solve such practical problems.

Soft computing also called computational intelligence, is an aspect of artificial intelligence (AI) that trains computers or computer programs to perform the tasks of experts in specific fields. Machine learning (ML) is a subset of soft computing, which teaches computer to learn and improve on past experience without being programmed. With ML techniques, cyber security systems can analyze huge data to uncover hidden patterns, detect subtle anomalies and make intelligent decisions to prevent, detect and respond to cyber intrusions. ML algorithms have unparalleled capacity to analyze network traffic, user behavior and system logs in real-time.

Cyber security threats are analyzed and detected by three approaches:

i. Misuse or static approaches
ii. Anomaly or dynamic approaches
iii. Hybridized approach

Figure 2.12 depicts the three machine learning techniques that are used to detect malware intrusions.

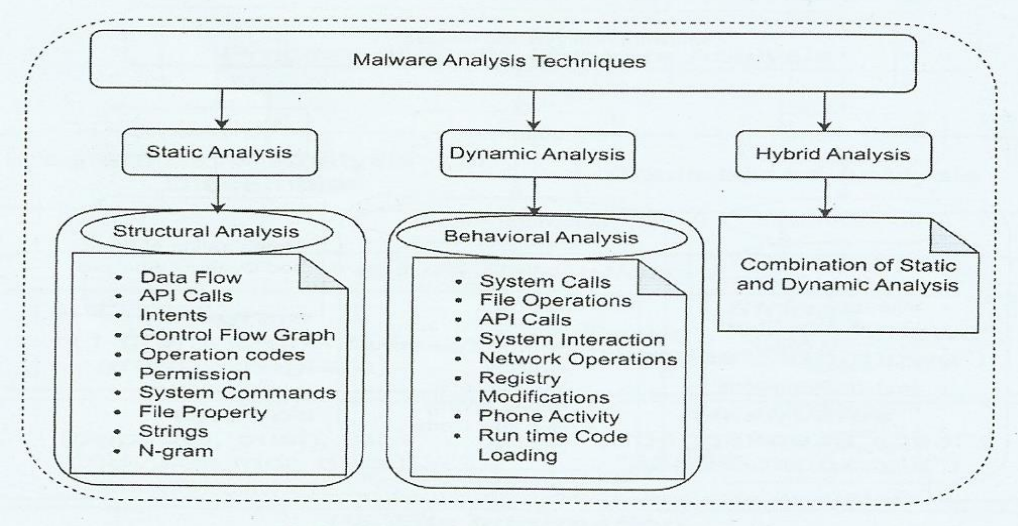


Figure 2.12: Taxonomy of Malware Analysis Techniques
Source: Smmarwar *et al.* (2024)

In misuse type, reliance is on signature-based detection, where known threat patterns are identified and blocked based on predefined rules and databases of signature families (Black lists). These techniques are very effective in detecting known threats and speedily too; they however, cannot detect unknown (new) malware and zero-day threats. More so, maintaining these families requires constant updating and can lead to false positive rates (FPR). Static analysis uses binary codes to identify harmful files and viruses. The biggest drawback of static analysis is the absence of program source code, and it is difficult to extract binary code (Kumar and Das, 2023).

Anomaly based detection aims at identifying deviations from normal applications or user behavior. However, these techniques are prone to high false positive rates and can be difficult to tune and maintain over time. More so, these anomaly detection types often operate in silos, focusing on specific aspects of the network or system such as endpoints, servers or applications, instead of focusing on the entire system or network. This fragmented approach can lead to blind spots and inefficiencies in detecting and responding to threats spanning multiple dimensions (Katiyar *et al.,* 2024).

Hybridized approach combines both static and anomaly approaches to produce a stronger and more effective intrusion detection system, to foil the sophistication of malware. Table 2.4 depicts ML malware detection techniques or methods, highlighting their features, advantages and disadvantages.

Table 2.4: Comparison of ML Based Malware Detection Approaches

IDS Techniques	Features	Advantages	Limitations
Static Analysis	File properties, byte sequences, and header information	Fast, low resource requirements, and comparatively low FPRs	Can be evaded using obfuscation techniques, and packing techniques.
Dynamic Analysis	API calls, Network traffic system or resource usage.	Captures run-time behavior, resilient to obfuscation attacks	Higher resources consumption, potential sandbox evasion.
Hybrid Analysis	Combination of both static and dynamic features	Improved accuracy, robustness to evasion techniques.	Increased complexity, may require manual feature engineering.

Source: Kaliyar *et al.* (2024)

2.8.1 Intrusion Model

Intrusion is an attempt by unauthorized persons or programs to undermine the security attributes of data: confidentiality, integrity and service availability (CIA) by exploiting their vulnerabilities (Bhuyan, Bhattacharyya, and Kalita, 2014). This attempt is perpetrated by malicious programs (malware) written and deployed by malware developers to achieve their nefarious intents. The diamond thread model is used to apply scientific principles to intrusion analysis. It maps the relationships and capabilities of adversaries to the target object. It consists of four main parts, as depicted in Figure 2.13, of an intrusion activity: the adversary using his or her capabilities to a target's infrastructure. It tracks attack groups because they usually change their targets, and their links over infrastructure against a target.

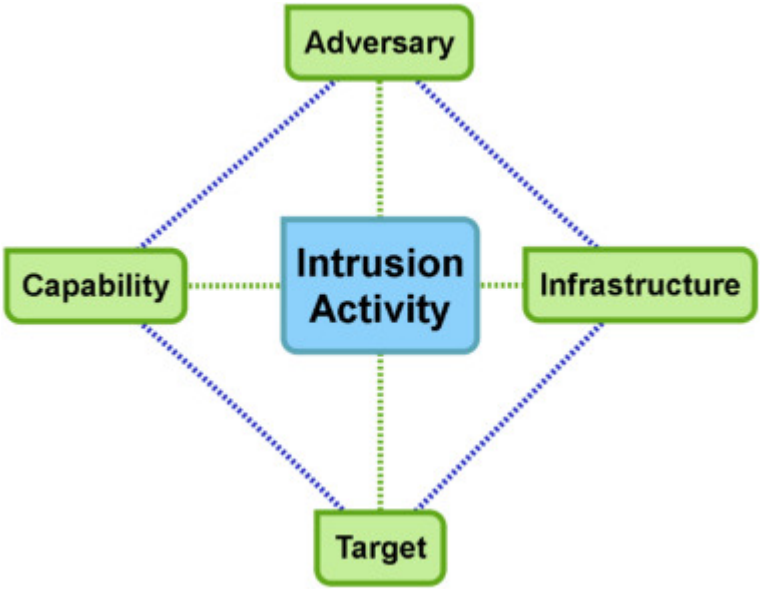


Figure 2.13: The Relationship Between the Attacker, His/Her Capabilities, and the Infrastructure of the Target Object
Source: Tatam *et al.* (2021)

The Adversary:

An intruder must have a reason for deciding to breach the security of a network. The reason, for modern malware developers, is to make money.

Capabilities:

An intruder must have the ability to carry out an intrusion, such as programming knowledge or access to attack gadgets, most of which are freely available on the net.

Infrastructure:

An intruder must have access to the network or the organization's infrastructure either through flaws in the security plan, bugs in software program, or physical proximity to network components.

The Target:

The target is the organization, the host system or network belonging to the organization, with the intent of disrupting the operation of the target.

Out of these four features, security experts or user or system Administrator can only influence or control the infrastructure; for capabilities dependent on the personality of the intruder or the type of data built into the network; there is no way anybody can stop a would-be hacker from acquiring programming knowledge or from obtaining the tools necessary to use in his or her nefarious intent (Ibrahim *et al.,* 2017). It is the responsibility of system Administrator to ensure that the network infrastructure is properly secured.

A good network security system should conceal all open ports, exploitable applications, and indeed, formulate good access control mechanism (ACM); the system should also be easy to use. These characteristics are achieved through the use of firewalls, cryptography, steganography, anti-virus applications, intrusion detection systems (IDS), and intrusion response systems (IRS) (Ibrahim *et al.,* 2017). Intrusion detection system (IDS) is one of the second-degree security mechanisms that detects and reports malware attacks.

2.8.2 Intrusion Detection Systems (IDS)

Intrusion is a set of activities used by adversaries to compromise systems or network components in terms of confidentiality, integrity and availability. It is a deliberate and unauthorized attempt to access information, manipulate data, and render a system unreliable or unusable (Bhuyan *et al*, 2014). These threats are achieved by either an insider of the system or outside agent, to gain unauthorized access and control of security mechanisms, if protective measures are not put in place.

To protect these security attributes (CIA), intrusion detection systems (IDS) are used. IDS gathers and analyzes information from various areas, within a host or a network, to identify possible security breaches, and report either to the user or system Administrator (Bhuyan *et al,* 2014). IDSs are broadly categorized into static or misuse-

based detection systems; and dynamic or anomaly-based detection systems. Misuse or static analysis techniques decompose or dismantle or de-compile applications off-line, and analyze their features and characteristics against signatures of malware families already gathered into a database. They are very efficient and fast in their analysis and classification between malicious and benign applications, with minimal false alarm (FPR) (Baby and Jeba, 2017; Azad and Jha, 2014; Dewa and Maglaras, 2016; Kruegel, N.D).

However, there are drawbacks to static analysis techniques: they are easily deceived with slight variations of the signatures they know; they cannot detect zero-day attack, nor can they detect new and unknown attacks. Signature variations are carried out using obfuscation techniques such as polymorphism and metamorphism (Baby and Jeba, 2017; Azad and Jha, 2014; Dewa and Maglaras, 2016; Kruegel, N.D). Polymorphic obfuscation encrypts the payload to evade detection, only to decrypt it back during execution. This is achieved using the following methods:

i. no operation (NOP),
ii. dead code insertion,
iii. code transposition (changing the order of instructions, and
iv. placing jump instructions, to maintain the original meaning (semantics),
v. register reassignment,
vi. behavioral insertion.

Whereas polymorphism attempts to encrypt only the payload, metamorphism encrypts the entire malicious application. These evasions are possible, partly because emphasis is placed on grammatical structure (syntax) of malware algorithm, ignoring the meaning (semantic) assigned to the symbols, characters, and words used in the grammatical

structure of the malware algorithm (Baby and Jeba, 2017). Grammar is a set of rules that defines whether or not the sentence is properly constructed.

However, security industries, as at now, are using static analysis method, which is pattern matching of targeted applications with family signatures found in the database. The consequences being regular update of the databases, inability to identify unknown malware and zero-day attacks, inability to detect dynamic code loading, native code reflection, java reflection and encrypted code. (Wang *et al,* 2015), Inability to analyze the source code (Tatam *et al.,* 2021). Figure 2.14 illustrates misuse-based IDS,

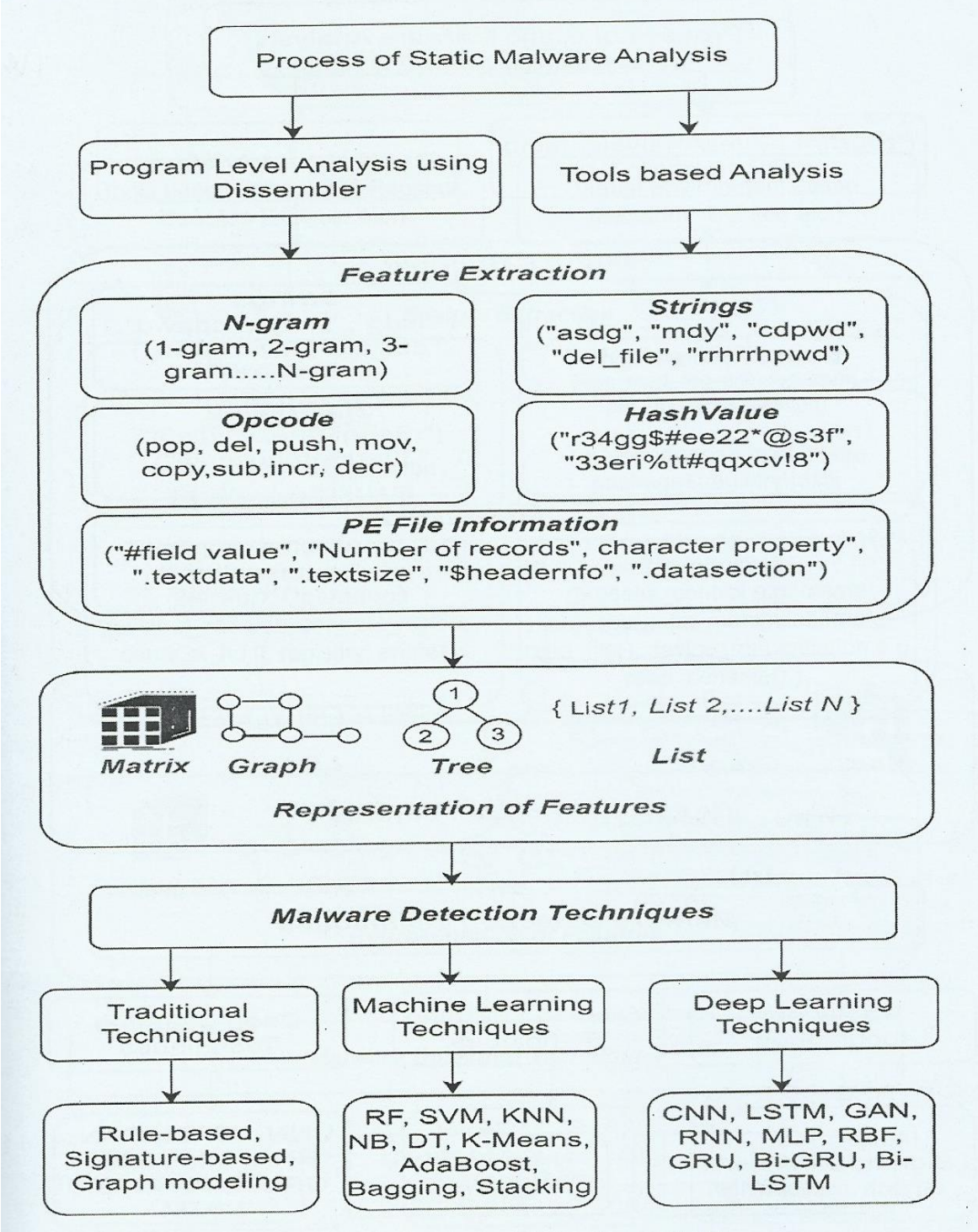


Figure 2.14: Schematic Diagram of Static Analysis for Malware Detection
Source: Smmarwar et al. (2024)

Anomaly based detection techniques are based on building and training classifiers (models) on normal behaviors, which is used as a basis to observe deviations by anomalous applications (Baby and Jeba, 2017). Dynamic IDS is trained to learn the

behavior and characteristics of normal applications, and report any deviation from the norm as malware. Some of the characteristics of normal applications include:

i. In Attribute Relation File Format (ARFF) file of NSL-KDD dataset, normal applications use TCP and UDP protocol types; while malware uses ICMP protocol type to execute DDOS attack on target objects such as ICMP flood, smurf attack with spoofed or faked IP address, and ping_of_death attack to cause buffer overflow and possible crashing of the device. Also, normal applications use SF flag, while malware have affinity for S0 and REJ flags.

ii. POLP – Principle of least privileges (Lord, 2020) states that "persons, applications, or processes be given the barest minimum resources to complete their tasks". Normal applications obey POLP principle by requesting for permissions and other resources within the acceptable range, but malware request for more than necessary resources for their nefarious activities.

iii. Android install time permissions – Permissions form the first barrier to malware attack. Normal applications declare permissions from manifest.xml file in keeping with POLP principle; but malware in addition to those in the manifest.xml file, ask for permissions to access hardware resources like sensors, camera, GPS, GPRS, and undocumented APIs from third party stores.

iv. Central processing unit (CPU) – Normal applications use CPU to process their data; but malware tends to use graphics processing unit (GPU) to process data and evade detection.

v. Dangerous permissions – These permissions tend to violate user privacy by requesting to access contact lists, photo albums, user ID/password,

credit/debit card numbers, etc. Malicious applications tend to use these permissions in their manifest.xml files, while normal applications request for normal permissions that do not temper with user privacy.

These techniques have two major advantages over misuse techniques: they are able to detect unknown attacks, as well as zero-day attacks; the aforementioned learned normal behavior is customized for every system, application or network. This makes it difficult for attackers to determine the attack pattern to use without being detected. The drawbacks of these techniques include: high resource consumption, high percentage of false alarms and difficulty in determining the event that triggers alarm; inability to detect privileged permissions, emulation detection system is prone to evasion by malware, and subject to low code coverage (Wang *et al.*, 2015). On the other hand, most researches in dynamic analysis use one subset of malware features to represent its behavior pattern and ignore other ones. For instance, API based sequences are used in analysis, ignoring network-based sequences which use Packet Capture (PCAP), an application programming interface for capturing network traffic, and files to extract the network flow information (Mashiri *et al.*, 2017). Also, it is observed that anti-virus vendors label malware samples differently, making them inconsistent with each other. Therefore, labeling malware in dynamic analysis may be less accurate, given the operators' approach in their heterogeneous methods (Mashiri *e al.,* 2017). These drawbacks have made the implementation of anomaly detection techniques slow and difficult to adopt. Figure 2.15 illustrates dynamic or anomaly intrusion detection system.

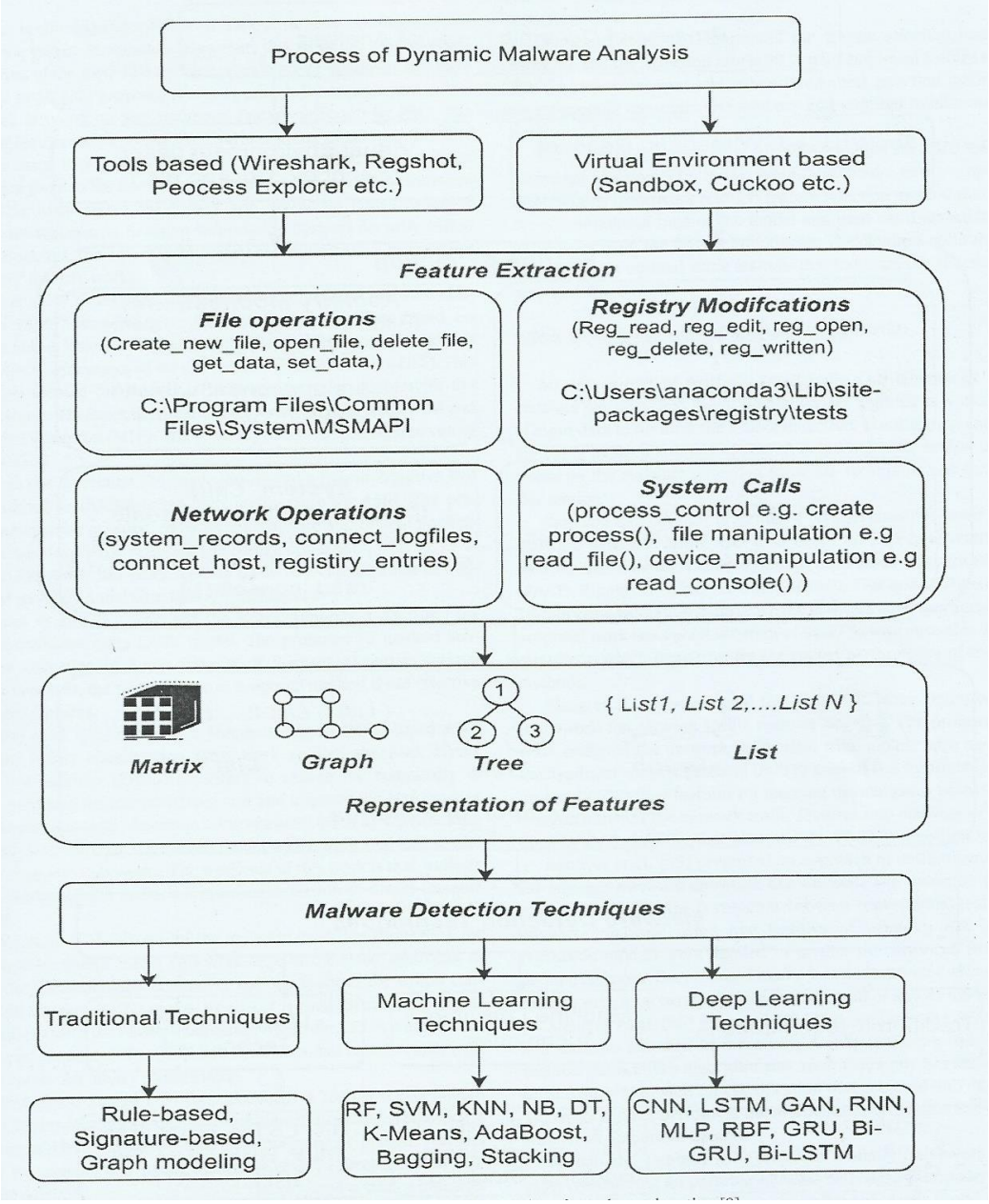


Figure 2.15: Schematic Diagram for Anomaly Detection of Malware
Source: Smmarwar *et al.* (2024)

From the foregoing, none of the aforementioned techniques is unique; they all have their strengths and weaknesses; as pointed out in "no free lunch" theory (Baby and Jeba, 2017, Bui *et al.,* 2017). The limitations of traditional techniques (misuse and anomaly types), coupled with increasing volumes, velocity and variety of threats has necessitated the use of machine learning techniques to solve the problems of incessant threats

(Kumar and Das, 2023). Using ML techniques to fight cyber threats has the following advantages:

i. Improved threat detection – ML algorithms can analyze massive datasets to identify patterns and anomalies that may indicate malicious activity, enabling the detection of previously unknown and zero-day threats.
ii. Faster incident response – ML models can automatically triage and prioritize security alerts, this reducing delay in response to attack initiated manually.
iii. Adaptive and scalable protection – ML models can continuously learn and adapt to new threats providing a more flexible and scalable approaches to cyber security compared to rule-based systems.
iv. Predictive analysis – By analyzing historical data and trends soft computing (SC) techniques can help predict potential future threats and vulnerabilities.

Some of these ML algorithms include Artificial Neural Network (ANN), Fuzzy logic, evolutionary computation (genetic algorithm and genetic programming), probabilistic computing including (KNN, SVM, NB, DT, RF, LR, DL (Deep learning)), and so on.

Artificial Neural Network (ANN) – This classifier predicts the behaviors of various users and daemons in the system. Its major advantage is it infer solutions from data without prior knowledge of the data structure. It is also tolerant to imprecise data and the uncertainty in it.

Fuzzy system – This algorithm or classifier models uncertainty in human expressions (experts) using Triangular Fuzzy Numbers (TFN) to convert the expressed variables or phrases into crisp values

Decision Trees – Decision trees are used to reduce variance in data sets. Its graphical representation consists of

i. Root – initial node

ii. Nodes – feature attributes

iii. Arcs – labels' categorical values

iv. Leaves – category of classes

Random Forest – This is a mega algorithm in ensemble bagging that combines the output of multiple decision trees, and other models to produce a more robust model or classifier. It does not require scaling or normalization. It can be used for classification or regression.

Naïve Bayes- This is a probabilistic classification based on Bayes' theorem, which assumes that the features are conditionally independent given the class label.

K-Nearest Neighbor – This is a non-parametric method that classifies new instances based on the majority class of the k nearest training instances in the feature space.

Support Vector Machine – This is a supervised learning classifier used for classification problems. Although it can be used in linear level analysis, it is most appropriate in higher level feature space. Basically, SVM is used for binary classification, it can also be used for multi-class problems.

Logistic Regression – Logistic regression is a classification algorithm. It is a data analysis technique that uses statistical models to find the relationship between two data factors. It then uses this relationship to find or predict the value of one factor based on the others. The prediction usually has a finite value say Yes or no.

The need to combine machine learning (ML) techniques based on their complementing features is now (Wang and Wang, 2015). The combination will be by using ensemble learning methods or hybridization method. In ensemble learning methods, weak or base learners are trained either sequentially or in parallel. The predictions of the trained models are aggregated using a voting technique, to produce a much better and stronger

model. Hybridization involves the use of binary models, which are trained, predict their results, and combine them to produce a better result than the individual algorithms. In this research work, ensemble technique used is bagging, with Random Forest as the training algorithm; while KNN, SVM, DT, and NB are base learners.

The performance of these models depends on the quality of input data used to train them. The process of transforming the raw data into informative features is called feature engineering. Feature engineering includes the following processes

i. Extraction of statistical properties
ii. Packet header information
iii. Byte sequences from network traffic data
iv. System logs metrics.

Feature selection is used to identify relevant and discriminative features from a larger set of data. It is aimed at improving model performance, reduce over fitting and enhance interpretability. Feature selection techniques include:

i. Filter methods
ii. Wrapper methods
iii. Embedded methods.

2.8.3 Classes of Machine Learning

Machine learning techniques are grouped into three main types:

i. Supervised learning methods
ii. Unsupervised learning methods
iii. Reinforcement learning methods

Supervised learning method – In this method, the algorithm learns from labeled training data set, where the desired output is known in advance. The goal is to learn a function

that maps input features to output labels enabling the prediction of labels to new, unseen data. That is, data that it did not train with (test dataset).

Unsupervised earning methods – In this technique, algorithms learn from unlabeled data, aiming to discover hidden patterns without any predefined output.

Re-enforcement learning methods – In this type, algorithms interact with its environment and receives reward or punishment for its action. The goal is to learn a policy that maximizes the cumulative rewords over time (Katiyar *et al.,* 2024). Table 2.5 depicts the main characteristics of ML types and applications.

Table 2.5: Characteristics and Applications of ML Types

ML Types	Characteristics	Application in cyber security setting
Supervised Learning	Learns from labeled data to predict output	Malware classification and spam detection
Unsupervised learning	Discovers patterns in unlabeled data	Anomaly detection, clustering, and so on.
Re-enforcement learning	Learns through interaction with its environment.	Adaptive Network security policies, agent-based systems

Source: Katiyar *et al.* (2024)

Similarly, Table 2.6 compares the ML types and notes their strengths and weaknesses

Table 2.6: ML Types and their Strengths and Weaknesses

Techniques	**Strengths**	**Weaknesses**
Re-enforcement learning Justine (2018)	Solves complex problems It may correct errors occurring during training It involves training data obtained through interaction with the environment It is flexible and can combine with other techniques	i. It is not good at solving simplified problems ii. It requires excessive data and computation iii. It is dependent on the reward function's quality iv. It is difficult to debug and interpret.
Unsupervised learning Aukur (2018)	i. Does not require manual data preparation ii. Capable of finding previously unknown patterns in data	v. Producing inconsistent results vi. Requires post-processing or interpretation to assign labels.

Supervised learning Mohamed (2017)	iii. iv. v. vi.	It can be normalized to avoid over fitting Linear models can be updated easily with new data It produces far more accurate results, and it is more reliable Efficient in finding solutions to linear and non-linear problems.	vii. viii. ix. x.	Takes a long time to train Limited performance especially in non-linear relationships It is not cost efficient with scalable data The accuracy is dicey. A higher accuracy does not imply higher performance.

Source: Katiya (2024)

2.9 Soft Computing Algorithms

Soft computing algorithms are algorithms trained and used to solve difficult and complex problems. They include:

2.9.1 K-Nearest Neighbor (K-NN) (Fix and Hodges, 1951)

K-Nearest Neighbor (K-NN) algorithm is one of the simple, efficient and interpretable algorithms that can be applied in classification as well as regression problems. Although simple, its performance competes favorably with complex classifiers like Support Vector Machines (SVM) and Artificial Neural Networks (ANN). In fact, its performance is used as a benchmark for complex classifiers (Prasath *et al.*, 2017).

K-NN was proposed by Fix and Hodges (1951), and ranks among the top ten (10) algorithms in Machine Learning (ML). It is a non-parametric and lazy learning algorithm. Non-parametric means that there are no parameters or a fixed number of parameters, irrespective of the size of the dataset. Parameters are rather determined by the size of the training dataset; K-NN is lazy learning because it stores the entire training data in memory and waits until the test data is introduced for analysis in real-time, without having to create a learning model (Prasath *et al.,* 2017). It is broadly divided into two sub types:

i. Structure less nearest neighbor (NN)

ii. Structure based nearest neighbor (NN)

In structure less K-NN technique, the whole data is classified into training and test data samples. Distance is measured from the training points to the test sample point, and the point with the shortest distance is called the nearest neighbor (Prasath *et al.*, 2017; Salvador-Meneses *et al.*, 2019). Structureless K-NN is used in combination with categorical (nominal or ordinal) data type, which is compressible in order to reduce the memory overhead; and decompresses in real-time during classification (Salvador-Meneses *et al.*, 2019). More so, traditional algorithms have problem working with datasets having very large attributes (or features or dimensions) because such datasets consume memory space (since the entire training dataset needs to be stored in memory prior to processing). For example, NSL-KDD dataset alone has forty-two (42) features of numerical and categorical types with two (binary) classes – benign and malicious classes. In machine learning, there are several types of data, such as numerical data, categorical data, text data, images, audio, video, etc. Current techniques of Machine Learning (ML) usually convert other forms of data into numerical data types before processing.

Structure based Nearest Neighbor (NN) techniques are of type tree such as ball tree, k-d (k-dimensional) tree, Orthogonal Structure Tree (OST), fixed axis tree, nearest feature line and central line. They all rely on continuous attributes. The difference between discrete and continuous data is that discrete data is countable while continuous data is measurable. Discrete data contains distinct or separate values. On the other hand, continuous data includes any value within a range, say [0, 1].

K-NN is applied in solving many practical problems like pattern recognition, text categorization, ranking models, object recognition and event recognition (Prasath *et al.*,

2017). Its simplicity, non-parametric, lazy learning (real-time analysis) and similarity of nearest neighbors categorized it among the top ten algorithms in ML (Salvador-Meneses *et al.*, 2019).

2.9.1.1 Principle of Similarity in K-NN

K-nearest neighbor algorithm is based on the principle that "similar things exist closer to one another or like things are near to each other" (Cheng *et al.*, 2014; Salvador-Meneses *et al.*, 2019). This principle enables the use of distance measuring techniques (metrics) for K-NN in classifying test sample points using the entire training dataset in memory.

2.9.1.2 Similarity and Metrics

The focal point of K-NN is its dependence on the distance measure (or similarity measure) between the test data point and the training data points stored in memory (Prasath *et al.*, 2017). There are up to fifty-four (54) distance measuring techniques in literature, grouped into eight families: L_p Minkowski distance measures, L_1 Distance measures, Inner product distance measures, Squared Chord distance measures, Squared L_2 distance measures, Shannon entropy distance measures, Vicissitude distance measures, and other distance measures (Prasath *et al.*, 2017).

Some of the distance measures, as given by Prasath *et al.*, (2017) include:

Minkowski distance measure, Manhattan distance measure, Euclidean distance measure, Chebyshev distance measure, Mahalanobis distance measure, Cosine distance measure, Hamming distance measure, Correlation distance measure, Soergel distance measure, and Lagrange distance measure. These distance measures must satisfy certain properties before they can be regarded as metrics or distance measuring functions.

2.9.1.3 Properties of Metrics

Distance is a numerical description of how far apart entities are. The distance function between two vectors **x** and **y,** denoted as d(**x, y**), is the distance between both vectors, which should be non-negative real number. This distance function is considered a metric if it satisfies the following properties

i. **Non-negativity**: The distance between **x** and **y** is equal to or greater than zero, as illustrated in Equation 2.1

$$d(x, y) \geq 0 \qquad \text{Equation 2.1}$$

ii. **Identity of discernible**: The distance between **x** and **y** is equal to zero if and only if (iff) **x** is equal to **y**, as defined in Equation 2.2.

$$d(\mathbf{x}, \mathbf{y}) = 0, \quad \text{iff } \mathbf{x} = \mathbf{y} \qquad \text{Equation 2.2}$$

iii. **Symmetry**: The distance between **x** and **y** is equal to the distance between **y** and **x**, as shown in Equation 2.3.

$$d(\mathbf{x}, \mathbf{y}) = = d(\mathbf{y}, \mathbf{x}) \qquad \text{Equation 2.3}$$

iv. **Triangle inequality**: The triangle inequality states that the shortest distance between any two points is a straight line**.** Given the presence of a third point, z, the distance between x and z is less than or equal to the sum of the distance between x and y and y and z, as illustrated in Equation 2.4.

$$d(x, z) <= d(x, y) + d(y, z) \qquad \text{Equation 2.4}$$

(Prasath *et al.*, 2017; Chumachenko, 2017).

When the distance is in a range, say, [0, 1], the calculation of the corresponding similarity measure**,** **s**, is defined in Equation 2.5.

$$s(x, y) = 1 - d(x, y) \qquad \text{Equation 2.5}$$

Performance of KNN also depends significantly on the distance measure used. In addition, KNN is tolerant to noise, with a performance reduction of only 20% when up to 90% noise is infused into both training and test datasets (Prasath *et al.*, 2017). Within the eight families of distance measure, there are many members in each family, as illustrated in the Minkowski family.

2.9.1.4 Minkowski Distance Measures

Minkowski distance measure family consists of three distinct members

i. Manhattan measure

ii. Euclidean measure

iii. Chebyshev measure

These measures are derived from the variation of p value in Minkowski formula, as illustrated in Equation 2.6.

$$D_{Mink}(x, y) = \sqrt[p]{\sum_{i=1}^{n} |xi - yi|^p} \qquad \text{Equation 2.6}$$

The distance measures above are derived from Equation 2.14, when the p value changes: when p=1, the Minkowski formula becomes Manhattan formula, but when p=2, the same Equation 2.14 becomes Euclidean formula; and when p=∞, it becomes Chebyshev formula. Xi is the i-th value in the vector **x,** and y_i is the i-th value in the vector **y**.

2.9.1.5 Manhattan distance measure

This measure is also known as the L_1 norm, and it represents the sum of the absolute difference between the opposite values in the vectors **x** and **y.** This difference is illustrated in Equation 2.7.

$$D_{Manh}(x, y) = \sum_{i=1}^{n} |xi - yi| \qquad \text{Equation 2.7}$$

2.9.1.6 Euclidean distance measure

This distance measure is also known as L_2 norm; and it is an extension of Pythagoras theorem as shown in Equation 2.8.

$$D_{Eucl}(x, y) = \sqrt{((x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2)}$$

$$= \sqrt{\sum_{i=1}^{n} |xi - yi|^2} \qquad \text{Equation 2.8}$$

(Zhang *et al.*, 2016; Prasath *et al,* 2017)

2.9.1.7 Chebyshev distance measure

This distance measure is also known as maximum value distance, Lagrange and Chessboard distance; and it is used when two objects are defined in different dimensions. It is illustrated in Equation 2.9.

$$D_{cheb}(x, y) = \max |xi - yi| \qquad \text{Equation 2.9}$$

2.9.1.8 L_1 Distance Measure Family

Has as an example the Lorentzian distance measure, which is measured by natural logarithm (ln) of the absolute difference between two vectors. From Equation 2.10 we see that one (1) is added to it to avoid negative values or log of zero.

$$D_{Lore}(x, y) = \sum_{i=1}^{n} ln(1 + |xi - yi|) \qquad \text{Equation 2.10}$$

Distance functions or metrics use categorical data or numerical data to determine the classes in a training dataset.

2.9.1.9 Categorical Data Type

In machine learning, various types of data are used to analyze malware effects on computers, which are categorized into independent, dependent and controlled variables. Depending on the type of data, in supervised learning, a set of known variables is labeled for instance, as benign or malicious. Current techniques in machine learning usually convert other forms of data to numerical data before preprocessing. These class labels are categorized by distance functions. Hamming distance measure is used for

categorical data; while Minkowski family of distance measures is used to categorize numerical data type.

Categorical variables are values of names or labels such as red, green, blue (for color); race, sex, age group, educational levels (Salvador-Meneses *et al.*, 2019). There are three types of categorical variables: nominal, ordinal and binary variables. A nominal variable has no intrinsic ordering, for example, gender with two categories (male and female). An ordinal variable has clear ordering, for example, educational level may be categorized into primary education (1 to 6 years), junior secondary (1 to 3 years), and senior secondary (1 to 3 years). Variables can be independent, dependent or controlled variables.

Continuous variables are numerical variables that have an infinite number of values within a range [0, 1]. Discrete variables are numeric variables that have countable number of variables between any two variables. For categorical values, hamming distance measure is used to compute the distance between two variables or points. Hamming distance is the number of mismatches between two equal strings in same position. XOR operator can also be used to compute the hamming distance. This computation is illustrated in examples that follow. The hamming distance measure is given in Equation 2.11.

$$D_{ham}(x_i, y_i) = \begin{cases} 1, & \text{If } x_i \neq y_i \\ 0, & \text{If } x_i = y_i \end{cases} \qquad \text{Equation 2.11}$$

The computation of Hamming distance is illu ıle 7 with many examples.

Example 1:

Table 2.7: Hamming Distance

X	Y	Distance
Male	Male	0
Male	Female	1

Example2:

W o r k $D_{ham} = 2$

W a l k

Example 3:

L a n e $D_{ham} = 2$

V a l e

Example 4: Using binary digits

0 1 1 0 0 1 0 0 $D_{ham} = 3$

0 0 1 0 0 0 0 1

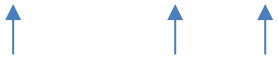


Table 2.8 illustrates the computation of hamming distance using the XOR operator

Table 2.8: Using XOR Operator to Compute Hamming Distance

INPUT		OUTPUT	
X	Y		
0	0	0	$D_{ham} = 2$
0	1	1	
1	0	1	
1	1	0	

As can be seen, with the XOR operator, when the inputs are the same, the output is zero, and vice versa. When the i-th attribute has numerical values, the range normalization difference distance is used, as illustrated in Equation 2.12

$$\mathbf{D}_N(xi - yi) = \frac{|xi - yi|}{\max(\{xi\}) - \min(\{yi\})} \quad \text{Equation 2.12}$$

Euclidean distance measure is most commonly used in conjunction with categorical data to determine classes of training data sets. However, the choice of these metrics (distance measures), given the type of data used, is influenced by the value (number) of nearest neighbors, **k**, in the training data set to the test data being analyzed.

2.9.1.10 Choosing the k Value (The Optimal Parameter)

K refers to the number of neighbors to consider for classification. The number should be odd, to avoid a tie. However, choosing k has to be done with care. If the value of k is small, it will cause low bias and high variance; resulting in an over fitting model. Similarly, if the value of k is very large, it will lead to high bias and low variance; resulting in an under fitting model.

Most researchers suggest that the value of K should be the square root of the total number of data points, n, above 100 (Prasath *et al.*, 2017; Zhang, 2016; Cheng *et al*, 2014). This is illustrated in Equation 2.13.

$$K = \sqrt{n} \quad \text{Equation 2.13}$$

Commonly used values of K as input include 3, 5, and 9. However, Cheng *et al.* (2014) is of the opinion that adopting Equation 2.13 is not to the best interest of all, as it does not hold in all cases. The paper further argued that one of the methods of classifying test data using fixed K value gives a serious drawback, as the values of the test data might differ from one another. The paper rather suggested that the value of K should be data driven. That is, it should be decided by the distribution of data. (Cheng *et al.*, 2014).

Cross validation is another technique used to determine the value of K, however, this too has the drawback of not considering the correlation of samples. Correlation coefficient measures the strength (norm) and direction of linear relationship between two numeric variables x and y.

2.9.1.11 K-NN and Distance Measure

K-Nearest Neighbor algorithm classifies an unlabeled test sample based on the majority of similar samples among the K nearest neighbors that are closest to the test sample (Prasath *et al.*, 2017). The distance between the test sample and each of the training samples in memory is determined by a specific distance measure. Figure 16 illustrates KNN classification with K=3 and K=5 values using Euclidean distance measure.

In Figure 2.16 there are two classes: stars and triangles, the test sample is represented by a filled circle.

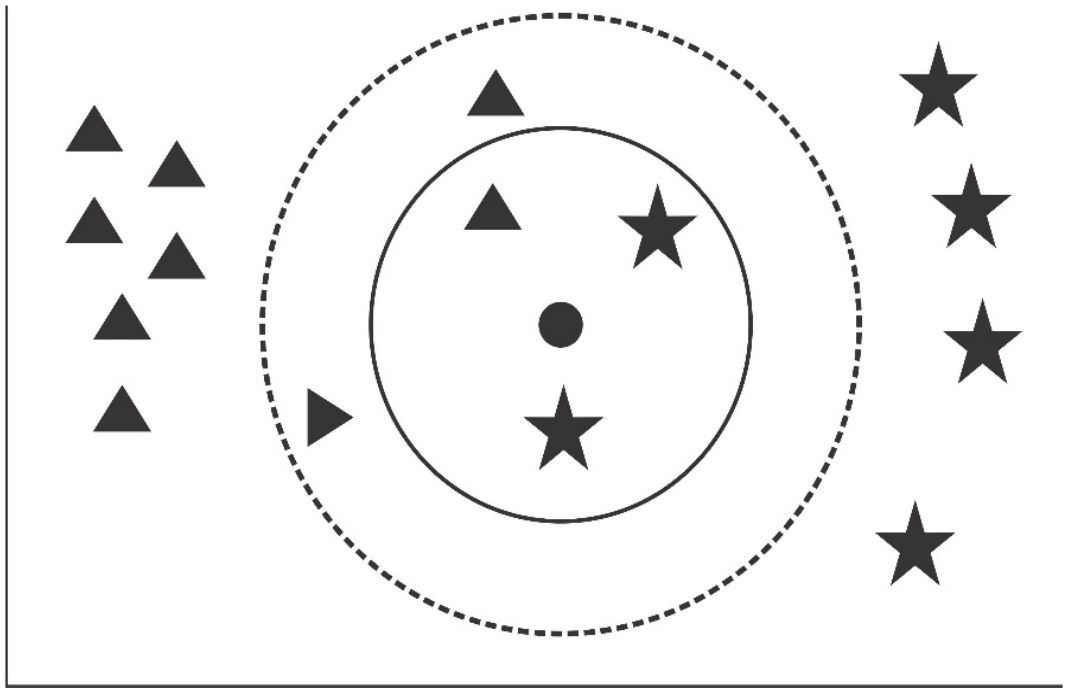


Figure 2.16: K-NN Classification with k=3 and k=5 using Euclidean Distance Measure

Source: Researcher (2024)

Finding the majority class among the K nearest neighbors predicts the class of the test sample. In Figure 2.16, where K=3, the test sample classifies to the star class, as it has the majority vote; but where K=5, the test sample classifies to the triangle class since it has the majority vote. It is noted that the dotted circle houses five objects (including both stars and triangles, while the inner circle houses three objects surrounding the filled circle test object. Algorithm to implement KNN is illustrated in subsection 2.9.1.12

2.9.1.12 K-NN Algorithm

The following steps depicts the process of measuring KNN distance to the target object:

- **i.** Load the data into the program (Application software implementing KNN)
- **ii.** Initialize the number of neighbors to be considered, K, which must be odd value.
- **iii.** To each entry or data point (or tuple) in the data file, do this:
    - a Calculate the distance between the data point (tuple) to be classified (test sample) and each data point in the training data file in memory
    - b Then add the computed distances in the training data file (i.e., add the column measures of each record)
    - c Sort the computed data points in ascending order by distances
    - d Pick the first K entries from the sorted data
    - e Observe the class of the majority vote (labels) and assign the test sample to it.

2.9.2 Naïve Bayes (NB) (Kohavi, 1996)

This soft computing algorithm relies on the Bayes theorem in its classification. It can be used for both binary and multi-class problems. Naïve Bayes method evaluates the probability of each feature independently, regardless of any correlation, and makes the

prediction based on Bayes theorem. This algorithm is based on the concept of class probabilities and conditional probabilities. A class probability is the probability of a class in the dataset. In other words, if we select a random item from the dataset, this is the probability of it belonging to a certain class. Conditional probability is the probability of the feature value, given the class.

Class probability is calculated as the number of samples in the class divided by the total number of samples, as illustrated in Equation 2.14

$$p(C) = \frac{\text{Count (instances in C)},}{\text{Count (instances in N total)}} \qquad \text{Equation 2.14}$$

Conditional probabilities are calculated as the frequency of each attribute value divided by the frequency of instances of that class. This is shown in Equation 2.15

$$p(V|C) = \frac{\text{Count (instances with V and C)}}{\text{Count (instances with V)}} \qquad \text{Equation 2.15}$$

Given the probabilities, we can calculate the probability of an instance belonging to a class, and take decision based on Bayes theorem, as shown in Equation 2.16

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \qquad \text{Equation 2.16}$$

The probabilities of the items belonging to all classes are then compared and the class with the highest probability is selected. This method is simple and easy to understand. It also performs well on noisy or irrelevant data features. Its main drawback is that each feature is treated independently, although in most cases this cannot be true (Narudin *et al.,* 2014; Nasteski, 2017; Berrar, 2018).

2.9.3 Decision Tree (J48) (Kotsiantis, 2007)

Decision trees are algorithms that have the structure of a tree. Training data set is used for the creation of the tree, which is subsequently used to make predictions on the unseen dataset (new data set). In this algorithm, the goal is to achieve the most accurate

results with the least number of decisions that must be made. The dataset used is tabulated.

The training set constitutes a table with a number of observations (rows) with various pieces of information about them called columns. One of these columns is the dependent variable, sometimes called ‘target’ or ‘label’ or ‘class’ variable. The other variables are the independent variables or features. The columns are usually many; say 80, out of which 79 will be independent variables, while only one will be the dependent variable. There is also test dataset, which also has observations with the same number of independent variables as that of the training dataset, but has no dependent variable; it is the job of the trained model to be able to predict the dependent variable in the test dataset as accurately as possible.

Therefore, the model, when fitted into the training dataset, learns the relationship between the independent variables and the dependent variable, and use that knowledge to predict the dependent (or target) variable for the test data set as accurately as possible.

Figure 2.17 illustrates a decision tree with a dataset (max-depth) of three.

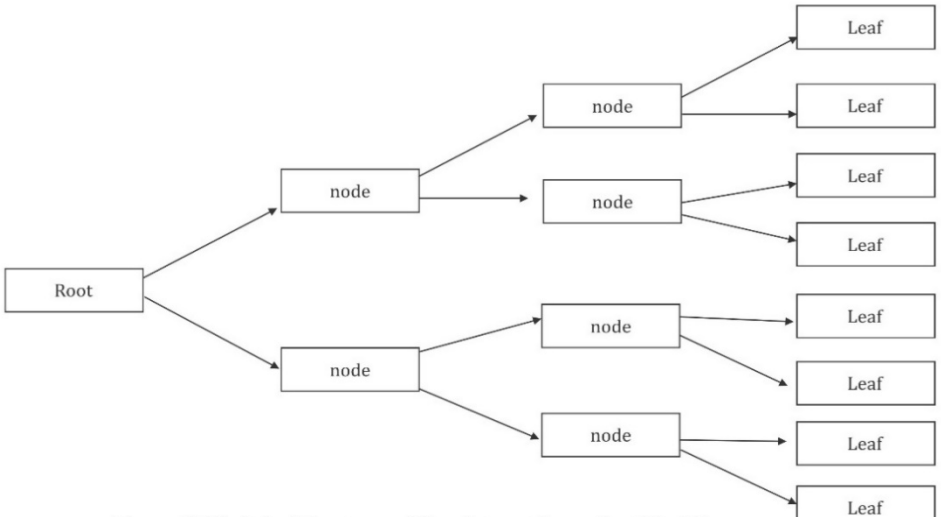


Figure 2.17: Decision Tree with a Dataset of Three Depth Three
Source: Kotsiantis (2007)

In each node, there are the following elements:

i. The node's split rule (the independent variable and its value)
ii. The Mean Square Error (MSE) of all the observations in that node
iii. Sample, being the number of observations in that node – The size of the group
iv. The target (dependent variable) – This is computed using natural logarithm

As the tree expands, it is observed that the MSE value decreases.

The common algorithm for decision trees is iterative dichotomizer 3 (ID3). It relies on the concepts of entropy and information gain. Entropy refers to the level of uncertainty in the data content.

Decision Tree has the following advantages:

i. It is simple
ii. It can deal with large dataset
iii. It can handle noise in the dataset
iv. It operates in a 'white box', unlike other algorithms like SVM and ANN. That is, it is transparent, and one can see clearly how the outcome is obtained.
v. It works with categorical variables

The disadvantages of Decision Trees include

i. It cannot predict numerical values
ii. It does not support online learning. That is, you have to rebuild your tree whenever you have new data set.
iii. Decision tree easily over fits
iv. Decision trees are slow, especially if they are complex with many branches.

2.8.4 Random Forest (RF) (Breiman, 2001)

In soft computing, there are meta-models that combine the predictions of several smaller models to generate a better model that will provide a final prediction from data it did not train with. This is called “ensemble learning”. Several decision trees are often combined together in an ensemble method called “bootstrap aggregation” or “Bagging”. The resulting aggregation is called Random Forest. Random forests are simple but effective. When being fitted to a training dataset, many decision trees are constructed, with each tree being fitted in a random subset of the data.

To generate a prediction for a new observation, the Random Forest simply averages the predictions of all its trees and returns that as its prediction. This works well because the bootstrap sampling and the feature subset are meant to make the trees as uncorrelated as possible (although they are all still based on the same dataset and feature set), allowing each tree to discover slightly different relationship in the data. This results in their average having less variance – fewer overflows – than any single tree, and therefore better generalization and prediction. It is observed that each tree is structured differently on different dataset. Hence, each tree will define similarity in a different way and predict a different value. Therefore, for a given unseen observation, the average of all the trees is basically the average of all the values of a lot of observations in the training set, which are somehow similar to it. It is a machine learning classifier frequently used in malware detection. One consequence about Random Forest is, its prediction is very poor when the test dataset is different in some fundamental way from the training set – different range of values.

2.9.5 Logistic Regression

Logistic regression is a supervised learning algorithm that solves classification problems. It is used for predicting categorical dependent variable using a given set of independent variables. The outcome of its prediction is either categorical or numeric

(discrete) values such as YES or NO, True or False, 0 or 1. However, it gives a probabilistic value that lies within the range [0, 1].

Logistic regression differs from linear regression. While logistic regression solves classification problems, linear regression solves regression problems, which are continuous in nature. Logistic regression fits an "S" shape sigmoid function, which is represented in Equation 2.17

$$\log\left[\frac{y}{1-y}\right] = b_o + b_1x_1 + b_2x_2 + b_3x_3, + \ldots + b_nx_n \qquad \text{Equation 2.17}$$

Figure 2.18 depicts the "S" shape sigmoid function.

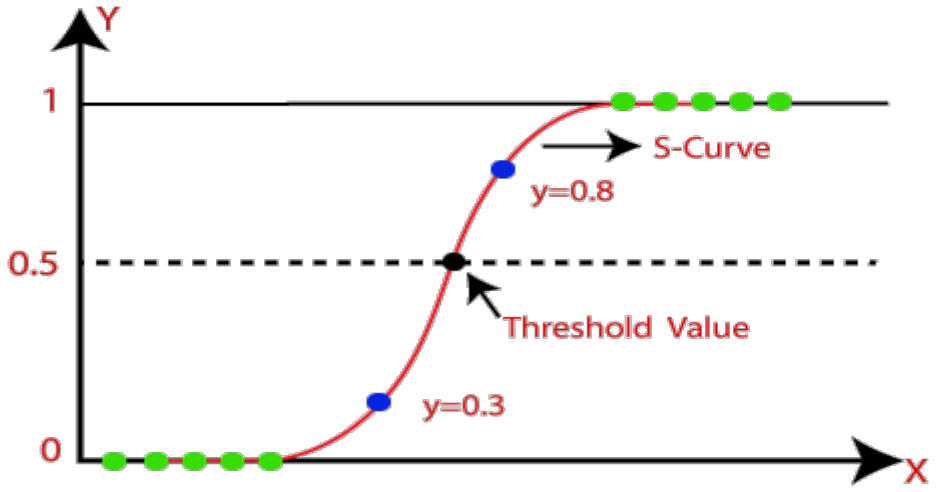


Figure 2.18: A Plot of Logistic Regression Sigmoid Function

In logistic regression, the concept of threshold value is used, and it defines the probability of either 0 or 1. Value above the threshold tends to 1, while those below the threshold tends to 0.

There are three types of logistic regression: binomial, multinomial, and ordinal logistic regressions.

i. Binomial LR has only two possible types of dependent variable, such as 0 or 1, True or False, Yes or No.

ii. Multinomial LR has a dependent variable with three or more possible unordered values such as Cat, Goad, or Sheep; train, car, tram, or byke (with transport as the dependent variable).

iii. Ordered LR has dependent variable with ordered values such as low, medium or high; red, blue, green (for color dependent variable).

Logistic regression can be applied in areas such as

i. Forecasting the effects or impact of specific changes

ii. Forecasting trends and future values

iii. Determining the strengths of different predictors

Advantages of Logistic Regression:

i. It is relatively fast compared to other supervised learning algorithms, such as SVM, K-NN, and ANN.

ii. It is easier to implement than other forms of ML algorithms

iii. It works well for cases where the dataset is linearly separable

iv. It provides useful insights. That is, it not only provides the magnitude and relevance of the independent variable, but its direction of the relationship with the dependent variable.

v. It works well with fairly large dataset

vi. It does not over fit, however, it over fits in non-linear spaces.

Disadvantages of Logistic regression:

i. It has poor accuracy

ii. It tends to underperform when the decision boundary is non-linear

iii. It does not work well with small dataset, which could result in over fitting

iv. It works well only when the dependent variable is categorical or dichotomous

v. It assumes linearity between predicted (dependent) variables and the independent variables, this is not so in real world situations.

2.9.6 Artificial Neural Networks (ANN) (Hagan et al., 1996)

Artificial Neural network is inspired by the human brain. However, it takes much more time during training (Hira and Gillies, 2015; Aljawarneh *et al.,* 2017), as such, it is difficult to apply on handheld mobile phones where real-time is a constraint. But multilayer perceptron (MLP) is commonly used because it has medium level complexity. ANN is flexible and supports high degree of complexity, but it is complex and hard to interpret.

2.10 Data Collection

The data that is used in this research work is obtained from Kaggle repository; it is Network Security Laboratory - Knowledge Discovery in Databases (NSL-KDD) dataset. It is a balanced dataset with normal data type, and four anomalous data types: denial of service (DOS) data type, probe or surveillance data type, user to root (U2R) data type, and remote to local (R2L) data type (GitHub Inc. 2020). Table 2.9 illustrates the anomalous attack types

Table 2.9: Attack Types

s/n	Attack names	Attack Types
1	Back	DOS
2	Buffer-overflow	U2R
3	ftp_write	R2L
4	Guess_passwd	R2L
5	Imap	R2L
6	Ipsweep	Probe
7	Land	DOS
8	Loadmodule	U2R

9	Multihop	R2L
10	Neptune	DOS
11	Nmap	Probe
12	Perl	U2R
13	Phf	R2L
14	Pod	DOS
15	Portsweep	Probe
16	Rootkit	U2R
17	Satan	Probe
18	Smurf	DOS
19	Spy	R2L
20	Teardrop	DOS
21	Warezclient	R2L
22	Warezmaster	R2L

(Source: GitHub Inc., 2020)

The data obtained consists of two types of records or classes: normal (Benign) and anomaly (Malicious) with columns or features contained in attribute relations file format (ARFF).

The steps used to collect data from Kaggle repository include:

i. Download and collect benign and malicious applications
ii. Decompress (unzip) the applications to extract content using APK tool
iii. Extract permission request features from each application using read manifest.exe
iv. Build the dataset in an ARFF format and load in the computer device used.
v. Normalize the dataset and split into training and test datasets in the ratio of 80: 20
vi. Train algorithms using the training dataset
vii. Test the trained model for generalization using test dataset.

viii. Carry out performance evaluation on the tested dataset using confusion matrix.

ARFF file is an ASCII text file that describes the list of instances that shares a set of attributes. It is divided into two: header subsection and data subsection. The header describes the features of the dataset, while the data section describes the observations of the dataset as illustrated in Figure 2.19.

```
%       The Header
%       Title:      NSL-KDD Data set
%       Source:     GitHub Inc. (2020)
%
%
@RELATION       NSL-KDD Data set

@ATTRIBUTE      “duration”                      real
@ATTRIBUTE      “protocol_type”                 {tcp, udp, icmp}

@ATTRIBUTE      “service”                       {aol, auth, bgp, courier
…}
@ATTRIBUTE      “flag”                          {OTH, REJ, RSTO, RSTR
...}
@ATTRIBUTE      “src_bytes”                     real
@ATTRIBUTE      “dst_bytes”                     real
@ATTRIBUTE      “land”                          {‘0’, ‘1’}
@ATTRIBUTE      “wrong_fragment”                real
@ATTRIBUTE      “urgent”                        real
@ATTRIBUTE      “hot”                           real
@ATTRIBUTE      “num_failed_logins”             real
@ATTRIBUTE      “logged_in”                     {‘0’, ‘1’}
@ATTRIBUTE      “num_compromised”               real
@ATTRIBUTE      “root_shell”                    real
@ATTRIBUTE      “su_attempted”                  real
@ATTRIBUTE      “num_root”                      real
@ATTRIBUTE      “num_file_creations”            real
@ATTRIBUTE      “num_shells”                    real
@ATTRIBUTE      “num_access_files”              real
@ATTRIBUTE      “num_outbound_cmds”             real
@ATTRIBUTE      “is_host_login”                 {‘0’, ‘1’}
@ATTRIBUTE      “is_guest_login”                {‘0’, ‘1’}
@ATTRIBUTE      “count”                         real
@ATTRIBUTE      “srv_count”                     real
@ATTRIBUTE      “serror_rate”                   real
@ATTRIBUTE      “srv_serror_rate”               real
@ATTRIBUTE      “rerror_rate”                   real
@ATTRIBUTE      “srv_rerror_rate”               real
```

@ATTRIBUTE	“same_srv_rate”	real
@ATTRIBUTE	“diff_srv_rate”	real
@ATTRIBUTE	“srv_diff_host_rate”	real
@ATTRIBUTE	“dst_host_count”	real
@ATTRIBUTE	“dst_host_srv_count”	real
@ATTRIBUTE	“dst_host_same_srv_rate”	real
@ATTRIBUTE	“dst_host_diff_srv_rate”	real
@ATTRIBUTE	“dst_host_same_src_port-rate”	real
@ATTRIBUTE	“dst_host_srv_diff_host_rate	real
@ATTRIBUTE	“dst_host_serror_rate”	real
@ATTRIBUTE	“dst_host_srv_serror_rate”	real
@ATTRIBUTE	“dst_host_rerror_rate”	real
@ATTRIBUTE	“dst_host_srv_rerror_rate”	real
@ATTRIBUTE	“class”	{‘normal’, ‘anomaly’}

% *The Body of ARFF*

@DATA

0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,150,25,0.17,0.03,0.17,0.00,0.00,0.00,0.05,0.00,normal,20

0,udp,other,SF,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,1,0.00,0.00,0.00,0.00,0.08,0.15,0.00,255,1,0.00,0.60,0.88,0.00,0.00,0.00,0.00,0.00,normal,15

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,123,6,1.00,1.00,0.00,0.00,0.05,0.07,0.00,255,26,0.10,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,http,SF,232,8153,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,5,0.20,0.20,0.00,0.00,1.00,0.00,0.00,30,255,1.00,0.00,0.03,0.04,0.03,0.01,0.00,0.01,normal,21

0,tcp,http,SF,199,420,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,30,32,0.00,0.00,0.00,0.00,1.00,0.00,0.09,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,121,19,0.00,0.00,1.00,1.00,0.16,0.06,0.00,255,19,0.07,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,166,9,1.00,1.00,0.00,0.00,0.05,0.06,0.00,255,9,0.04,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,117,16,1.00,1.00,0.00,0.00,0.14,0.06,0.00,255,15,0.06,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,remote_job,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,270,23,1.00,1.00,0.00,0.00,0.09,0.05,0.00,255,23,0.09,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,133,8,1.00,1.00,0.00,0.00,0.06,0.06,0.00,255,13,0.05,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,205,12,0.00,0.00,1.00,1.00,0.06,0.06,0.00,255,12,0.05,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,199,3,1.00,1.00,0.00,0.00,0.02,0.06,0.00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21 *

0,tcp,http,SF,287,2251,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0.00,0.00,0.00,0.00,1.00,0.00,
0.43,8,219,1.00,0.00,0.12,0.03,0.00,0.00,0.00,0.00,normal,21

0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,2,20,1.00,0.00,1.00,0.20,0.00,0.00,0.00,0.00,warezclient,15

0,tcp,name,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,233,1,1.00,1.00,0.00,0.00,0.00,0.06,0.0
0,255,1,0.00,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,netbios_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,96,16,1.00,1.00,0.00,0.00,0.17,0.0
5,0.00,255,2,0.01,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,tcp,http,SF,300,13788,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,9,0.00,0.11,0.00,0.00,1.00,0.00
,0.22,91,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21

0,icmp,eco_i,SF,18,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.
00,1,16,1.00,0.00,1.00,1.00,0.00,0.00,0.00,0.00,ipsweep,18

0,tcp,http,SF,233,616,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.00,0.00,0.
00,66,255,1.00,0.00,0.02,0.03,0.00,0.00,0.02,0.00,normal,21

0,tcp,http,SF,343,1178,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,10,0.00,0.00,0.00,0.00,1.00,0.00
,0.20,157,255,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21

0,tcp,mtp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,223,23,1.00,1.00,0.00,0.00,0.10,0.05,0.0
0,255,23,0.09,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,280,17,1.00,1.00,0.00,0.00,0.06,0.05,
0.00,238,17,0.07,0.06,0.00,0.00,0.99,1.00,0.00,0.00,neptune,18

0,tcp,http,SF,253,11905,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,10,0.00,0.00,0.00,0.00,1.00,0.0
0,0.20,87,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21

5607,udp,other,SF,147,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,255,1,0.00,0.85,1.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,mtp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,248,2,1.00,1.00,0.00,0.00,0.01,0.06,0.00
,255,2,0.01,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

507,tcp,telnet,SF,437,14421,0,0,0,0,0,1,3,0,0,0,0,0,1,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,25,0.10,0.05,0.00,0.00,0.53,0.00,0.02,0.16,normal,20

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,279,7,1.00,1.00,0.00,0.00,0.03,0.06,0.
00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,http,SF,227,6588,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,22,0.00,0.00,0.00,0.00,1.00,0.00
,0.18,43,255,1.00,0.00,0.02,0.14,0.00,0.00,0.56,0.57,normal,21

0,tcp,http,SF,215,10499,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,14,14,0.00,0.00,0.00,0.00,1.00,0.
00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,241,1400,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,33,33,0.00,0.00,0.03,0.03,1.00,0.0
0,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,12,0.00,0.00,0.00,0.00,1.00,0.00,1.
00,1,53,1.00,0.00,1.00,0.51,0.00,0.00,0.00,0.00,ipsweep,17

0,tcp,finger,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,57,16,1.00,1.00,0.00,0.00,0.28,0.07,0.
00,255,59,0.23,0.04,0.01,0.00,1.00,1.00,0.00,0.00,neptune,20

0,tcp,http,SF,303,555,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,9,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,255,1.00,0.00,0.11,0.01,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0.00,0.00,1.00,1.00,0.50,1.00,0.00,255,1,0.00,0.31,0.28,0.00,0.00,0.00,0.29,1.00,portsweep,20

0,udp,domain_u,SF,45,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,181,181,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,250,0.98,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,18

1,udp,private,SF,105,147,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,41,5,0.12,0.05,0.05,0.00,0.00,0.00,0.00,0.00,normal,19

0,udp,domain_u,SF,43,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,122,202,0.00,0.00,0.00,0.00,1.00,0.00,0.01,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal,18

0,tcp,supdup,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,205,17,1.00,1.00,0.00,0.00,0.08,0.06,0.00,255,17,0.07,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,http,SF,324,2302,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,22,28,0.00,0.00,0.00,0.00,1.00,0.00,0.14,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,uucp_path,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,228,13,1.00,1.00,0.00,0.00,0.06,0.07,0.00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,20

0,tcp,ftp_data,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,50,11,1.00,1.00,0.00,0.00,0.22,0.08,0.00,255,63,0.25,0.02,0.01,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,Z39_50,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,262,12,1.00,1.00,0.00,0.00,0.05,0.06,0.00,255,9,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

2,tcp,smtp,SF,1591,372,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.00,1.00,181,147,0.81,0.02,0.01,0.00,0.00,0.00,0.00,0.00,normal,21

9052,udp,other,SF,146,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,2,0.00,0.00,0.00,0.00,0.50,0.50,0.00,255,2,0.01,0.66,0.99,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,290,3006,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,csnet_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,108,11,1.00,1.00,0.00,0.00,0.10,0.08,0.00,255,11,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,80,80,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,80,0.31,0.02,0.31,0.00,0.00,0.00,0.00,0.00,teardrop,16

0,tcp,http,SF,255,861,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,44,44,0.02,0.02,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,4,38,1.00,0.00,1.00,0.18,0.00,0.00,0.00,0.00,warezclient,12

0,tcp,ftp_data,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,258,17,1.00,1.00,0.00,0.00,0.07,0.05,0.00,255,5,0.02,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,tcp,http,SF,302,498,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,10,0.00,0.00,0.00,0.00,1.00,0.00,0.00,131,255,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,61,4,0.00,0.00,1.00,1.00,0.07,0.10,0.00,255,4,0.02,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,2,0.01,0.02,0.01,0.00,0.00,0.00,0.77,0.00,teardrop,15

0,tcp,http,SF,220,1398,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,26,42,0.00,0.00,0.00,0.00,1.00,0.00,0.05,26,255,1.00,0.00,0.04,0.03,0.00,0.00,0.00,0.00,normal,21

0,udp,domain_u,SF,43,69,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,120,120,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,245,0.96,0.01,0.01,0.00,0.00,0.00,0.00,0.00,normal,18

0,udp,domain_u,SF,44,133,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,73,75,0.00,0.00,0.00,0.00,1.00,0.00,0.03,122,212,0.88,0.02,0.88,0.01,0.00,0.00,0.08,0.00,normal,21

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,15,0.00,0.00,0.00,0.00,1.00,0.00,1.00,2,46,1.00,0.00,1.00,0.26,0.00,0.00,0.00,0.00,nmap,17

0,tcp,uucp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,135,9,1.00,1.00,0.00,0.00,0.07,0.06,0.00,255,11,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,20

0,tcp,finger,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,24,12,1.00,1.00,0.00,0.00,0.50,0.08,0.00,255,59,0.23,0.04,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,udp,domain_u,SF,43,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,148,228,0.00,0.00,0.00,0.00,1.00,0.00,0.01,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal,18

0,tcp,ftp_data,SF,641,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,175,48,0.25,0.02,0.25,0.04,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,206,4,0.00,0.00,1.00,1.00,0.02,0.06,0.00,255,4,0.02,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,175,1,0.10,0.00,0.89,1.00,0.01,1.00,0.00,255,1,0.00,0.84,0.00,0.00,0.07,0.00,0.62,1.00,satan,18

0,tcp,http,SF,310,2794,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,63,10,0.00,0.00,1.00,1.00,0.16,0.08,0.00,255,10,0.04,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,smtp,SF,696,333,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,109,133,0.39,0.04,0.01,0.02,0.00,0.00,0.00,0.00,normal,21

0,tcp,netbios_dgm,RSTR,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.00,0.00,0.00,236,1,0.00,0.58,0.58,0.00,0.00,0.00,0.58,1.00,portsweep,14

0,tcp,http,SF,209,442,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,26,26,0,0,0,0,1,0,0,235,255,1,0,0,0.02,0,0,0,0,normal,21

5,tcp,pop_3,SF,26,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,233,214,0.92,0.01,0,0,0,0,0.04,0,guess_passwd *

0,tcp,http,SF,327,294,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,10,0,0,0,0,1,0,0.2,255,255,1,0,0,0,0,0,0,normal,21

4,tcp,pop_3,SF,32,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,161,0.63,0.02,0,0,0,0,0.13,0,guess_passwd,15 *

0,udp,private,SF,52,54,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,0,0,0,1,0,0,25
5,255,1,0,0.87,0,0,0,0,0,normal,17 *

0,tcp,http,SF,295,527,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0,0,0,0,1,0,0,124,255,
1,0,0.01,0.03,0,0,0,0,normal,21

0,tcp,http,SF,316,356,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0,0,0,0,1,0,0,38,255,1,
0,0.03,0.02,0,0,0,0,normal,

0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,10,0,0,0,0,1,0,0,35,10,
0.29,0.11,0.29,0,0,0,0,0,teardrop,11

3,tcp,pop_3,SF,30,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,247,0
.97,0.01,0,0,0,0,0.02,0,guess_passwd,18 *

1,tcp,telnet,RSTO,123,178,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,255,
12,0.05,0.01,0,0,0,0,0.03,0.67,guess_passwd,13 *

0,tcp,http,SF,311,294,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,27,29,0,0,0,0,1,0,0.1,255,
255,1,0,0,0,0,0,0,0,normal,21 *

0,tcp,http,SF,227,406,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,34,0,0,0,0,1,0,0.09,64,2
55,1,0,0.02,0.03,0,0,0,0,normal,21

0,udp,private,SF,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,5,0,0,0,0,1,0,0,255,253,
0.99,0.01,0,0,0,0,0,0,snmpguess,13 *

0,tcp,http,SF,314,358,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,10,0,0,0,0,1,0,0,231,25
5,1,0,0,0.02,0,0,0,0,normal,21

0,tcp,http,SF,243,269,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,21,21,0,0,0,0,1,0,0,255,25
5,1,0,0,0,0,0,0,0,normal,21

0,tcp,ftp_data,SF,19,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,122,15,0.
04,0.05,0.04,0.13,0,0,0,0,normal,21 *

210,tcp,telnet,SF,126,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,2
48,0.97,0.01,0,0,0,0,0.02,0.02,guess_passwd,16

0,tcp,http,SF,203,454,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,11,0,0,0,0,1,0,0.18,255
,255,1,0,0,0,0,0,0,0,normal,21

0,tcp,finger,SF,5,381,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,2,0.0
1,0.03,0,0,0.02,0,0.7,0,normal,16 *

0,tcp,http,SF,279,526,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,13,0,0,0,0,1,0,0.15,2,25
5,1,0,0.5,0.02,0,0,0,0,normal,21

0,udp,private,SF,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,253,
0.99,0.01,0.01,0,0,0,0,0,snmpguess,14 *

1,tcp,telnet,SF,24,715,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0.67,0.33,0.33,1,0.
67,255,67,0.26,0.02,0,0,0,0.01,0.7,0.69,mscan,11 *

0,udp,domain_u,SF,46,86,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,248,246,0,0,0,0,0.99,0
.01,0,255,254,1,0.01,0,0,0,0,0,0,normal,18 *

4,tcp,pop_3,SF,28,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,90,77,0.8
4,0.03,0.01,0.03,0,0,0.11,0,guess_passwd,7

4,tcp,pop_3,SF,25,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,70,67,0.9
4,0.03,0.01,0.03,0,0,0,0,guess_passwd,6

0,udp,private,SF,49,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,4,0,0,0,0,1,0,0,255,254,
1,0.01,0.01,0,0,0,0,0,snmpguess,16

2,tcp,imap4,RSTO,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,6,0,0,0.67,1,0.33,1,1,5
1,54,0.27,0.1,0.02,0.04,0,0,0.92,0.26,mscan,12 *

0,tcp,telnet,SF,125,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,109
,0.43,0.02,0,0,0,0,0.01,0.03,guess_passwd,10

4,tcp,pop_3,SF,31,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,208,0
.82,0.02,0,0,0,0,0.07,0,guess_passwd,18

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,27,0,0,0,0,1,0,1,2,60,1,0,1
,0.5,0,0,0,0,ipsweep,17 *

0,udp,domain_u,SF,46,77,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,188,188,0,0,0,0,1,0,0,
255,255,1,0,0.01,0,0,0,0,0,normal,18 *

0,udp,domain_u,SF,45,82,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,159,160,0,0,0,0,0.99,0
.01,0.02,255,250,0.98,0.01,0,0,0,0,0,0,normal,18 *

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,21,0,0,0,0,1,0,1,2,42,1,0,1
,0.26,0,0,0,0,nmap,16

0,tcp,ftp_data,SF,43,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,62,0.
24,0.02,0.01,0,0.87,0.97,0,0,normal,14 *

0,udp,private,SF,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,156,146,0,0,0,0,0.94,0.01,0
,255,208,0.82,0.03,1,0,0,0,0,0,satan

0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,5,17,1,0
,1,0.12,0,0,0,0,warezclient,13 *

0,tcp,finger,SF,9,196,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,5,5,0,0,0,0,1,0,0,155,7,0.0
5,0.06,0.01,0,0.01,0,0,0,satan,0 *

0,icmp,ecr_i,SF,520,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,184,184,0,0,0,0,1,0,0,255,
255,1,0,1,0,0,0,0,0,smurf,18 *

0,icmp,eco_i,SF,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,3,0.0
1,0.02,0.01,0,0,0,0,0,satan,6 *

1,tcp,ftp,RSTO,0,133,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,2,2,0.5,1,0.5,1,0,0,1,1,ipsweep,9 *

0,udp,domain_u,SF,34,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,0,0,0,0,1,0,1,213,22,0.1,0.03,0.1,0,0,0,0,0,normal,21 *

0,icmp,ecr_i,SF,1480,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,1,25,1,0,1,0.52,0,0,0,0,pod,17

0,tcp,pop_3,RSTO,0,36,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0,0,1,1,0.33,1,1,189,60,0.24,0.03,0.01,0.03,0,0,0.88,0.98,mscan,15 *

0,tcp,ftp,SF,26,157,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,0,155,71,0.46,0.03,0.01,0,0,0,0,0,guess_passwd,7

0,tcp,telnet,S3,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,1,0,0,255,117,0.46,0.02,0,0,0.53,0.37,0.01,0.01,processtable,18 *

0,udp,private,SF,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,254,1,0.01,0,0,0,0,0,0,snmpguess,12 *

0,tcp,telnet,RSTO,124,188,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,255,254,1,0.01,0,0,0.01,0.01,0.02,0.02,guess_passwd,10

8169,tcp,telnet,SF,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,19,0.07,0.82,0,0,0,0,0.8,0,processtable,12 *

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,192,191,0.99,0.01,0.01,0,0,0,0,0,snmpgetattack,2 *

1,tcp,telnet,RSTO,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7,0,0,1,0.57,1,0,0.57,96,87,0.34,0.05,0.01,0.02,0,0,0.9,0.7,mscan,11

0,tcp,pop_3,SF,30,217,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,2,0.01,0.02,0,0,0,0,0,0,guess_passwd,4

0,udp,other,SF,23,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,1,0,0.03,0,0,0,0,0,0,multihop,9

0,tcp,telnet,SF,123,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,49,0.19,0.02,0,0,0,0,0.01,0.04,guess_passwd,11

0,udp,private,SF,105,147,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,237,235,0.99,0.01,0,0,0,0,0,0,snmpgetattack,1

,tcp,telnet,SF,6,54,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,5,0,0.2,0.5,0.4,0.5,0.75,0.6,255,83,0.33,0.1,0,0,0,0,0.33,0.48,mscan,11

0,udp,private,SF,47,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,255,
1,0,0.01,0,0,0,0,0,snmpguess,15

0,tcp,telnet,SF,122,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,25,
0.1,0.02,0,0,0,0,0,0.04,guess_passwd,9

0,tcp,telnet,SF,120,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,115
,0.45,0.02,0,0,0,0,0.01,0.03,guess_passwd,11

0,icmp,ecr_i,SF,508,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,182,182,0,0,0,0,1,0,0,255,
255,1,0,1,0,0,0,0,0,smurf,18 *

1,tcp,smtp,SF,2599,293,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0,0,1,0,0,255,13
8,0.54,0.15,0,0,0,0,0.44,0,mailbomb,11

0,udp,other,SF,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,140,1,0,0,0,0,0.01,0.01,0,8,2,
0.25,0.62,0.25,0,0,0,0.12,0,satan,10

0,icmp,eco_i,SF,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,55,0,0,0,0,1,0,1,3,47,1,0,
1,0.26,0,0,0,0,saint,15

0,udp,private,SF,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0,0,1,0,0,255,255,
1,0,0.02,0,0,0,0,0,snmpguess,16 *

8,tcp,ftp,SF,220,688,0,0,0,4,0,1,0,0,0,0,4,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,53,27,0.51,0.08,0.02,0.00,0.00,0.00,0.00,0.00,buffer_overflow,1
*

0,udp,private,SF,47,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.
00,0.00,0.00,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,15

0,tcp,ftp_data,SF,12,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.
00,0.00,0.00,7,25,1.00,0.00,1.00,0.12,0.00,0.00,0.00,0.00,warezmaster,12

0,tcp,ftp,SF,26,157,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,129,0.51,0.03,0.00,0.00,0.00,0.00,0.00,0.00,guess_passwd,9

0,udp,private,SF,46,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.
00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,snmpguess,15
*

282,tcp,ftp,SF,163,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1
.00,0.00,0.00,255,5,0.02,0.24,0.00,0.00,0.34,0.00,0.24,0.00,warezmaster,15

0,udp,other,SF,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,145,1,0.00,0.00,0.00,0.00,0.0
1,1.00,0.00,255,1,0.00,0.62,0.99,0.00,0.00,0.00,0.00,0.00,satan,19

282,tcp,ftp,SF,161,597,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1
.00,0.00,0.00,255,97,0.38,0.03,0.00,0.00,0.00,0.00,0.00,0.00,warezmaster,13

282,tcp,ftp,SF,164,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1
.00,0.00,0.00,255,31,0.12,0.03,0.00,0.00,0.40,0.00,0.02,0.00,warezmaster,15

,tcp,ftp,SF,162,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,17,0.07,0.02,0.00,0.00,0.14,0.00,0.25,0.00,warezmaster,12

282,tcp,ftp,SF,162,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,57,0.22,0.03,0.00,0.00,0.13,0.00,0.02,0.00,warezmaster,11

1179,udp,other,SF,146,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,33,14,0.42,0.12,1.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack,0

0,udp,private,SF,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack,1 *

0,udp,private,SF,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,5,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,15

0,icmp,ecr_i,SF,1008,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,303,303,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00,smurf,18 *

282,tcp,ftp,SF,158,597,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,72,0.28,0.03,0.00,0.00,0.05,0.00,0.02,0.00,warezmaster,12

0,udp,private,SF,49,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,253,0.99,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,14

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,253,0.99,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpgetattack,3

4,tcp,pop_3,SF,28,103,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,214,0.84,0.02,0.00,0.00,0.00,0.00,0.04,0.01,guess_passwd,15 *

282,tcp,ftp,SF,160,597,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,50,0.20,0.02,0.00,0.00,0.00,0.00,0.55,0.00,warezmaster,14

,udp,private,SF,41,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,12 *

0,icmp,eco_i,SF,30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,247,247,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00,normal,20

0,icmp,eco_i,SF,30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,227,227,1,0,1,0,0,0,0,normal,20 *

0,udp,domain_u,SF,35,91,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,4,0,0,0,0,1,0,0,75,255,249,0.98,0.01,0.01,0,0,0,0,0,normal,21 *

Figure 2.19: Attribute-Relation File Format (ARFF)
Source: GitHub Inc. (2020)

NSL-KDD dataset may not be a perfect dataset, but it is an improvement on the KDD'CUP 99 dataset, for it does not include redundant, noisy and missing records (Anthony, 2014; GitHub Inc, 2020). The dataset contains 262,178 attack records and 812,814 normal records in the training dataset; and 29,378 attack records and 49,911 normal records in the test dataset. These records contain forty-one (41) permission features. They are reduced to a fixed range of [0, 1] using normalization.

2.10.1 Data Preprocessing

Figure 2.20 depicts the processing techniques used in this research work. It consists of the dataset used to train and test the models, NSL-KDD dataset. The dataset is is then uploaded to the cloud via mobile host agent. In the cloud, the data is cleaned, preprocessed and extracted and balanced using SMOTE, Pearson correlation, and information gain (IG); then normalized (singh *et al.,* 2015; Ikram and Cherukuri, 2016), and selected to an appropriate size using PCA. The extracted data features are then split into training (80%) and test datasets (20%). The training dataset is used by random forest (RF), the training algorithm, which samples the data patterns and develop models to learn the patterns. The models are built from base algorithms such as decision tree (DT), K-Nearest Neighbors (K-NN), Naïve Bayes (NB) and Logistic Regression (LR). The trained models each predicts the labels in the test dataset. The predictions are then aggregated using voting classifier to a concensus model called soft vote. This aggregated soft model is trained and used to predict the labels in the test dataset into normal and malicious threats.

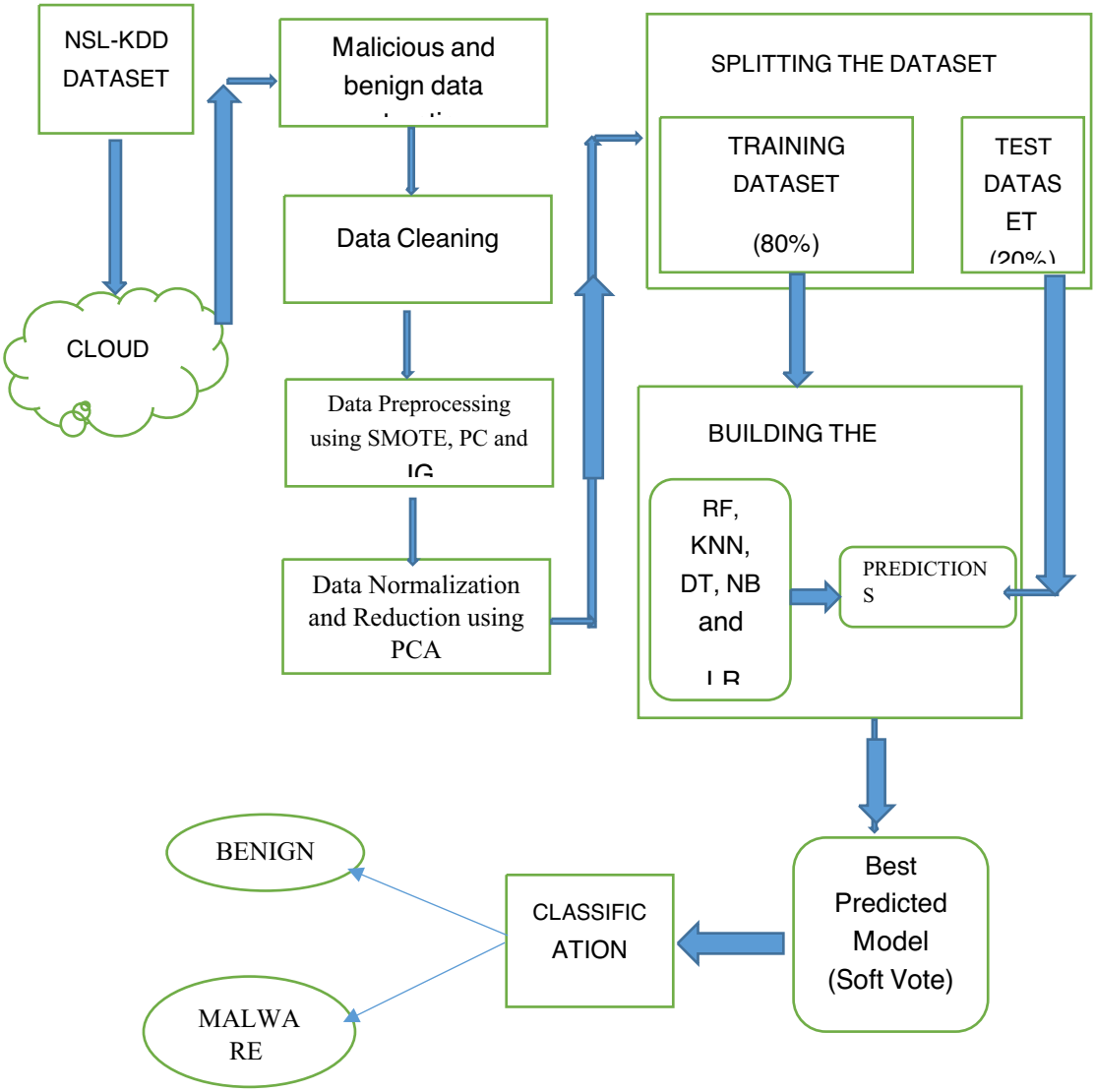


Figure 2.20: Normalization, Feature Reduction and Feature Classification
Source: Researcher (2024)

2.10.2 Min_Max Normalization

Min-Max normalization performs linear transformation on original data. Let (X_1, X_2) be min and max boundaries of an attribute (feature), and (Y_1, Y_2) be the new scale at

which normalization is done, then for V_i value of the attribute, the normalized value U_i is given in Equation 2.17.

$$U_i = \frac{Vi - X1}{X2 - X1} (Y_2 - Y_1) + Y_1 \quad \text{Equation 2.17}$$

To apply this model on the NSL-KDD dataset, features from the ARFF file are used, which not only define the attributes but also present the observations in comma separated values (CSV) format. The significant features to use without distorting information flow are described in Table 2.10. These significant features are extracted during the computation of principal components (CPs) by PCA using Eigen values and Eigenvectors.

Table 2.10: Significant Features

S/N	Attribute Name	Description
1	Src_bytes	This is the number of data bytes that is transferred from source to destination in a single connection.
2	Count	This is the number of connections to the same destination host, as the current connection in the past two seconds
3	Srv_count	This is the number of connections to the same service (port number), as the current connection in the past two seconds.
4	Logged_in	This indicates the log in status: 1 if successfully logged in, and 0 otherwise
5	Num_compromised	This is the number of compromised conditions, especially network systems infected and turned into zombies, then added into the botnet family. Another example is the “SQL injection” where malicious code is embedded into a normal application to corrupt it.
6	Dst_host_count	This is the number of connections having the same destination host IP address

7	Num_outbound_cmd	This is the number of outbound commands issued by command and control (C&C) servers in a file transfer (ftp) protocol session to steal data, customer contact list, and carry out nefarious acts like SEND_SMS, READ_CONTACT, WRITE_CALL_LOG.

These parameters are applied in Equation 2.17 where

X_1: is the *src_byte* feature, and is the initial number of bytes sent to the destination host

X_2: is the *count* feature, which is the number of times the destination host is connected

Y_1: is the initial *logged_in* feature, which is usually zero (0)

Y_2: is the successfully *logged in* status, which is one (1)

V_i: is the *destination host count* with the same IP address

U_i: is the number of *compromised* systems in a connection session of not more than two Seconds.

Using Neptune, a DOS attack type, with the following extracted features from ARFFfile

$X_1 = 0$;

$X_2 = 199$

$Y_1 = 0$

$Y_2 = 1$,

$V_i = 255$,

U_i: is the number of compromised systems being computed.

Therefore,

$$Ui = \frac{255 - 0}{199 - 0}(1 - 0) + 0 = 1.30$$

That is, the number of systems compromised and added to the botnet family, within the two seconds DOS attack window is one.

Using ipsweep, a probe attack type, the number of compromised systems include:

$X_1 = 18$;

$X_2 = 1$,

$Y_1 = 0$,

$Y_2 = 1$,

$V_i = 1,$

$$Ui = \frac{1 - 18}{1 - 18}(1 - 0) + 0 = \frac{-17}{-17} = +1$$

Since the result is one, then the compromised systems are within the range

To compute the number of local systems compromised by a remote attacker (R2L), warezclient attack type was used, with the following features from ARFF file.

$X_1 = 334;$

$X_2 = 2,$

$Y_1 = 0,$

$Y_2 = 1,$

$V_i = 4$

$$Ui = \frac{4 - 334}{2 - 334}(1 - 0) + 0 = \frac{-330}{-332} = 0.99$$

The compromised system in this case is 0.99, which is within the range.

To rank and properly generalize the model, the entire dataset needs to be cleaned, filtered and the dimensions reduced

2.10.3 Dimensionality Reduction

Dimensionality refers to the number of features in a dataset. When the number of features in a dataset is very large relative to the number of observations, some algorithms struggle to train effective models. This is called *curse of dimensionality.*

Current trends have it that high volumes of data are generated, and can contain noise and redundant data. These irrelevant data need to be removed for effective learning of the patterns. The process is called dimensionality reduction. It is a process of reducing the number of random variables under consideration, and obtaining a set of relevant variables. Dimensionality reduction is divided into feature selection and feature extraction (Hira and Gillies, 2015):

i. **Feature Selection**: In this type of dimensionality reduction, a subset of the original features is found that can be used to model the problem at

hand. There are three ways to achieve the subset: by filtering, by wrapping, and by embedding.

- **A filter:** A Filter selects features based on their relationship with the target. Filters can be active or passive and constitute four main types: low-pass, high-pass, notch and band-reject.
- **Wrapper methods** are based on greedy search algorithm as they evaluate all combinations of the features and select the combination that produces the best result for the specific ML algorithm.
- **In Embedded technique,** malicious code like virus is infiltrated into a legitimate code; and executes each time the legitimate app is executed (Jiang *et al.*, 2020).

  Feature selection filters irrelevant, redundant and noisy data from the original data set. Some algorithms used for the purpose include decision trees (DT), random forests, genetic algorithms and principal component analysis (PCA).

ii **Feature Extraction –**Feature extraction creates new smaller set of features that still captures most of the useful information. While feature selection keeps a subset of the original dataset, feature extraction creates entirely new ones. One such example is deep learning. Feature extraction can be supervised such as linear dimensional analysis (LDA) or unsupervised (PCA).

2.10.3.1 Principal Component Analysis (PCA)

Principal component analysis is a dimension reduction tool, which reduces a large set of variables to a small set without distorting information flow in the large dataset

(Wallisch, 2014). It is a mathematical procedure that transforms a number of correlated variables into a smaller number of uncorrelated (orthogonal) variables called principal components. It increases interpretability, but at the same time minimize information loss and successively maximize variance (Wallisch, 2014).

The first principal component accounts for as much of the variability in the dataset as possible, and each succeeding principal component accounts for as much of the remaining variability as possible, and so on. Variability is achieved using Eigen vectors and Eigen values. An Eigen vector is a unit vector pointing in the direction of the new coordinate axis, and the axis with the highest Eigen value explains the most variation (Wallisch, 2014). Traditionally, PCA is performed on a square symmetric matrix, which can be covariance matrix or singular value decomposition (SVD) or correlation matrix. Correlation matrix is used if the variance of the individual variant differs much or if the individual variants differ.

The Goals of PCA.

The goals of CPA include:

i. Finding the relationships between observations
ii. Extracting the most important information from data
iii. To detect outliers and remove them
iv. To reduce the dimension of the data by keeping only relevant information.
v. It ensures that original variables have the highest correlation (relationship) with the principal components.

These goals are achieved by finding the PCA space, which represents the maximum variance of a given data (Jolliffe and Cadima, 2016; Holland, 2019). To appreciate the works of PCA, certain terms used in its extraction process are defined.

Definition of PCA Relevant Terms:

The Mean

The mean of a data set helps us to ascertain the spread of a given data.

Let X = [1, 2, 4, 6, 12, 15, 25, 45, 65, 67, 98]

The mean of X is $\tilde{x}$ and it is represented in Equation 2.18.

$$\bar{x} = \sum_{i=1}^{n} \frac{xi}{n}, \forall i = 1, 2, 3, \ldots, n \qquad \text{Equation 2.18}$$

In the above sample X,

$$\sum_{i=1}^{n} X = 1 + 2 + 4 + 6 + 12 + 15 + 25 + 45 + 65 + 67 + 98 = 340$$

$$\bar{x} = \frac{340}{11} = 30.9 = 31.$$

Standard Deviation (σ)

The standard deviation, (σ), of a dataset also measures the spread of the dataset. It is the average distance of the mean of a dataset to a point. It is represented or computed using Equation 2.19.

$$\sigma = \sqrt{\sum_{i=1}^{n} \frac{(xi - \bar{x})2}{(n - 1)}} \qquad \text{Equation 2.19}$$

Division by n-1 is due to the fact that samples are used rather than the entire population.

As an illustration, let two samples X and Y be:

X = [0, 8, 12, 20] and Y = [8, 9, 11, 12] as illustrated in Table 2.11

Table 2.11: Parameters to Compute Standard Deviation (σ)

X	$X - \bar{x}$	$(X - \bar{x})^2$
0	-10	100
8	-2	4
12	2	4
20	10	100
Total: **40**		**208**

Mean $(\bar{x}) = \frac{40}{4} = 10$

$$\sigma = \sqrt{\left(\frac{208}{3}\right)} = \sqrt{69.33} = 8.33$$

The computation of $\bar{y}$ and **σ** in Y = [8, 9, 11, 12] is illustrated in Table 2.12

Table 2.12: Illustrating the Spread of Data

Y	$Y - \bar{y}$	$(Y - \bar{y})^2$
8	-2	4
9	-1	1
11	1	1
12	2	4
Total: 40		10

Mean $(\bar{y}) = \frac{40}{4} = 10$

$$\sigma = \sqrt{\sum_{i=1}^{n} \frac{(yi - \bar{y})2}{(n-1)}} = \sqrt{(\frac{10}{3})} = \sqrt{3.33} = 1.82$$

Observe that the spread of the set X from the mean is wider than that of the set Y as its σ_x is 8.3, whereas that of Y, σ_y, is 1.82. That is, the difference between the mean and the value of X or Y is wider in X than that of Y.

Variance (σ^2)

The variance is another measure of the spread of data from the mean. The formula is illustrated in Equation 2.20

$$\text{Var (x)} = \sigma^2 = \sum_{i=1}^{n} \frac{(xi - \bar{x})2}{(n-1)} \qquad \text{Equation 2.20}$$

Covariance (cov (x, y))

Covariance is always measured between two dimensional spaces, unlike standard deviation and variance which operate on single dimensional space. If you have a dataset of 3-dimension in R^3, such as (x, y, z), you can measure the covariance of x and y, or x and z or y and z. The covariance of a dimension and itself results in the variance of that dimension. That is, cov (x, x), cov (y, y), and cov (z, z) result in the variance of x, y, and z respectively. Equation 2.21 can be rewritten as

$$\sigma^2 = \sum_{i=1}^{n} \frac{(xi - \bar{x})(xi - \bar{x})}{(n-1)} \qquad \textbf{Equation 2.21}$$

The formula for covariance is similar to that of variance, except that it deals with two dimensions at a time, rather than one dimension used by variance and standard deviation. It is illustrated in Equation 2.23.

$$\textbf{Cov (x, y) =} \sum_{i=1}^{n} \frac{(xi - \bar{x})(yi - \bar{y})}{(n-1)} \qquad \text{Equation 2.23}$$

What is more important in Equation 2.23 is whether or not the value of covariance is positive (+ve) or negative (-ve). If the value is positive, then the two dimensions increase together; but if it is negative, then as one-dimension increases, the other is decreasing. However, if the covariance value is zero, then the two dimensions are independent of each other. Since covariance is measured between any two dimensions, for n-dimensional space, covariance values are computed using Equation 2.24.

$$C^{nxn} = \frac{n!}{(n-2)! * 2} \qquad \text{Equation 2.24}$$

A useful way to compute all the possible covariance values between all dimensions is to use matrices. The definition of covariance matrix is illustrated in Equation 2.25.

$$C^{n \times n} = (C_{ij}, C_{ij}) = \text{cov}(\text{Dim}_i, \text{Dim}_j) \qquad \text{Equation 2.25}$$

Where:

$C^{n \times n}$: is a matrix with n rows and n columns;

$\text{Dim}_{i,j}$: is the ith and jth dimensions

For example, 3-dimension in R^3, such as (x, y, z) is illustrated in Figure 2.21

$$C_{x,y,z} = \begin{pmatrix} \text{Cov}(x,x) & \text{cov}(x,y) & \text{cov}(x,z) \\ \text{Cov}(y,x) & \text{cov}(y,y) & \text{cov}(y,z) \\ \text{Cov}(z,x) & \text{cov}(z,y) & \text{cov}(z,z) \end{pmatrix}$$

Figure 2.21: Covariance Matrix

The entry in row 2, column 3 in Figure 2.22 is the covariance value computed between the 2nd and 3rd dimensions. We have already seen that the covariance of a dimension with itself produces the variance of that dimension. - cov (x, x); cov (y, y) and cov (z, z) are variances of x, y, and z respectively. Since covariance matrix is commutative, ie, cov (a, b) = cov (b, a), then the matrix is symmetric about the main diagonal. That is, the diagonal values of the covariance matrix represent the variance of the variables – x, y and z.

Matrices

A matrix is a rectangular array of numbers. The numbers there in are called entries. Most of the time, matrices are bordered by square brackets, [ ]. The size of the matrix is described in terms of rows and columns. For example, in a 3 x 3 matrix, such as

$$A = \begin{pmatrix} e & x & -\sqrt{2} \\ 0 & ½ & 1 \\ 0 & 0 & 0 \end{pmatrix},$$

there are three rows and three columns**.** Other types of matrices include:

Row vector, for example, a 1 x 4 vector is [2 1 0 -1], which has one row and four columns.

A column vector, say a 2 x 1 matrix has two rows and one column, $\begin{bmatrix} 1 \\ 3 \end{bmatrix}$

A unit vector, has both one row and one column, 1 x 1 matrix, [4]

A 3 x 2 matrix has three rows and two columns, $\begin{pmatrix} 1 & 2 \\ 3 & 0 \\ 1 & 4 \end{pmatrix}$

(Anton and Rorres, 2014).

Capital letters are used to denote matrices, while lower case letters denote numerical quantities. For example,

$$A = \begin{pmatrix} 2 & 1 & 7 \\ 3 & 4 & 2 \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix}$$

In matrix, numerical quantities are called scalars. Let **A** be the matrix; and in **A**, a_{11}, a_{12} … a_{34} are numerical quantities.

$$A = \begin{pmatrix} a11 & a12 & a13 & a14 \\ a21 & a22 & a23 & a24 \\ a31 & a32 & a33 & a34 \end{pmatrix}, \text{ A is a 3 x 4 matrix}$$

Transpose of a matrix

The transpose of a matrix **A,** denoted by $\mathbf{A^T}$, is the matrix A with rows and columns interchanged. That is, the first column of transpose $\mathbf{A^T}$ is the first row of **A;** the second column of transpose $\mathbf{A^T}$ is the second row of **A,** and so on. Transpose $\mathbf{A^T}$ is represented below.

$$A^T = \begin{pmatrix} a11 & a21 & a31 \\ a12 & a22 & a32 \\ a13 & a23 & a33 \\ a14 & a24 & a34 \end{pmatrix}, \ A^T \text{ is a 4 x 3 matrix}$$

Identity matrix

A square matrix with 1s on the main diagonal and zeros elsewhere is called an identity matrix.

$$\mathbf{I} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The identity matrix is denoted by **I**. If **A** is any m x n matrix, then the product of **A** and the identity matrix, **I,** is shown in Equation 2.26.

$$\mathbf{AI = A} \text{ and } \mathbf{IA = A} \qquad \text{Equation 2.26}$$

Inverse of a matrix

If **A** is a square matrix, and if a matrix **B** of the same size can be found such that

$$\mathbf{AB = BA = I,}$$

then **A** is invertible (or nonsingular) and **B** is called an inverse of **A.** If **B** cannot be found, then **A** is said to be singular.

An invertible matrix

An invertible matrix (non-zero matrix) has exactly one inverse, and the product of **A** and its inverse, $\mathbf{A^{-1}}$ results in an identity matrix, **I,** as shown in Equation 2.27.

$$\mathbf{AA^{-1} = A^{-1}A = I} \qquad \text{Equation 2.27}$$

Matrix A is invertible iff the product of the entries on the main diagonal minus the product of the entries off the main diagonal is not equal to zero. That is, ad - bc $\neq 0$.

The inverse of a matrix **A** is given by Equation 2.28

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\mathbf{A^{-1}} = \begin{bmatrix} a & -b \\ -c & d \end{bmatrix} \qquad \text{Equation 2.28}$$

To multiply two matrices together, the number of rows of one matrix must be equal to the number of columns of the other matrix. That is, if **A** is an m x r matrix and **B** is an r x n matrix, then **AB =BA = m x n.** The product entry is determined by multiplying each element of a row in matrix **A** by a corresponding element in the column of matrix **B** and add them up. For example, multiply a 2 x 3 matrix by a 3 x 4 matrix.

$$A = \begin{pmatrix} 1 & 2 & 4 \\ 2 & 6 & 0 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 4 & 1 & 4 & 3 \\ 0 & -1 & 3 & 1 \\ 2 & 7 & 5 & 2 \end{pmatrix}$$

$$AB = \begin{pmatrix} 1 & 2 & 4 \\ 2 & 6 & 0 \end{pmatrix} \begin{pmatrix} 4 & 1 & 4 & 3 \\ 0 & -1 & 3 & 1 \\ 2 & 7 & 5 & 2 \end{pmatrix}$$

Matrix multiplication requires that the number of columns of **A** be equal to the number of rows of **B;** however, multiplying a square matrix with a vector results in another vector. For example, a square matrix **D** and a column vector **V**, when multiplied produces another vector.

$$\mathbf{D} = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$$

$$\mathbf{V} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

$$\mathbf{DV} = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 12 \\ 8 \end{bmatrix} = 4 \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

As can be seen, the resulting vector is 4 times the original vector. Therefore, the vector

$\begin{bmatrix} 3 \\ 2 \end{bmatrix}$ is an Eigen vector, and the number 4 is the Eigen value

Determinant

The determinant of a 2 x 2 matrix **A**, is the product of the entries of the main diagonal minus the product of the entries off the main diagonal, as illustrated in Equation 2.29.

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\mathbf{det\ (A) = ad - bc} \qquad \text{Equation 2.29}$$

Eigen vectors and Eigen values

Eigen vectors and Eigen values are used to determine the principal components in PCA process and filter out the less important variables in training dataset. Let **A** be n x n matrix and let $\mathbf{x} \in \mathbb{R}^n$ (where n represents the level of dimensions) be a non-zero vector,

if the product of the matrix and the vector is equal to the product of a scalar, λ**,** and the vector as illustrated in Equation 2.30,

$$\mathbf{Ax} = \lambda\mathbf{x} \qquad \text{Equation 2.30}$$

Then λ is called an Eigen value of the matrix **A**; and **x** is called an Eigen vector of **A** associated with λ**.** The set of all Eigen values of an n x n matrix **A** is denoted by $\sigma(A)$ referred to as the spectrum of **A**. When an Eigen vector **X** is multiplied by **A**, the resulting vector is in the same direction as **X** or opposite to it. From Equation 2.31.

$$\mathbf{Ax} = \lambda\mathbf{x}$$

$$\mathbf{Ax} - \lambda\mathbf{x} = \mathbf{0} \qquad \text{Equation 2.31}$$

To ensure that we are dealing with matrices, we multiply λ**X** by **I.** As illustrated in Equation 2.32. That is,

$$\mathbf{Ax} = \lambda\mathbf{Ix}$$

$$\mathbf{Ax} - \lambda\mathbf{Ix} = \mathbf{0} \qquad \text{Equation 2.32}$$

$$(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{x} = \mathbf{0}$$

Or

$$(\lambda\mathbf{I} - \mathbf{A})\,\mathbf{x} = \mathbf{0}$$

The matrix used to compute Eigen values and Eigen vectors should not be invertible but a singular matrix. Therefore, the determinant of a singular matrix is equal to zero as shown in Equation 2.33.

$$\text{Det}\,((\lambda\mathbf{I} - \mathbf{A}) = \mathbf{0} \qquad \text{Equation 2.33}$$

The expression $((\lambda\mathbf{I} - \mathbf{A})\,\mathbf{X} = \mathbf{0})$ is a polynomial called the characteristic polynomial of **A, w**hile $\mathbf{det}\,((\lambda\mathbf{I} - \mathbf{A}) = \mathbf{0})$ is called the characteristic equation**.** It therefore means that the roots of the characteristic polynomial are the Eigen values of **A** (Anton and Rorres, 2014). While Eigen vectors should not be zero, Eigen values can be zero. For example,

$$\text{Let A} = \begin{pmatrix} 0 & 5 & -10 \\ 0 & 22 & 16 \\ 0 & -9 & -2 \end{pmatrix} \quad \text{and} \quad X = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\text{Therefore AX} = \begin{pmatrix} 0 & 5 & -10 \\ 0 & 22 & 16 \\ 0 & -9 & -2 \end{pmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = 0 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

(Anton and Rorres, 2014**).**

Given the importance of Eigen values and Eigen vectors to PCA in the determination of Principal components, it becomes necessary to illustrate how they are computed.

Let $A = \begin{pmatrix} -2 & -4 & 2 \\ -2 & 1 & 2 \\ 4 & 2 & 5 \end{pmatrix}$ and let **I** be a 3 x 3 identity matrix

$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, then $\lambda \mathbf{I} = \lambda \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}$

Now

Det $\mathbf{(A - \lambda I) = 0}$

$$\text{Det}\left[\begin{pmatrix} -2 & -4 & 2 \\ -2 & 1 & 2 \\ 4 & 2 & 5 \end{pmatrix} - \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}\right] = \begin{pmatrix} -2-\lambda & -4 & 2 \\ -2 & 1-\lambda & 2 \\ 4 & 2 & 5-\lambda \end{pmatrix} = 0$$

There are many ways to solve or expand this determinant. It could be done using *reduced row echelon* form, *or row factor* form. In this example, we will use the row factor form. The resulting quadratic equations are illustrated in Equation 2.33, Equation 2.34, and Equation 2.35

$$\text{Det}\begin{pmatrix} -2-\lambda & -4 & 2 \\ -2 & 1-\lambda & 2 \\ 4 & 2 & 5-\lambda \end{pmatrix} = (-2-\lambda) \begin{bmatrix} 1-\lambda & 2 \\ 2 & 5-\lambda \end{bmatrix}$$

$$= (-2-\lambda)\,[(1-\lambda)(5-\lambda) - 2 \text{ x } 2]$$

$$= (-2-\lambda)\,(5 - \lambda - 5\lambda + \lambda^2 - 4)$$

$$= (-2-\lambda)\,(-6\lambda + \lambda^2 + 1)$$

$$= (-2-\lambda)\,(\lambda^2 - 6\lambda + 1)$$

$$= -2\lambda^2 + 12\lambda - 2 - \lambda^3 + 6\lambda^2 - \lambda$$

$$= -\lambda^3 + 11\lambda + 4\lambda^2 - 2$$

$$= -\lambda^3 + 4\lambda^2 + 11\lambda - 2 \qquad \text{Equation 2.34}$$

$$\text{Det}\begin{pmatrix} -2-\lambda & -4 & 2 \\ -2 & 1-\lambda & 2 \\ 4 & 2 & 5-\lambda \end{pmatrix} = -(-4)\begin{bmatrix} -2 & 2 \\ 4 & 5-\lambda \end{bmatrix}$$

$$= 4[(-2)(5-\lambda) - 2x4]$$

$$= 4[(-2)(5-\lambda) - 8]$$

$$= 4(-10 + 2\lambda - 8)$$

$$= 4(2\lambda - 18)$$

$$= 8\lambda - 72 \qquad \text{Equation 2.35}$$

$$\text{Det}\begin{pmatrix} -2-\lambda & -4 & 2 \\ -2 & 1-\lambda & 2 \\ 4 & 2 & 5-\lambda \end{pmatrix} = 2\begin{bmatrix} -2 & 1-\lambda \\ 4 & 2 \end{bmatrix}$$

$$= 2[(-2) \times 2 - 4(1-\lambda)]$$

$$= 2(-4 - 4 + 4\lambda)$$

$$= 2(-8 + 4\lambda)$$

$$= -16 + 8\lambda$$

$$= 8\lambda - 16 \qquad \text{Equation 2.36}$$

Summing up Equation 2.36, Equation 2.37 and Equation 238, we have Equation 2.39.

$$[-\lambda^3 + 4\lambda^2 + 11\lambda - 2] + [8\lambda - 72] + [8\lambda - 16] = 0$$

$$-\lambda^3 + 4\lambda^2 + 11\lambda - 2 + 8\lambda - 72 + 8\lambda - 16 = 0$$

$$-\lambda^3 + 4\lambda^2 + 27\lambda - 90 = 0$$

$$\lambda^3 - 4\lambda^2 - 27\lambda + 90 = 0 \qquad \text{Equation 2.37}$$

Equation 2.38 is a quadratic equation in the third degree. To solve it, we first need to find the factors of 90, which are:

$\pm 1, \pm 2, \pm 3, \pm 5, \pm 6, \pm 9, \pm 10, \pm 15, \pm 18, \pm 30, \pm 45$, and ± 90

We need to find a factor of 90 that when plugged into Equation 2.92 will result in zero (0). By trial and error, $3^3 - 4 \times 3^2 - 27 \times 3 + 90 = 0$

$= 27 – 36 – 81 + 90 = 0$

Therefore $(\lambda$ **- 3)** is a factor of the quadratic equation. Hence

$$\lambda^3 - 4\lambda^2 - 27\lambda + 90 = 0$$

$$(\lambda\text{-}3)\ (\lambda^2 - \lambda - 30) \qquad \text{Equation 2.38}$$

From the second-degree quadratic equation, $(\lambda^2 - \lambda - 30)$, the factors are numbers that when multiplied together gives you -30, and when added together gives you the coefficient of λ. That is, 5 x (-6) = -30

And 5 + (-6) = -1. Therefore, from observation, the factors are $(\lambda +5)\ (\lambda - 6) = 0$

From Equation 2.84, we have

$$(\lambda - 3)\ (\lambda +5)\ (\lambda - 6) = 0$$

$(\lambda - 3) = 0$; hence $\lambda = 3$

$(\lambda +5) = 0$; hence $\lambda = -5$

$(\lambda - 6) = 0$; hence $\lambda = 6$.

Therefore, the spectrum of matrix A, $\sigma(A)$, (the set of Eigen values of the matrix A) is

$\sigma\ (A) = 3\ -5 + 6. = 4$

Now, for each Eigen value, there is a corresponding (associated) Eigen vector. We now need to find the Eigen vectors. Let the Eigen vector V be:

$$V = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

For $\lambda = 3$, and from the characteristic polynomial $(A - \lambda I)V = 0$, we have

$$\begin{pmatrix} -2-\lambda & -4 & 2 \\ -2 & 1-\lambda & 2 \\ 4 & 2 & 5-\lambda \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Substituting 3 for λ and subtract we have

$$\begin{pmatrix} -5 & -4 & 2 \\ -2 & -2 & 2 \\ 4 & 2 & 2 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Let X = 1, then the first two equations are Equation 2.39 and Equation 2.40

$$-5 - 4Y + 2Z = 0 \qquad \text{Equation 2.39}$$

$$-2 - 2Y + 2Z = 0 \qquad \text{Equation 2.40}$$

Subtracting Equation 2.40 from Equation 2.41, we have Equation 2.42

$$3 + 2Y = 0$$

$$Y = \frac{-3}{2} \qquad \text{Equation 2.41}$$

Substitute the value of Y in Equation 2.42 and solve for Z as illustrated in Equation 2.43

$$-2 - 2Y + 2Z = 0$$

$$-2 - 2\left(\frac{-3}{2}\right) + 2Z = 0$$

$$-2 + 3 + 2Z = 0$$

$$1 + 2Z = 0$$

$$2Z = -1$$

$$Z = -\tfrac{1}{2} \qquad \text{Equation 2.42}$$

Check for correctness by substituting the values of X, Y, and Z in Equation 2.42

Therefore, the Eigen vector, V, for $\lambda = 3$ is

$$V_3 = \begin{bmatrix} 1 \\ \frac{-3}{2} \\ -\frac{1}{2} \end{bmatrix}, \text{ and scaling up V by 2 produces } \begin{bmatrix} 2 \\ -3 \\ -1 \end{bmatrix}$$

Following the same technique, the Eigen vectors for $\lambda = -5$ and $\lambda = 6$ are

$$V_{-5} = \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix} \text{ and } V_6 = \begin{bmatrix} 1 \\ 6 \\ 16 \end{bmatrix} \text{ respectively.}$$

The Eigen vector with the highest Eigen value produces the first principal component; and the Eigen vector with the second highest Eigen value produces the second principal component. Therefore, from our example,

Principal Component 1 (PC 1) has $\lambda = 6$ and

$$V_6 = \begin{bmatrix} 1 \\ 6 \\ 16 \end{bmatrix}$$

Principal Component II (PC II) has $\lambda = 3$ and

$$V_3 = \begin{bmatrix} 2 \\ -3 \\ -1 \end{bmatrix}$$

Principal Component III (PC III) has $\lambda = -5$ and it is rejected.

$$V_{-5} = \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}$$

From the foregoing, V_6 and V_3 are selected, while V_{-5} is rejected. Normally, in situations where there are many vectors, a threshold is set, say >= 70%; and Eigen values that form 70% or more of the spectrum determine the selection of principal components of the affected vectors. Therefore, the selection is in line with the threshold.

The spectrum of **A** is (3 – 5 + 6) = 4;

$\frac{3}{4} * 100 = 75\%$, and

$\frac{6}{4} * 100 = 150\%$

$\frac{-5}{4} * 100 = -125\%$

Algorithm to Compute Eigenvectors and Eigen Values

i. From the characteristic polynomial, form the characteristic equation

$$(A - \lambda I) X = 0$$

$$\text{Det}(A - \lambda I) = 0$$

OR

$$\text{Det}(\lambda I - A) = 0$$, this form is used in most literature

Expand the determinant using row factor form (or reduced row echelon form)

ii. Solve the resulting quadratic equation to get the Eigen values of the given singular matrix A (The matrix should not be an invertible matrix)

iii. Find the corresponding Eigen vectors to the Eigen values.

iv. Sort Eigen vectors according to their corresponding Eigen values.

v. Select the Eigen vectors that have the highest Eigen values

vi. The selected Eigen vectors represent the projection of PCA.

(Jolliffe and Cadima, 2016; Holland, 2019).

PCA process is illustrated in the algorithm that follows.

Algorithm to Compute PCA Using Covariance Matrix Method

i. Get some data, say two or three dimensions

ii. Compute the mean

iii. Subtract the mean from the data points of each dimension

iv. Calculate the covariance between two or more dimensions

v. From the covariance matrix, find the Eigen vectors and Eigen values

vi. Sort Eigen vectors according to their corresponding Eigen values.

vii. Select the Eigen vectors that have the highest Eigen values

viii. The selected Eigen vectors represent the projection of PCA.

(Jolliffe and Cadima, 2016; Holland, 2019).

Eigen vectors are usually unit matrices, they are extensible; and produce the same number of Eigen values, which are multiples of the Eigen vectors.

The pre-processed data is then used by Random Forest algorithms to train base models such as SVM, KNN, DT, NB that will develop models to analyze, detect.

2.11 Training the Algorithm

In training, input is mapped to the output, over a sequence of pairs $\{(x_1,y_1), (x_2,y_2), \ldots, (x_t, y_t)\}$ and each connection has a weight (w_t) associated with it. Each x_t is an example of feature vector; $yt \in \{-1, +1\}$ is its label and wt is the weight vector. The weight describes the likelihood that the learning patterns will reflect the actual relationship in the data. At each step t during training, the algorithm makes a label prediction $h_t(x_t)$, which for linear classification is $h_t(x) = \text{sign}(w_t.x)$. After making the prediction, the

algorithm then compares the value with the label yt, if $h_t(x_t) \neq y_{t,}$ an error is recorded for time t. The algorithm maintains different confidence metrics for each feature so that less confident weights (misclassified weights) are updated more aggressively than more confident weights.

However, if there is no difference between the predicted label and the actual label $h_t(x_t)$, then learning for this data feature is complete, and there is no need to continue the training; otherwise the training algorithm updates the connection weights with the aim of reducing their differences until the expected label is achieved (Dewa and Maglaras, 2016; Nyuyen *et al.,* 2017; Dong *et al.,* 2020; Shi *et al.,* 2020; AWS, 2021).

2.11.1 Some Normal Characteristics Learned by Base Classifiers

In dynamic analysis, the base models are trained to learn the features of normal applications. This knowledge is used to assess applications on the net or host systems, and any deviation from the normal characteristics attracts the application being flanked as abnormal (malware). Some of these characteristics are inherent in NSL-KDD dataset. For instance:

i. Categorical features in ARFF file:

   (a) Protocol types used by normal applications are transmission control protocol (tcp) and user datagram protocol (udp); while malware tend to use more of internet control message protocol (icmp). Icmp not only report errors in data transmission, and diagnosis of host system performance through traceroute and ping; but it is also used to execute distributed denial of service (DDOS) attack type in the form of icmp

flood, smurf attack type with spoofed (faked) IP address, and ping_of_death attack type. Unlike tcp and udp, icmp does not require that devices must connect via three-way handshake before messages are sent; nor does it require specific port to send its messages.

(b) Services used by normal applications include http, private, smtp, ftp_data, other, telnet, and domain_u; while malware tend to use the following services private, mtp, finger, netbios_path, z39_50, and cs_net_ns. If any of these services is used in conjunction with tcp or udp and SF flag, then a malware is detected.

(c) Flag used by normal applications is safe_flag (SF); while malware tend to use S0 and REJ flags (very common with Neptune – DOS attack).

ii. POLP – Principle of least privileges (Lord, 2020). It states that "persons, applications, or processes should be given the bare minimum privileges (resources) to complete a given task". Normal applications obey POLP principle, while abnormal applications tend to request for more than necessary resources for their tasks.

iii. Android install_time permissions**:**

(a) Normal applications declare permissions from Android_manifest.xml file

(b) Malware declare permissions that include hardware such as GPS, GPRS, Camera, and external and undocumented APIs (Application Programming Interfaces, usually from third party stores)

iv Central process unit (CPU): Normal applications use CPU to process data, but malware tend to use graphics processing unit (GPU) to process data and evade detection.

v. Dangerous permissions: These permission types, which are 24 in number out of 135 approved permissions, are used by malware. They tend to violate user privacy (Raymond *et al.,* 2020; Sun *et al.,* 2016).

vi. Over privileged permissions: These permissions are found in manifest.xml file but they are not used at run_time. These types of permissions are requested for from the innocent user by malware in defiance to POLP principle for their nefarious intents.

vii. Business Transactions – Malware applications tend to eavesdrop and intercept (hijack) business transactions online; while normal applications do not.

viii. Third Party Applications - These applications are developed by other companies that clone applications (70% of these applications are malware) (Atkinson, 2015). While Google patches the vulnerabilities in normal applications regularly, third parties are reluctant to effect the patches in their versions, hence they continue to send older versions with vulnerabilities.to Google Play Store.

2.12 Ensemble Learning Methods

Ensemble learning combines several base classifiers to form one optimized predictive classifier (Pooja and Pushpalatha, 2019; Mary and Kumar, 2020; Brownlee, 2021). It can be used to decrease variance (bagging), decrease bias (boosting), and improve prediction (stacking). It is also divided into two groups:

i. Sequential learning, with different models trained sequentially (independently), and the mistakes of previous models are learned by their successors. Thus, giving the mislabeled classifiers higher weights. A good example is boosting technique.

ii. Parallel learning, where base models are trained in parallel, as in bagging technique, and their results aggregated into a stronger classifier using voting scheme (majority voting or plurality voting or hard voting).

2.12.1 Bagging (Bootstrap and Aggregation)

Bagging, also called bootstrap aggregation, seeks a diverse group of ensemble members by varying the training dataset. It is classified into bootstrapping and aggregation. Bootstrapping is a sampling technique where samples are derived from the whole population with replacement. Replacement means that if a row (record) is selected, it is returned to the training dataset for potential reselection. That is, a record may be selected zero, one, or more times for a given training dataset (Brownlee, 2021; Singh, 2018). Bootstrapping involves using a single machine learning algorithm, typically an unpruned decision tree, and training each model with a different sample of the same training dataset. Bagging reduces variance and over fitting. Example bagging algorithms include

i. Bagging decision tree
ii. Random Forest
iii. Extra trees

The predictions made by ensemble members are combined or aggregated using voting classification techniques.

Algorithm of Bagging Process

The following are the steps used to perform bagging process

i. Multiple subsets are created from the original dataset, selecting the observations (records) with replacement.

ii. A base learner (weak model) is trained on each of the data subsets

iii. The models run in parallel and are independent of each other.

iv. The final prediction is determined by combining the predictions from all the trained models using a voting process.

Figure 2.22 depicts the bagging process.

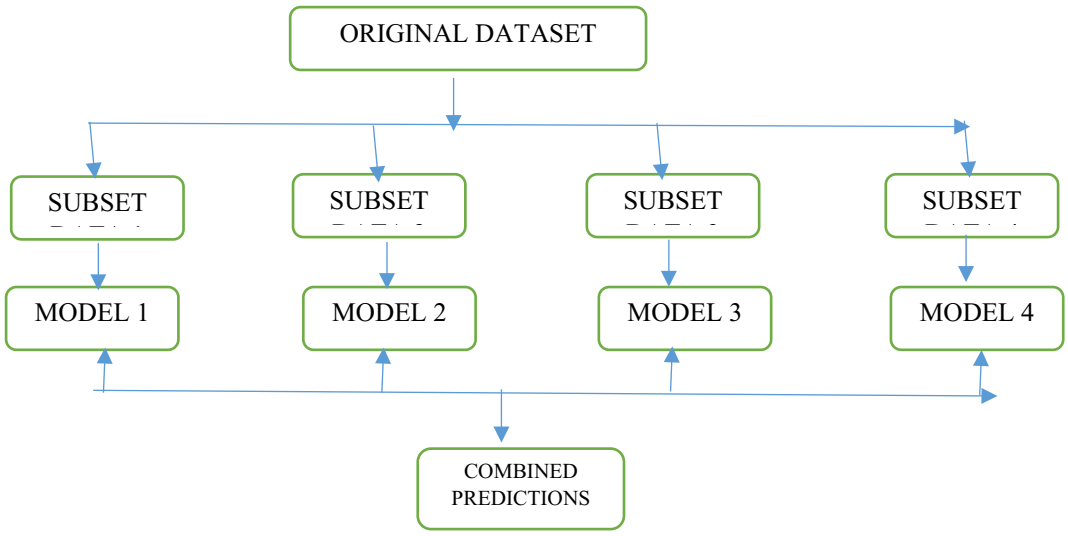


Figure 2.22: Architecture of Bagging Process
Source: Singh (2018)

2.12.2 Boosting

Boosting algorithm tries to build a strong learning (predictive) model from the mistakes of several base models. A base model is one that performs better than random guessing, but still performs poorly at assigning classes to objects. Boosting can be either binary with two labeled classes {-1, +1} (malicious or benign) respectively or multi-class.

It starts by assigning weights to the training dataset in memory, chooses and trains appropriate classifiers sequentially, focusing on misclassified results by previous models. It updates misclassified labels of previous base model. Then a second model is

trained using the updated dataset, thus reducing the errors made by the previous one. Boosting reduces bias error, which arises when models are unable to identify relevant trends in the data. Types of boosting include:

i. AdaBoost (Adaptive Boosting)
ii. Gradient Tree Boosting
iii. XGBoosting (Extreme Gradient Boosting)

Using Adaboost as boosting algorithm, with KNN, SVM, DT, and LR as base models a more formidable model is formulated by harnessing the predictions of the base models through a voting process (hard vote). The hard voting method uses the predicted labels of the classifiers and a majority rule system.

Adaboost (Freund and Schapire, 1996)

Adaboost, also called Adaptive Boosting, is a boosting method that improves the predictive capacity of base classifiers. It can combine rough or moderately accurate results to produce a more accurate result. It can function as a binary classifier, where only two classes are considered within a range {-1, 1}; and as a multi-class method, where more than two possible classes are considered. Boosting tends to solve two main problems:

i. How to adjust the training set to suit the base classifier (feature selection)
ii. How to combine the base classifiers into a strong classifier.(voting classification)

Algorithm to carry out a boosting process

The following steps are followed to carry out a boosting process

i. A subset is randomly selected from the original set
ii. Initially, all data points are given equal weights
iii. A base model is trained with this subset

iv. This model is used to make predictions on the entire dataset.

v. Errors are calculated using the actual values and predicted values.

vi. Observations which are incorrectly predicted are given higher weights.

vii. Another model is trained using the updated dataset, and predictions are made on the dataset.

viii. Similarly, multiple models are created (trained), each correcting the errors of the previous model before it.

ix. The final model (strong learner) is the weighted mean of all the weak (base) learners.

Therefore, boosting algorithm combines the weak learners to form the strong learner. Figure 2.23 depicts the architecture of boosting algorithm.

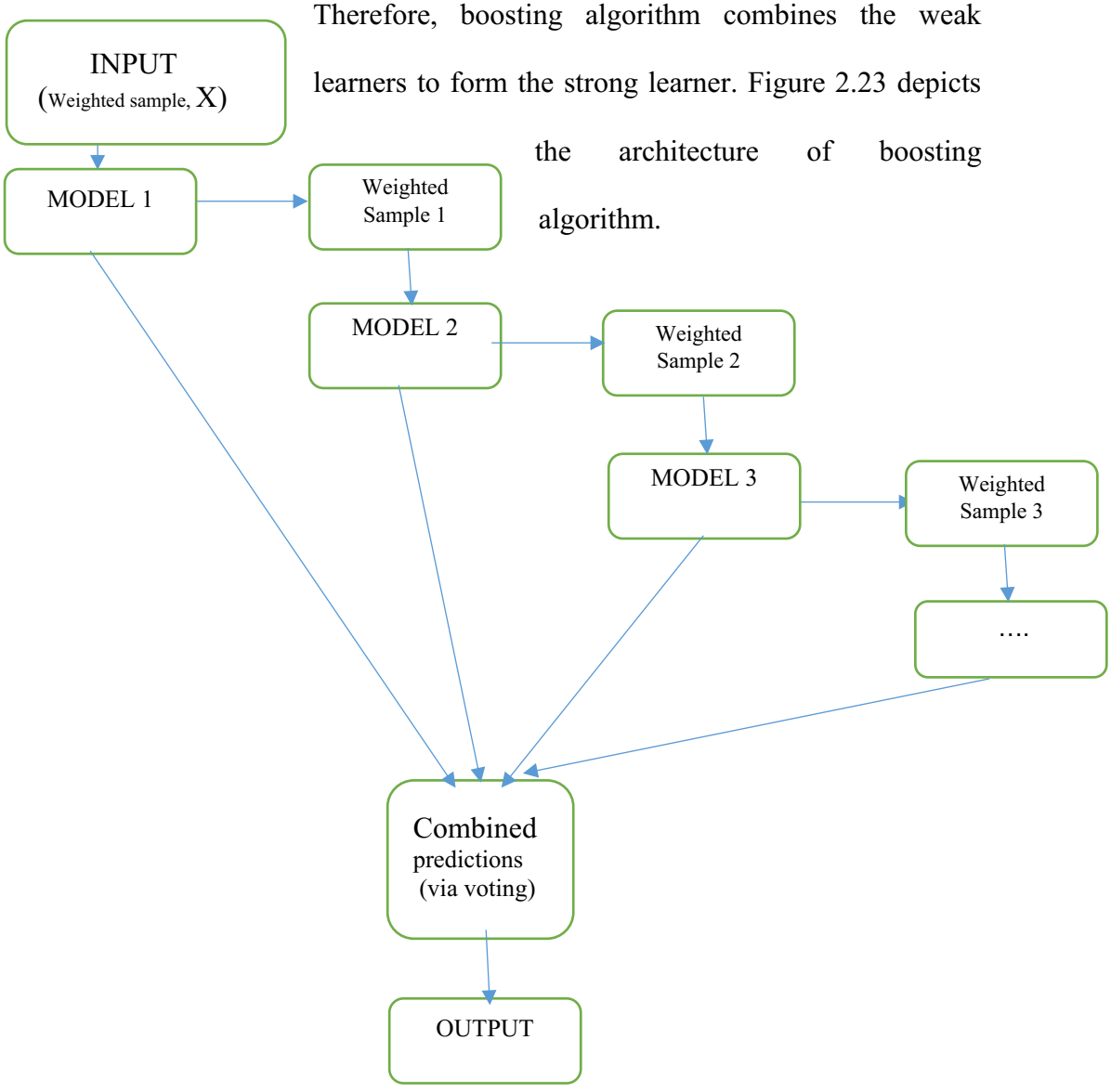


Figure 2.23: Architecture of Boosting Algorithm

Singh, 2018.

2.12.3 Stacking Ensemble Learning (Stacked Generalization)

Stacking seeks for a diverse group of base algorithms by varying the model types to fit the training dataset, and using another model to combine the predictions. In stacking, a learner is trained to combine individual learners. The individual learners are called zero-level learners; while the combiner is called first-level learner or meta-learner. In stacking, ensemble members are referred to as level-zero models, while the combiner is referred to as level-1 model. Examples of stacking process include stacked model, blending and super ensemble. Figure 2.24 depicts the architecture of Stacking process.

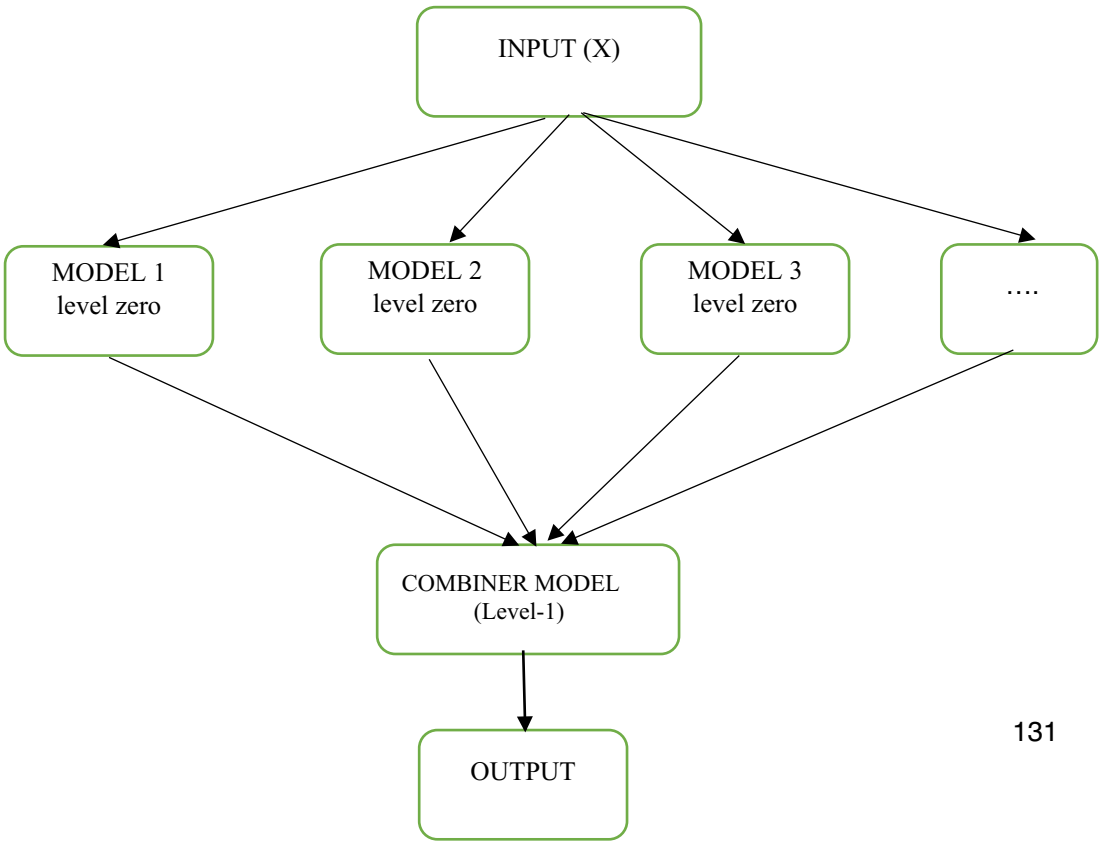


Figure 2.24: Architecture of Stacking Process.
Source: Singh, 2018

2.12.4 Voting Techniques in Ensemble Learning

In ensemble learning, different machine learning methods, working independently of each other, are moderated by the training algorithm such that they predict the class the target object is to be assigned. The training algorithm then uses a voting technique to determine the final prediction. Voting involves each model that made a prediction to assign a vote to the class that was predicted. The votes are then tallied and an outcome is chosen using the tallies. In classification, there are four types of voting techniques (Brownlee, 2021):

i. Plurality voting technique
ii. Majority voting technique
iii. Unanimous voting technique
iv. Weighted voting technique

Plurality voting selects the class label with most votes. When a tie occurs, the votes are sorted and the first one is taken for the expected prediction, instead of taking a random selection. Majority voting (Hard vote) selects the class that has more than half the votes, where no such class exists, then no prediction is made. That is, in hard voting, every individual classifier votes for a class, and the majority wins. Majority voting is best for

independent models' outputs (that is, the results of the independent predictions are diverse).

Weighted voting weighs the predictions made by each model, e.g. based on the average performance of the model, such as classification accuracy. Assigning weights to classifiers can involve using an optimization algorithm and a holdout dataset, a linear model or any machine learning model. Performance evaluation metrics can also be used to assign weight such as accuracy, precision, recall and f-measure. Weighting is done because some classifiers are more accurate than others, as such get a larger share of the vote (Brownlee, 2021).

In soft voting, every individual classifier provides a probability value that a specific data point belongs to a particular target class.

2.13 Performance Evaluation

2.13.1 Cross Validation in Soft Computing Techniques.

Cross validation is a technique used to assess how the results of statistical analysis generalize to a test dataset. That is, it provides an insight on how the trained or learned model adapts to an unseen dataset. A round of cross validation comprises the partitioning of data into complementary subsets, and then analysis is performed on all but one subset. After this, the analysis is validated on the one subset (testing set). To reduce variability, many rounds of cross validation are performed using many different partitions and then an average of the results is taken. Types of cross validation include (Berrar, 2018):

Holdout Method of Cross Validation

In this method, a part of the training data is removed and used to get predictions from the model trained on the rest of the data. The error estimation (bias) tells how the model is doing with the unseen data or the validation set. However, this method suffers from

high variance, because it is not certain which data might end up in the validation set and the results might be entirely different for different sets.

K-fold Crosses Validation

There is never enough data to train a model, therefore removing a part of it for validation poses a problem of under fitting. By reducing the training data, we risk losing important pattern/trends in the dataset, which in turn increases error induced by bias. Therefore, in K-fold cross validation, the data is divided into k subsets. The holdout method is then repeated k times, so that each time, one of the k subsets is used as the test/validation data set and the other K-1 subsets are used as the training set. The error estimation is averaged over all k trials to get the total effectiveness of the model trained. This gives a bias-variance tradeoff. That is, the process reduces bias as we are using most of the data for training, and also reduces variance as most of the data is also being used in the validation set (testing set). Generally, K can be equal to 5 or 10; but it can take any value including K=1- called leave out 1 cross validation. The data used to train and test models is stored in the desktop, laptop or mobile device from where it is ported to the cloud using mobile host agent for analysis and detection.

2.13.2 Standard Evaluation Metrics

The task of algorithms is to search for patterns in training dataset and construct mathematical models to learn those patterns. These models are then evaluated on the basis of their predictive capacities. The evaluation is either

i. 10% split cross-validation
ii. 33% split cross-validation
iii. K-fold cross validation

With 10% split cross-validation, 10% of data is used for testing, while the remaining 90% is used for training the model. For 33% split cross-validation, 33% of the data is

used for testing purpose, and the remaining 67% data set is used for training the model. The K-fold cross validation applies the classifier on the data k times, and each time with an 80:20 ratio. That is, 80% data for training and 20% data for testing. The final model is the average of the k-iterations.

Whereas, split cross-validation takes shorter time comparatively to evaluate results, it has over-fitting drawback, especially the 33% cross-validation evaluation. This drawback occurs when the classifier memorizes the training set features instead of getting trained. Because of over-fitting problem, k-fold validation produces much better results. In order to evaluate the performance of the classifiers, the following standard metrics are computed:

i. **True Positive Rate (TPR)**, which is the proportion of the correctly classified instances.

ii. **False Positive Rate (FPR)**, which is the proportion of incorrectly classified instances.

iii. **Precision**: This is the number of true positives divided by the total number of elements labeled as belonging to the positive class.

iv. **Area under the curve (AUC),** which provides the relationship between false negative and false positive is illustrated in Equation 2.43

$$\text{AUC} = \tfrac{1}{2}\left(\frac{TP}{TP+TN}+\frac{TN}{TN+FP}\right) \qquad \text{Equation 2.43}$$

AUC, which is a graphical representation of receiver operating curve (ROC), has some unanticipated problems in its application to IDS evaluation, which include: problems in detecting appropriate units for analysis; bias towards unrealistic detection approaches; and questionable presentation of false alarm data (Dewa and Maglaras, 2016; Azad and Jha 2014; Anthony, 2014). Confusion metrics characteristics used in measuring the

precision of IDS are computed in Equation 2.44, Equation 2.45, Equation 2.46, and Equation 2.47:

$$\text{TPR (Recall)} = \frac{TP}{TP + FN} \quad \text{Equation 2.44}$$

$$\text{FPR} = \frac{FP}{TN + FP} \quad \text{Equation 2.45}$$

$$\text{Precision} = \frac{TN}{TN + FP} \quad \text{Equation 2.46}$$

$$\text{F-Measure} = \frac{(2 \times \text{Recall} \times \text{Precision})}{(\text{Recall} + \text{Precision})} \quad \text{Equation.2.47}$$

Where:

i. True Positive (TP) is the number of normal data correctly classified
ii. True Negative (TN) is the number of malware samples correctly classified
iii. False Positive (FP) is the number of normal samples classified as malware
iv. False Negative (FN) is the number of malware samples classified as normal

Precision, also called **specificity**, returns the rate of relevant results, rather than irrelevant results. Recall is sensitivity for the most relevant results. F-Measure is the value that estimates the entire system performance by combining precision and Recall into a single number. The maximum number of 1.00 indicates the best result (Narudin *et al.*, 2014).

Accuracy: - This is a metric that measures how correctly an IDS works, in terms of precision of detection, number of false alarm and stability; because intrusion data set is relatively lower than normal dataset, making them harder to detect than normal dataset, it results in excessive false alarm. Accuracy measures include:

i. **Sensitivity and Specificity**: These two measures (metrics) attempt to measure the accuracy of a two-class problem. When IDS classifies data, its decision is either right or wrong.

ii **Total Delay**: This is the difference between t_{attack} and $t_{response}$. The smaller the value of total delay, the better the IDS is, with respect to its response.

Where t_{attack} is the time of attack; and $t_{response}$ is the time it takes to respond after an attack.

iii **Quality of Data**: - The quality of data is influenced by several factors:

- **Source of the data**– Data should be from reliable and appropriate sources.
- **Selection of sample** – Sample data should be unbiased
- **Sample size** – The sample size should be neither over nor under size
- **Time of data** – Data should be frequently updated in real time
- **Complexity of data** – Data should be simple.

Matthew Correlation coefficient (MCC)

MCC is another form of performance evaluation technique, it is illustrated in Equation 2.48.

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)\,(TP + FN)\,(FN + FP)\,(TN + FN)}} \quad \text{Equation 2.48}$$

Kappa Score:

Kappa statistics measures the agreement of prediction with the true class. The true class has a value of 1.0. Kappa formula is illustrated in Equation 2.49.

$$Kappa = \frac{\text{Total accuracy} - \text{Random accuracy}}{1 - \text{Random accuracy}} \quad \text{Equation 2.49}$$

Mean Absolute Error (MAE)

MAE measures the average magnitude of the errors in a set of forecasts, without considering their direction. It measures accuracy for continuous variables. It also calculates the closeness between the predictions to the actual outcome. It is the average of the absolute errors, as illustrated in Equation 2.50

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |Q - Q_i| \quad \text{Equation 2.50}$$

RMSE (Root Mean Square Error)

This is a measure to calculate the values predicted by a model when compared to the actual observed values. It is illustrated in Equation 2.51

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} |Q - Q_i|} \quad \text{Equation 2.51}$$

Anti-malware developers and research community have written so much on the subject matter, as reviewed in related works

2.13.3 Confusion Matrix

The results of the trained model are evaluated to ascertain its generalization and performance with unknown dataset (test dataset). One way to achieve this is the use of confusion matrix. Confusion matrix is a cross table that records the number of occurrences between the predicted and actual classifications. While the columns represent model predictions, rows represent actual values (Kulkarni, 2022; Grandini *et al.*, 2020; Bhandari, 2020). Figure 2.25 depicts confusion matrix of multiclass classification problem for KNN with the following classes (labels): DOS, Probe, R2L and U2R. With this type of confusion matrix, unlike confusion matrix of binary classification problem, parameters such as True positive (TP), True Negative (TN), False Negative (FN), and False Positive (FP) do not apply directly (Markoulidakis *et al.*, 2021; Grandini *et al.*, 2020; Bhandari, 2020).

Because of that, the classes are analyzed one by one, with confusion matrix parameters TP, TN, Probe, R2L and U2R determined in each case. Then performance parameters such as Accuracy, Precision, Recall, F1-score, etc. are computed using appropriate formulae.

		PREDICTED VALUES			
Classes	ACTUAL VALUES	**DOS**	**Probe**	**R2L**	**U2R**
DOS		Cell 1 13979	Cell 3 36	Cell 4 0	Cell 5 0
Normal		Cell 6 355	Cell 8 562	Cell 9 0	Cell 10 0
Probe		Cell 11 114	Cell 13 3213	Cell 14 1	Cell 15 0
R2L		Cell 16 28	Cell 18 10	Cell 19 0	Cell 20 1
U2R		Cell 21 0	Cell 23 0	Cell 24 0	Cell 25 0

Figure 2.25: Multi-class Confusion Matrix for KNN

Algorithm to Compute Multiclass Confusion Matrix Parameters

To calculate TP, TN, FP, and FN for each class, the following observations on Figure 2.23 are taken into consideration (Grandini *et al.,* 2020; Bhandari, 2020). For ease of explanation, the cells are numbered, while the numerical value there-in are generated by the developed python program using scikit learn and other external libraries:

TP: This is the cell value where the predicted and actual value are the same, and for each class, only one TP value is considered

FN: For a class analysis, FN is the sum of values in cells of the corresponding row, except the TP cell value.

FP: The FP value for a class analysis is the sum of cells' values in the corresponding column, except the TP cell value.

TN: In a class analysis, the TN value is the sum of all the values in columns and rows, aside from those in the class being considered.

2.14 Choosing the Right Algorithm in the Design of Intrusion Detection System (IDS)

In machine learning, there is no one algorithm that can solve all problems (Bui et al., 2017; Pham et al., 2018). Several factors affect the choice of machine learning (ML) algorithm to build a model. Business decisions take the center stage when choosing an algorithm to build a model. From technical perspective, there is the need to run all the algorithms on the target dataset and choose the one that gives minimum loss and best available accuracy, this process is called *brute force* process. Common algorithms that feature in IDS design and development include K-NN, Naïve Bayes (NB), Decision tree (DT), GA, among others. (Ravanshad, 2018).

The list is a long one, and given the constrained period of research, and the fact that the researcher may not be exposed to all the tools used to carry out the analysis; other characteristics are considered in the choice of the right set of algorithms, they include (Ravanshad, 2018):

i. **The Type of problem to solve** – Algorithms are designed to solve specific problems using structured data set. Such algorithms may not be adequate in training with unstructured dataset (Ravanshad, 2018). ML algorithms are categorized into supervised learning, unsupervised learning, and reinforcement learning. Supervised learning algorithms are further categorized into classification and regression types. Unsupervised learning is categorized into clustered and outlier types. However, the researcher's interest is in solving a supervised learning problem.

ii. **Size of the training dataset** - High bias and low variance classifiers are good for small training datasets, such algorithms include naïve bayes (NB). They have advantage over low bias and high variance classifiers, like K-NN,

because the latter will likely over fit. However, the low bias and high variance classifiers win in the end, due to incremental learning as the training set grows, because they have low asymptotic error (training error) (Ravanshad, 2018). Since most algorithms may not be good with unstructured dataset, which are usually large, incremental learning is mostly employed.

iii. **Training time** – Different algorithms have different training times, which is a function of size of data and target accuracy. For instance, NB and K-NN train faster than ANN and SVM, which are computationally slow.

iv. **Parameters** – Parameters affect algorithm's behavior, such as error tolerance or number of iterations. It takes trial and error to find a good combination with algorithms having many parameters; although many parameters give greater flexibility to algorithms, accuracy and training time of the algorithm can be sensitive to the right setting (Ravanshad, 2018).

v. **Number of features** – When the number of features in a dataset is very large compared to the observed data points, curse of dimensionality sets in. Indeed, large number of features drags down some learning algorithms.

vi. **Outliers** – Outliers are features of one class that appear in another class, either due to classification error, typographical error, etc., and affect the learning model in training. In this research work, they are considered in the choice of learning algorithms; for instance, both SVM and K-NN ignore outliers in choosing the optimum separable hyper plane and making real-time predictions, respectively (GoodworkLabs, 2018; Harlalka, 2018). Secondly, they use small proportion of the dataset in analysis: support vectors that lie on the maximum margin of the optimal separating plane for

SVM; while K-NN uses only the k value of the nearest neighbors with the shortest distance to determine the class that will house the target object.

Ensemble learning (Bagging) is used to train base classifiers such as LR, KNN, DT, and NB using Random Forest as the learning algorithm. Ensemble learning produces a more efficient classifier, compared to the use of single algorithm, to analyze, detect and classify applications. To justify the combination of these algorithms, the following are their strengths: and shortcomings.

i. K-NN is lazy learning, it loads the entire training data set in memory and waits until the target object is introduced for analysis before it acts.
ii. K-NN selects the class for its target object by computing the shortest distance of selected training elements (the k value) using Euclidean measure; whereas SVM is required to train a model to classify the applications (malware from benign ones).
iii. K-NN is costly when used in higher dimensional spaces, while SVM is more comfortable in higher dimensional spaces.
iv. K-NN consumes memory, while SVM rather conserves memoryKNN is prone to over fitting (being low bias and high variance algorithm), while SVM is not affected by over fitting.

Combining them will correct the over fitting error in KNN.

There are also similarities that make their combination a befitting one:

i. K-NN is scalable and uses incremental learning (Heidari, 2017).
ii. K-NN is biased and uses small section of the training elements as nearest neighbors (Inayat *et al.*, 2016);

iii. K-NN has one optimal hyper parameter, the k value; and uses Euclidean distance measure to compute the shortest distance used to select nearest neighbors.

iv. K-NN is simple and easy to implement

v. Random Forest as a mega algorithm combines the predictions of base models to produce a more realistic model to assess.

Other issues to consider in choosing an algorithm for the development of IDS include (GoodworkLabs, 2018):

i. Business demands

ii. Stakeholders concerns

iii. Awareness of the rules and regulations of e-commerce

2.15 Python Programming Language

Python programming language is a high-level programming language that is easy to read and simple to implement. It is open source, interactive, interpretive, object oriented, and a scripting language. It was created by Guido Van Rossum in 1985 and it has many external libraries such as Scikit-learn, Numpy, Scipy, Pandas, Matplotlib, among others, which are used to extend python functionality, especially in machine learning (Heinold, 2016).

Python is adaptive and can be run in many platforms such as Windows, Macintosh, Unix-Linux systems; Java and dot net (. Net) virtual machines. It is copy righted and its source code is available in http://www.python.org/. It is widely applied. Some Python libraries are briefly discussed:

2.15.1 Scikit-Learn

Scikit-learn is an open-source python library used for machine learning. It can be used in classification, regression, and clustering algorithms in solving Supervised and Unsupervised machine learning problems; such algorithms include support vector machines, random forests, gradient boosting, k-means and DBScan. It is also designed to interoperate with Python numerical and scientific libraries like Numpy, Scipy, Pandas, and Matplotlib.

Scikit-learn is focused on modeling data in collaboration with NumPy and pandas that load, manipulate and summarize data. Some models provided by Skit-learn include (Brownlee, 2014):

i. Clustering – for grouping unlabeled data, such as k-means
ii. Cross-validation – for estimating the performance of supervised models on unseen data.
iii. Datasets – for training and testing models or classifiers to determined their performances.
iv. Dimensionality reduction – for reducing the number of attributes in data: summarization, visualization, and feature selection, such as PCA.
v. Ensemble methods – for combining the predictions of multiple supervised models
vi. Feature extraction – for defining attributes in image and text data.
vii. Feature selection – for identifying meaningful attributes from which to create supervised models.
viii. Parameter tuning – for getting the most out of supervised models
ix. Manifold learning – for summarizing and depicting complex multi-dimensional data.

x. Supervised models – A vast array, not limited to generalized linear models, but includes discriminate analysis, naïve bayes, lazy methods, neural networks, support vector machines, and decision trees.

Scikit_learn also allows users to load data from an external drive or data source. It works on any numeric data stored as NumPy arrays or scipy sparse metrics. To load standard columnar data into a format usable by sciket-learn, use the following (Brownlee, 2014):

NumPy – This is the database structure used for data and model parameters. Input data is presented as NumPy arrays, thus integrating with other python libraries.

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension, NumPy. It provides an object-oriented API (Application Programming Interface) for embedding plots into applications using general-purpose graphics user interface (GUI) toolkits like Tkinter, wxPython, Qt, or GTK+.

Scipy – This algorithm is good at linear algebra, sparse matrix representation, special functions and basic statistical functions

Cython – This language combines C in python and high-level operations.

Pandas library module – It is used for data manipulation and analysis. That is, it offers data structures and operations for manipulating numerical tables and time series. It is free and released under the BSD license. **T**he scikit learn library will be used to implement the proposed system, that not only detect malware but to foil their intrusion in real-time.

Various types of data are used to train and test the models developed; and tools are also found on the internet that are used to evaluate their performances.

2.16 Ethics

Ethics is a method, procedure or perspective for deciding how to act and for analyzing complex problems and issues. Ethics may be rules for distinguishing between right and wrong, such as the golden rule. Ethics may also be defined as norm for conduct that distinguish between acceptable and unacceptable behavior.

While ethical standards govern conduct in medicine, law, engineering and business, ethical norm rather serve the goals and aims of research, and apply to scientific researches and other scholarly or creative activities.

Ethics, therefore, is well funded standards of right and wrong that prescribe what humans ought to do, usually in terms of rights, obligations, benefits to society, fairness or specific virtues. Standards to refrain from include rape, stealing, murder, assault, slander and fraud. Ethical standards enjoin virtues such as honesty, compassion, and loyalty; and right to life, right to freedom from injury and right to privacy (Resnik, 2020).

To adhere to ethical norms in research has many benefits, such as:

i. Norms promote the aims of research such as knowledge, truth, and avoidance of errors. For example, prohibition against plagiarism, falsifying or misrepresenting research data

ii. Ethical standards promote the values that are essential to collaborative work. This includes trust, accountability, mutual respect and fairness. Others include guidelines to authorship, copyright and patenting policies, data sharing policies, and confidentiality rules in peer review.

iii. Ethical standards or norms ensure that researchers can be held accountable to the public. For example, conflict of interest, the human subject's protection and animal care and use.

iv. Ethical norms in research also help to build public support for research. This encourages people to fund research projects, when they trust the quality and integrity of the research.

v. Ethical norms promote moral and social values such as social responsibility, human right, animal welfare, compliance with the law, public health and safety.

2.16.1 Ethical Decision Making in Research

It is important for researchers to learn to interpret, assess and apply various research rules; and make decisions and act ethically in various situations. The following are unethical research practices:

i. Publishing the same paper in two different journals without telling the Editors.

ii. Submitting the same paper to different journals without telling the Editors.

iii. Not informing a collaborator of your intent to file a patent in order to make sure that you are the sole inventor.

iv. Including a colleague as an author on a paper in return for favour even when he or she did not make meaningful contribution to the paper.

v. Failing to keep good research records

vi. Failing to maintain research data for a reasonable period of time.

2.16.2 Ethical Perspectives

At this point, unacceptable trade-offs in health-related issues relating to universal coverage are discussed. Unacceptable choices include low-income countries, to expansion of services covering poor people in society. Universal coverage is currently among efforts to strengthen health systems and include the distribution of health and

health related services. The key issues of fairness and equity that arise on the path of universal health coverage (UHC) include ethically unacceptable trade-offs.

Every person has a right to health in terms of fairness and equity. To realize this, countries must allocate sufficient resources. Fairness is concerned with the overall distribution of benefits and burdens in society. While equity in healthcare is concerned with equitable access to services regardless to socio-economic status. In other words, a fair system will expand service coverage with financial risk protection by giving priority to policies benefiting the worst-off, where worst-off is defined in terms of health and socio-economic status. While in equity, priority to the worst-off is motivated by the right to health.

According to the world health organization (WHO), universal healthcare assumes that all peoples receive quality healthcare services that meet their needs without being exposed to financial hardship in paying for the services (Norheim, 2015). UHC goes beyond clinical and curative services, but include public health, population measures and rehabilitative services. Public health and population measures include campaigns, hygiene, and food safety, vector control and tobacco regulation.

However, in most countries, available resources fall short of what is required to meet all needs. Therefore, priority should be given to services that are likely needed by all. Unfair choices on UHC involves trade-offs between competing goals. A trade-off is a compromise between two or more desirable but competing considerations. Ethical theory is not explicit on the choice of trade-off (Norhein, 2015). Therefore, unacceptable trade-off include:

i. To reduce out-of-pocket (OOP) payments for low and medium priority services before eliminating OOP payments for high priority services. High priority services tend not to benefit the worst-offs such as skilled birth

attendance, easily treatable fatal childhood diseases such as oral dehydration therapy for children with diarrhea and pneumonia.

ii. To first include in the universal coverage scheme only those with the ability to pay, and not include the poor and informal workers, is an unacceptable trade-off. Services should be based on need and not ability to pay.

iii. It is unacceptable to give high priority to every costly service when the health benefits are very small

iv. It is unacceptable to expand coverage for the well-offs before doing so for the worse-off when the cost and benefits are virtually the same. Example, it is unfair to provide health services for tuberculosis detection and treatment in the towns before doing so in the rural villages.

v. It is unacceptable to shift from OOP payment to mandatory pre-payment to boost financing. Other constraints include discrimination based on race, ethnicity, religion, gender, political beliefs and sexual orientation.

2.17 Review of Related Literature

Security of communication systems, data, and transmission channels remains a disturbing issue in recent trends of technological developments. This is because malware developers and hackers employ various means of intrusion using sophisticated tools, most of which are freely available on the internet. More so, cyber threats to critical infrastructure continue unabated, posing a serious national security challenge. Undetected cyber-attacks force users, companies and governments to incur serious financial losses, information loss; and even the reputations of affected industries are destroyed (Sullivan, 2015). To protect and prevent the destruction of security characteristics, anti-malware developers and research community are fighting back using various techniques.

One way to fight the menace of malware and its developers is the use of machine learning techniques.

Electronic Health System

Fan *et al.* (2024) se ek to know why electronic health system is being rebuffed in developing countries like Nigeria. The method used was stimulus-organism-response (SOR) and theory of planned behavior (TPB). The work concluded that fear, education, anxiety and infrastructure accessibility are contributory factors.

Aijaz *et al.* (2023) conducted a systematic literature review on threat modeling and assessment methods in healthcare information technology system. The work opined that with the implanting of medical devices (sensors) in patients, medical healthcare is now seamless, which has made locating, monitoring and treating patients, irrespective of where they may be, common and cheap.

In furtherance to the quest for acceptance of electronic health system, Joukes *et al.* (2019) compared the EHS with the paper-based health record system, using questionnaires. The work discovered that half the people using paper-based version were not satisfied with HER for it did not meet their expectations.

Fatima and Colomo-Palacios. (2018) conducted a systematic mapping review on Health Information System (HIS) security to identify, select, classify and analyze primary studies published in scientific papers. The work observed that electronic healthcare system has become popular targets for ransomware, crypto mining, data theft, and phishing and insider threat because of their high monetary value in the black market.

Olaniyi *et al.* (2015) presented a tele-clinical diagnostic system for effective delivery of medical services to patients in an academic environment. However, the model was not secure enough as it used password-based authentication to control access to the patients' record system, which is weak enough to be breached using brute force method.

Similarly, Tashobya *et al* (2014) conducted assessment on the performance of healthcare systems, especially in low income countries like Nigeria. The objective of this work is tow fold: to develop a set of attributes for good health system performance assessment (HSPA) framework from literature; and to utilize the attributes for a structured approach to learn lessons from international experiences in HSPA.

Electronic Health Record (EHR)

Alobo *et al.* (2020) studied the pioneering work of EHR system to determine health workers perception, challenges, motivation and satisfaction with EHR using structured questionnaire. The results were reduction in transcription cost (88.5%), paper work (97.1%) and administrative cost (91.4%). However, the challenges were

i. Threat to patient privacy (17%)
ii. Poor internet service (65.7%)
iii. Information overload (31.5%)
iv. Power outages (62.9%)

On the whole, health workers were satisfied with electronic Patient Health Records (PHR) for it eased their work. Wesoloski *et al.* (2016) proposed ensemble learning technique using machine learning (ML) tools or models, to monitor keystroke-interface to stop intrusion or unauthorized access to EHR server or computer. However, the work did not mention the type of ensemble learning tools used nor did it mention the data used for the proposed work.

Security

Yeng *et al.* (2021) determined security challenges faced by healthcare workers while performing their duties. The work also attempted to investigate anomaly practices in healthcare system using big data and machine learning techniques. To justify this research, Verizon opined that in 2018 the healthcare sector experienced 503 data

breaches, compromising 15 million records (Yeng *et Al.,* 2021). Unfortunately, more than half of the threats were perpetrated by insiders. The reason for these breaches could be because healthcare records have economic, scientific, and social values, which attracts malicious actors like hackers. It is believed that healthcare records are sold at $1,000 in the black market. However, the ML technique used to analyze these breaches were not mentioned. Another problem is the lack of human firewall, which is information security consciousness in insider staff. That is, access control, which is flexible and subject to abuse by insiders.

Sardi *et al.* (2020) organized a systematic literature review on cyber risk or threat in the healthcare sector. It is unfortunate that not much attention is given to this topic by research community. Indeed, it is research gap for future work. Fortunately, this research work is closing the gap by designing and developing a framework that classifies malware threats using ensemble learning with random forest as the algorithm and DT, SVM, K-NN and NB as base classifiers.

Kumar and Tripathi (2016) modeled threat evaluation for dynamic targets using Bayesian network. The work, however, focused on external aggression and ignored internal attackers (insiders) who may affect the target through sabotage and espionage. Adebisi *et al.* (2015) designed and implemented an automated framework for managing patient's information with the view to reducing inappropriate data, false alarm, wasted time and cost in storage, retrieval and processing patient data. However, no threat model was used to plan security implementation of the system. Shin *et al.* (2014) attempted to prevent information leakage by looking into many security models for health care applications, to ensure security and privacy in electronic health system. To achieve this, the authors employed extended Role-Based Access Control (RBAC) security model. However, the model was not suitable for distributed environment. Also, to protect data

exchange between servers, securehealth architecture based on transport layer security/secure socket layer (TLS/SSL) protocol was implemented. It has the advantage of preventing foreign applications from unauthorized access However, the framework is platform dependent and not scalable (Simplicio *et al.,* 2015).

Cloud Computing Technology

Okediran *et al.* (2022) proposed a cloud based electronic health record framework that is capable of automating storage, retrieval, updating and maintaining patients' medical records in Nigeria. The work used service-oriented architecture (SOA) software development method to derive the framework. However, the work builds the framework from existing infrastructure using routers, servers firewall, gateway, desktop, laptop and smartphones. The authors did not mention the type of software developed by them, and the programming language used to develop the software. In this research work, the framework is designed and developed using ensemble learning tools (bagging) such as RF as training algorithm and base classifiers: SVM, DT, K-NN and NB. Python programming language is used for the development of the framework.

Yeng *et al.* (2020) compared threat modeling methods to determine their suitability for identifying and managing healthcare related threats in cloud computing environment. The work identified threats modeling in pervasive computing to be the best method to use in healthcare security because it can be combined with attack tree (AT), attack graph (AG) and practical thread analysis (PTA) to identify cloud related threat for healthcare. Kushala and Shaylaja (2020) investigated the essential characteristics of cloud computing and multi- cloud computing, their computing difficulties and potential solutions. The work concluded that the shift from on-premises computation to cloud computing technology has not only brought benefit but also introduced new vulnerabilities, not just for end users, but also cloud service providers (CSP). Also,

Mondal *et al.* (2020) discussed the difficulty associated with sharing resources in the cloud as a weakness that should be investigated in future study. It emphasized the difficulties that need addressing to be confidentiality, validity, privacy, cryptography, scalability, and so on.

Syed *et al.* (2020) conducted a review of security threats, procedures and control associated with cloud storage. Other concerns associated with CCT include poor data visibility, storage sinks without protected pointers, enormous data spills, and more. The work provided the following security risks associated with CCT storage to be lack of control, shared servers, data leakage, API storage sinks, and shared data. The work also proposed the following best practices: multi-factor, authentication, data categorization, security encryption and assessment of cloud framework. The work summarized the security risks in CCT to include (Saeed *et al.,* 2022):

i. Account hijacking
ii. Data sanitization
iii. Data control and
iv. Harmful insiders

Azeez and der Vyver (2018) opined that the numerous advantages of using CCT notwithstanding, the full utilization is still being obstructed by security and privacy challenges. The work, therefore, reviewed the various existing literature on mechanisms use to handle security and privacy in electronic healthcare (e-healthcare)

Anand *et al.* (2016) proposed the use of STRIDE-DREAD model to assess threats in cloud-based environment and measure the consequences of their actions. However, the DREAD framework is focused solely on technical threats and does not consider other types of threats, nor does it consider the complexity of an attack and the likelihood of

attacking a particular asset. More so, STRIDE model is complex and not suitable for organizations with limited resources.

Machine learning

Feizollah *et al.* (2014) assessed five ML algorithms (NB, K-NN, DT, MLP and SVM) with 100 malware samples of Android mal-Genome project and 12 samples of benign applications. The work used WEKA ML tools. KNN performed best with TPR of 99.94% against FPR result of 0.06%. However, the training and test data samples were grossly inadequate.

Android Intrusion Detection system (IDS)

Shatnawi *et al.,* (2022) used recursive feature selection (RFS) approach to reduce dimensionality using API calls and permission features. Logistic regression was used as the classifier. However, only one classifier was used, and the data size to use was not mention.

Wang *et al.* (2021) introduced the use of genetic algorithm (GA) to select appropriate features to improve android malware detection (AMD). The ML algorithms used were RF, LR, K-NN and GNB. The best result was obtained from GNB (95.5%) and K-NN (93.3%). However, the data type and size were not mentioned.

Akram *et al.* (2021) proposed an approach for effective intrusion detection system in the electronic healthcare environment to safe guard patient health record (PHR) using adaptive-neuro-fuzzy Inference System (ANFIS). ANFIS is used to resolve uncertainty, reasoning and reducing security threats in network and cloud servers. However, the dataset and type used was not stated or may be the proposal was not implemented.

Cai *et al.* (2020) presented an information gain (IG) based Android malware detection (AMD) framework to select optimum features from extracted features of eight categories. However, the authors did not provide the dataset used, the ML tools used,

nor even the features extracted. Similarly, Singh *et al.* (2020) proposed a framework using latent semantic indexing (LST) to reduce dimensionality of the dataset and improve detection rate. The dataset used was CICInvestAndMal2019, with RF as the classifier scoring 93.92% accuracy. However, the number of records in the dataset used was not provided, and only one classifier was used. In these days of high sophistication of malware evasion techniques, and resistant to detection, one classifier is not good enough, as recommended by the research community.

Garg and Baliyan (2019) attempted to detect zero-day attack vectors using machine learning. The models used were pruning rule-based classification tree (PART), Ripple down rule learner (RIDOR0), SVM, and MLP on a 10-fold cross validation, to improve malware detection accuracy. However, the dataset and size were not mentioned.

Mashiri *et al.* (2017) proposed a framework of malware classification, which analyzed malware dynamically using the concept of information theory and machine learning techniques. The problem with this work is it did not disclose the ML technique used nor did it disclose the data type and size used in the analysis.

Wu and Hung (2014) developed DroidDolphin, a dynamic analysis framework that used GUI, big data and machine learning tools for the detection of malicious applications in Android platform. The data used to train SVM consists of 32,000 benign and malign records, and the test dataset was 3,000 benign applications and 1,000 malign applications consisting of API calls. The results had 86.1% precision and F1-score of 0.875.

However, the dataset to train and test the model was small, more so, the authors used only one classifier to detect malware, when malware has gone so sophisticated that a combination of classifiers is advocated by research community. Hybridization or

ensemble learning methods are recommended. This research work uses ensemble learning (bagging) to train the classifiers.

Lee and Mody (2006) analyzed malware using API call sequences as data; with normalized compression distance (NCD) technique. However, most researches in dynamic analysis use one subset of malware feature to represent its behavior pattern and ignore other ones, generating blind spot, which is exploited by malware. For instance, API based sequences are used in analysis, ignoring network-based sequences, which use packet capture (pcap) files to extract the network flow information.

In this review of related literature, some frameworks presented are contextual with no constraint in ICT resources for a developing country like Nigeria. The techniques used are existing legacy systems or equipment gathered and linked together by heterogeneous healthcare systems. That is, the developed architectures integrated already existing legacy systems into EHR system rather than developing an EHR from scratch using modern ensemble techniques and network-based programming languages like python programming language.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Preamble

Healthcare system in Nigeria is predominantly paper based, and the needed interoperability among healthcare workers across units, departments in a particular hospital, and related hospitals are greatly constrained. The paper-based patient health system is fraught with many problems such as delay in fetching the patient's record, in case of emergency, the insecurity of the files from pests and internal threats, among other problems. These problems, and a lot more have motivated the researcher to design and develop a framework that will identify and report security and privacy issues in electronic healthcare system in Nigeria, which has more advantages than the paper-based healthcare system.

The focus in this research work is to examine the cyber security issues inflicting the electronic health care system, especially in Nigeria, and the safety and protection of patients' records privacy. The dataset used to develop the framework is national science laboratory – knowledge discovery in databases (NSL-KDD) dataset. To effectively use the dataset, it is cleaned, balanced using STROKE, normalized and reduced to an appreciable number using principal component analysis (PCA), and used for the training and testing of machine learning (ML) classifiers to detect and report abnormal applications in the health sector. Cloud computing technology (CCT) is used to train, test and evaluate the classifiers in their ability to detect attack types or threats. These attack (threats) types are grouped into four: DOS, PROBE, U2R and R2L.and identified using the characteristics of normal applications as bench mark.

Ensemble learning technique of ML is used to train classifiers to learn the characteristics of normal applications, and use that knowledge to detect abnormal applications. ML classifiers used in this research work include RF, KNN, DT, NB and LR. Confusion matrix will be used to compare the predictions of the record labels with the expected labels, being a supervised learning problem. The models are further evaluated to determine their performances using the following parameters: accuracy, precision, recall, f1 score and area under the curve.

Related works in literature are compared with the current work to determine their weaknesses in relation to the framework being developed.

3.2 Problem Definition

This research work intends to solve the security and patients' privacy issues in electronic healthcare delivery in Nigeria. Anomaly detection system is used in designing and developing a framework, which will learn the characteristics of normal applications and use that knowledge to detect threats in electronic healthcare system and protect patients' privacy. Machine learning tools such as Random Forest, Decision Tree, K-Nearest Neighbor, Naïve Bayes and Logistic Regression are trained, tested and evaluated for performance using confusion matrix; and performance parameters expected include accuracy, precision, recall, f1 score and area under the curve. The process is carried out in the cloud using the internet as the communication medium. Python is the programming language used to implement the framework.

3.3 Conceptual Framework (Architecture)

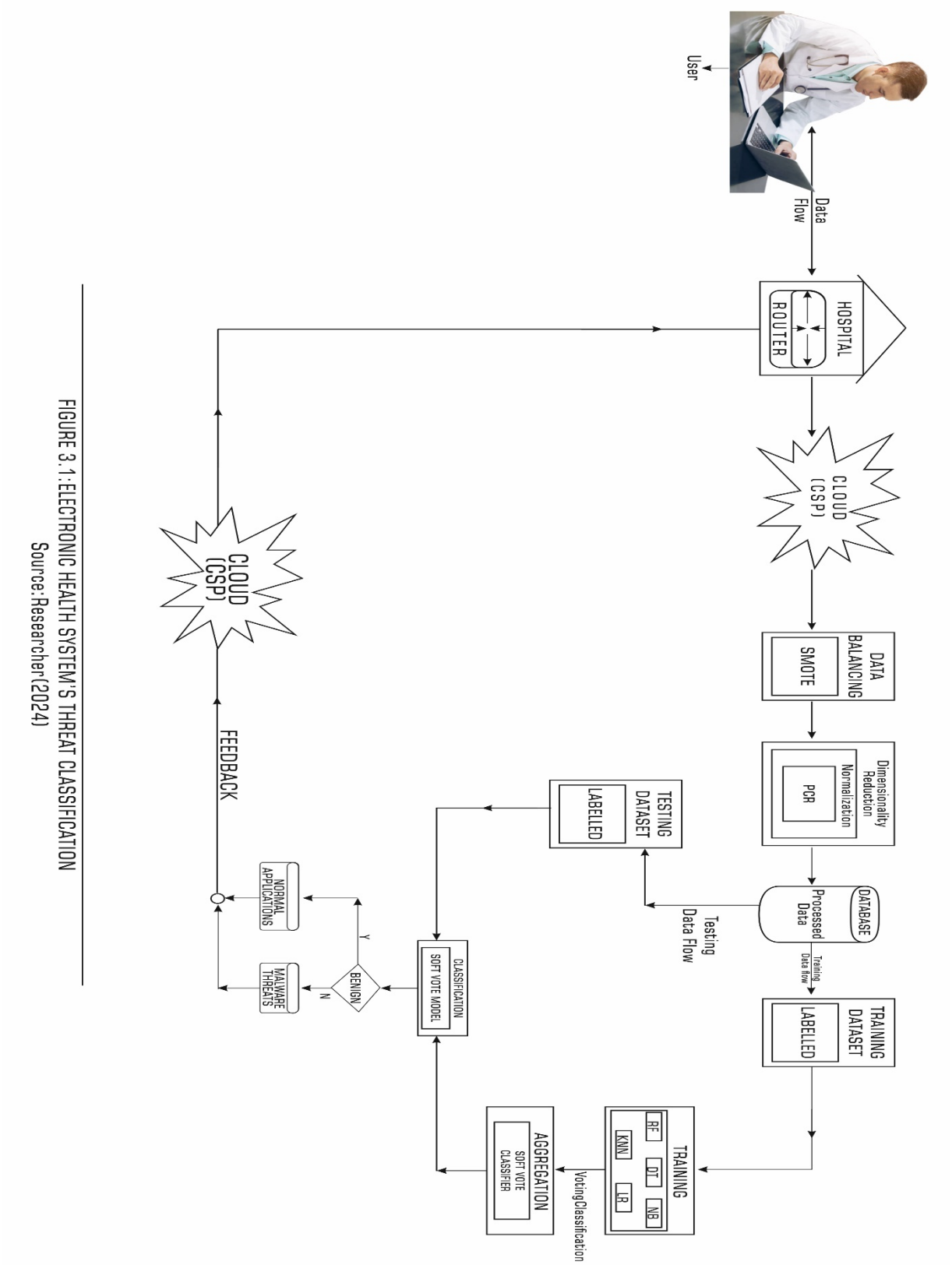


FIGURE 3.1: ELECTRONIC HEALTH SYSTEM'S THREAT CLASSIFICATION
Source: Researcher (2024)

Figure 3.1: Electronic Health System's Threat Classification Framework
Source: Researcher (2024)

Figure 3.1 depicts the system's architecture. The system trains models that analyze applications requests for patients' health records (PHR), either during updating or with the intent of mischieviously infringing on the patient's privacy. It consists of the following components.

i. **The User: -** The user is the human component of this system, who uses either the personal computer, the laptop or mobile device to communicate and access patients' records in healthcare delivery.

ii. **The Hospital -** The hospital reserves the right to control access to the medical records database, which is hoisted in the cloud and managed by cloud service providers (CSP). It receives and stores patients' data obtained from hospital workers, especially medical doctors, laboratory scientists or tecnicians, nurses, pharmacists and even the patients. To reduce its overhead, the data and virtualized device are uploaded to the cloud using mobile host agent, via the proxy device. Figure 3.2 depicts the process of uploading data and vitualized device to the cloud.

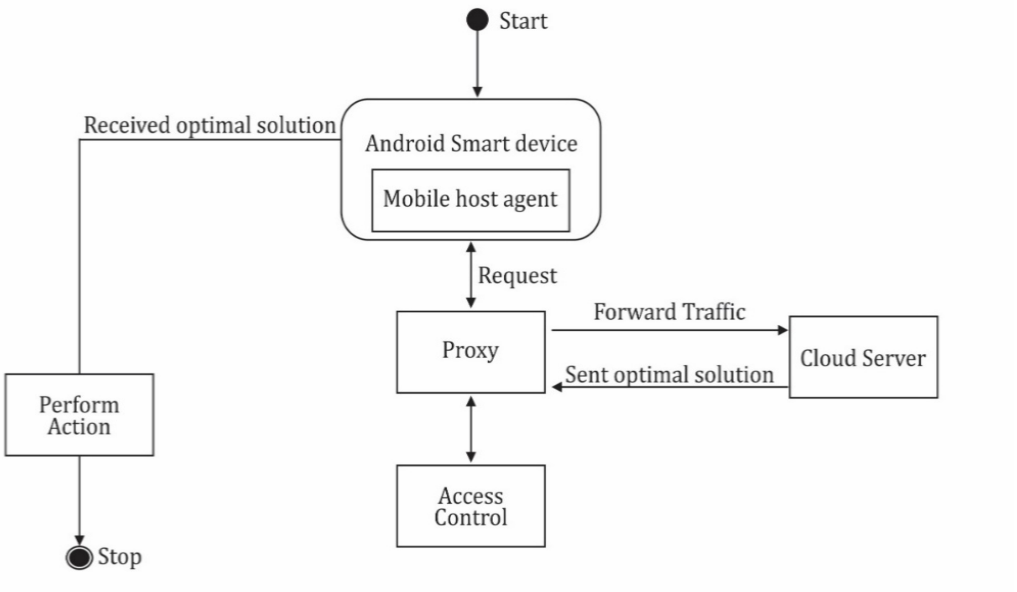


Figure 3.2: Data Movement from the User to the Server in the Cloud

i. **The Cloud**: This is the channel of communication between the hospital staff, the hospital and the patients. It is managed by cloud service providers (CSPs). Another name for the cloud in this field of study is the Internet, which is the platform for data communication to and from the user. The user can be actors such as medical doctors, nurses, laboratory scientists or technicians, pharmacists, the patients, administrators and the suppliers of srvices to the hospital such as insurers and contractors.

ii. **Dataset –** The dataset used in this research work is national science laboratory –knowledge discovery database (NSL-KDD). It consists of 125,973 training data records, and 22,544 test data records. The dataset also consists of 42 features, including labels.

iii. **Data balancing –** The dataset consists of unbalanced data, where some attack types are much more than others. For instance, normal data, neptune (an aspect of DOS) are much more than probe, U2R and R2L labels. To reduce bias in model performance, systematic minority over-sampling technique (SMOTE) from the 'imblearn' library of python, is used to oversample the data types that have insignificant values to balance with the greater ones. After the balance, the data types were reduced to 67,342 records each. The balancing of the record types becomes necessary to anable the training of models that perform well across all data types.

iv. **Dimensionality Reduction –** The dataset used in this research work is preprocessed using normalization, PCA, One_hot_encoder(), and Label_encoder():

    a. **Normalization –** This is the process tthat reduces the data set to a
    b. specific numeric range of say zero to one [0,1].

v. **PCA** further reduces the dataset by combining the original features to form new ones in a much smaller set that will fit the selected training model. It computes eigen values and eigen vectors which are used to determine the principal components (PCs) –accepted features.within a threshold of 95%.

vi. **Model training**: Random Forest algorithm is used to train four base classifiers, such as KNN, NB, DT, and LR. The classifiers predict labels. Voting classifier aggregates the predicted labels into a consensus classifier, called soft vote. The soft vote is used to predict the test dataset and compare with the actual labels using confusion matrix.

vii. **Test data classification**: The test dataset is used to classify abnormal threats from the normal applications, to provide generalization and performance evaluation of the models using the following parameters: accuracy, precision, recall, f1-score and area under the curve. Test dataset serves as proxy to the actual practical dataset, which the model did not train with.. The results of the classification are sent as feedback to the user via the hospital.

viii. **Normal Application** – These are applications from the test dataset, which have the characteristics of normal or accepted data labels. At the detection of this application, the user is notified via the hospital.

ix. **Malware Threats** – these are applications from the test dataset, whose characteristics deviate from the norms of normal applications. They are classified as threats to the system and are mischievious. Once the anomalous application is identified, the report is sent to the user, through the hospital.

Figure 3.3 depicts the flow chat representing the various stages of data processing in the architecture (Figure 3.1).

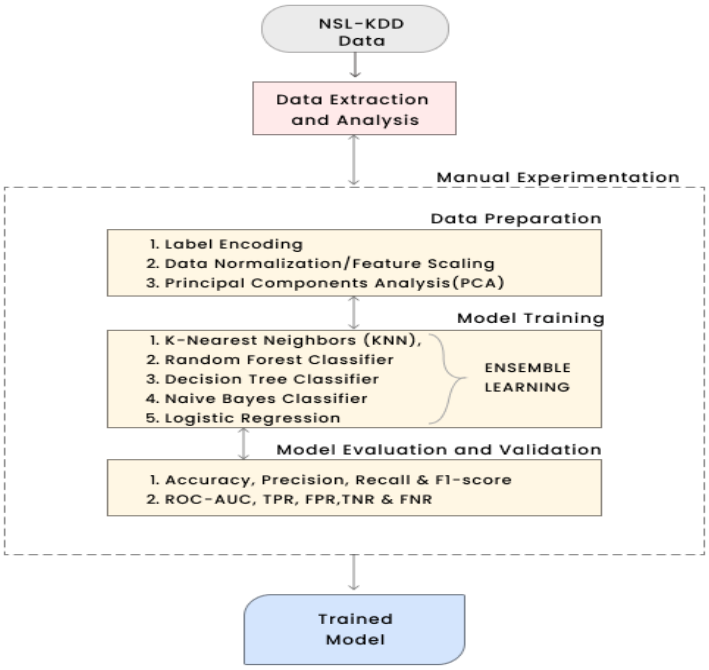


Figure 3.3: Flow Chart Showing the Stages of Data Processing in Figure 3.1

3.4 Framework Development Tools/Algorithms

The laptop system used has the following configurations: Operating System: Windows 10 Pro, with 64-bit word length, CPU: Intel ® Celeron® 1000M, 1.80GHZ; RAM: 4.00 GB, HDD 500 GB, DVD Drive, Keyboard, and Mouse.

Printer: hp LaserJet p2035

Applications: Microsoft Office Suite ver. 16: MS Word, Excel, & Power Point

Programming language used: Python 3.8 and its external libraries (Scikit learn, Pandas, Numpy, matplotlib.pyplot, etc.)

Environment used is: Cloud Computing Technology

3.4.1 Ensemble (Bagging) Learning Classifiers Used

IDS is designed and developed using the following bagging classifiers:

i. **Random Forest** – Is bagging training algorithm; for classification, the output of RF is the class selected by most trees. It corrects over fitting of the training set.

ii. **K-Nearest Neighbor (K-NN)** – Base model 1; measures the distance of each Training element in memory to the object data using Euclidean measure; and assigns the test object to the class with majority vote.

iii. **Logistic Regression** – This is supervised machine learning algorithm that accomplishes binary classification tasks by predicting the probability of an outcome. Its outcomes are limited to two possibilities, yes/no, true/ false, 1/ 0.

iv. **Decision Tree**: These are tree-based models that learn hierarchical decision rule from training data, they are used for both classification and regression problems.

v. **Naïve Bayes (NB)** – This is a probabilistic classifier based on Bayes' theorem, which assumes that the features are conditionally independent given the class labels.

vi. **Voting technique** (Soft vote).- This is a tool (function) in scikit learn module of Python. It groups the predictions of the base models; determines a consensus classifier using soft-vote technique (probabilistic vote), and uses that classifier to classify test dataset.

vii. **Output:** These are classified normal or abnormal records and their labels, grouped into Normal, DOS, Probe, U2R and R2L. The results are sent back to the user as feedback, via the hospital.

3.5 Data Collection

To identify threats or attacks ravaging the electronic health system, the dataset used to design and develop the framework is NSL-KDD dataset, obtained from Kaggle - a public data repository (Github, 2020). It is presented in the form of Attribute Relation File Format (ARFF). ARFF file is an ASCII text file that describes the list of instances that shares a set of attributes. It is divided into two subsections: Header subsection and Data subsection. The Header describes the features of the dataset, while the Data section describes the observations of the dataset as depicted in Figure 3.4.

```
%       The Header
%       Title:      NSL-KDD Data set
%       Source:     GitHub Inc. (2020)
%
%
@RELATION       NSL-KDD Data set

@ATTRIBUTE      “duration”                      real
@ATTRIBUTE      “protocol_type”                 {tcp, udp, icmp}

@ATTRIBUTE      “service”                       {aol, auth, bgp, courier
…}
@ATTRIBUTE      “flag”                          {OTH, REJ, RSTO, RSTR
...}
@ATTRIBUTE      “src_bytes”                     real
@ATTRIBUTE      “dst_bytes”                     real
@ATTRIBUTE      “land”                          {‘0’, ‘1’}
@ATTRIBUTE      “wrong_fragment”                real
@ATTRIBUTE      “urgent”                        real
@ATTRIBUTE      “hot”                           real
@ATTRIBUTE      “num_failed_logins”             real
@ATTRIBUTE      “logged_in”                     {‘0’, ‘1’}
@ATTRIBUTE      “num_compromised”               real
@ATTRIBUTE      “root_shell”                    real
@ATTRIBUTE      “su_attempted”                  real
@ATTRIBUTE      “num_root”                      real
@ATTRIBUTE      “num_file_creations”            real
@ATTRIBUTE      “num_shells”                    real
@ATTRIBUTE      “num_access_files”              real
@ATTRIBUTE      “num_outbound_cmds”             real
@ATTRIBUTE      “is_host_login”                 {‘0’, ‘1’}
@ATTRIBUTE      “is_guest_login”                {‘0’, ‘1’}
@ATTRIBUTE      “count”                         real
```

@ATTRIBUTE “srv_count” real
@ATTRIBUTE “serror_rate” real
@ATTRIBUTE “srv_serror_rate” real
@ATTRIBUTE “rerror_rate” real
@ATTRIBUTE “srv_rerror_rate” real
@ATTRIBUTE “same_srv_rate” real
@ATTRIBUTE “diff_srv_rate” real
@ATTRIBUTE “srv_diff_host_rate” real
@ATTRIBUTE “dst_host_count” real
@ATTRIBUTE “dst_host_srv_count” real
@ATTRIBUTE “dst_host_same_srv_rate” real
@ATTRIBUTE “dst_host_diff_srv_rate” real
@ATTRIBUTE “dst_host_same_src_port-rate” real
@ATTRIBUTE “dst_host_srv_diff_host_rate real
@ATTRIBUTE “dst_host_serror_rate” real
@ATTRIBUTE “dst_host_srv_serror_rate” real
@ATTRIBUTE “dst_host_rerror_rate” real
@ATTRIBUTE “dst_host_srv_rerror_rate” real
@ATTRIBUTE “class” {‘normal’, ‘anomaly’}

% *The Body of ARFF*

@DATA

0,tcp,http,SF,209,442,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,26,26,0,0,0,0,1,0,0,235,255,1,0,0,0.02,0,0,0,0,normal,21

5,tcp,pop_3,SF,26,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,233,214,0.92,0.01,0,0,0,0,0.04,0,guess_passwd

0,tcp,http,SF,327,294,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,9,10,0,0,0,0,1,0,0.2,255,255,1,0,0,0,0,0,0,normal,21

4,tcp,pop_3,SF,32,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,161,0.63,0.02,0,0,0,0,0.13,0,guess_passwd,15

0,udp,private,SF,52,54,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,0,0,0,1,0,0,255,255,1,0,0.87,0,0,0,0,0,normal,17

0,tcp,http,SF,295,527,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,8,8,0,0,0,0,1,0,0,124,255,1,0,0.01,0.03,0,0,0,0,normal,21

0,tcp,http,SF,316,356,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,4,4,0,0,0,0,1,0,0,38,255,1,0,0.03,0.02,0,0,0,0,normal,

0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,10,0,0,0,0,1,0,0,35,10,0.29,0.11,0.29,0,0,0,0,0,teardrop,11

3,tcp,pop_3,SF,30,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,247,0.97,0.01,0,0,0,0,0.02,0,guess_passwd,18

1,tcp,telnet,RSTO,123,178,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,255,12,0.05,0.01,0,0,0,0,0.03,0.67,guess_passwd,13

0,tcp,http,SF,311,294,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,27,29,0,0,0,0,1,0,0.1,255,255,1,0,0,0,0,0,0,0,normal,21

0,tcp,http,SF,227,406,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,34,0,0,0,0,1,0,0.09,64,255,1,0,0.02,0.03,0,0,0,0,normal,21

0,udp,private,SF,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,5,0,0,0,0,1,0,0,255,253,0.99,0.01,0,0,0,0,0,0,snmpguess,13

0,tcp,http,SF,314,358,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,10,0,0,0,0,1,0,0,231,255,1,0,0,0.02,0,0,0,0,normal,21

0,tcp,http,SF,243,269,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,21,21,0,0,0,0,1,0,0,255,255,1,0,0,0,0,0,0,0,normal,21
0,tcp,ftp_data,SF,19,00,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,122,15,0.04,0.05,0.04,0.13,0,0,0,0,normal,21

210,tcp,telnet,SF,126,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,248,0.97,0.01,0,0,0,0,0.02,0.02,guess_passwd,16

0,tcp,http,SF,203,454,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,11,0,0,0,0,1,0,0.18,255,255,1,0,0,0,0,0,0,0,normal,21

0,tcp,finger,SF,5,381,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,2,0.01,0.03,0,0,0.02,0,0.7,0,normal,16

0,tcp,http,SF,279,526,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,13,0,0,0,0,1,0,0.15,2,255,1,0,0.5,0.02,0,0,0,0,normal,21

0,udp,private,SF,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,253,0.99,0.01,0.01,0,0,0,0,0,snmpguess,14

1,tcp,telnet,SF,24,715,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0.67,0.33,0.33,1,0.67,255,67,0.26,0.02,0,0,0,0.01,0.7,0.69,mscan,11

0,udp,domain_u,SF,46,86,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,248,246,0,0,0,0,0.99,0.01,0,255,254,1,0.01,0,0,0,0,0,0,normal,18

4,tcp,pop_3,SF,28,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,90,77,0.84,0.03,0.01,0.03,0,0,0.11,0,guess_passwd,7

4,tcp,pop_3,SF,25,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,70,67,0.94,0.03,0.01,0.03,0,0,0,0,guess_passwd,6

0,udp,private,SF,49,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,4,0,0,0,0,1,0,0,255,254,1,0.01,0.01,0,0,0,0,0,snmpguess,16

2,tcp,imap4,RSTO,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,6,0,0,0.67,1,0.33,1,1,5
1,54,0.27,0.1,0.02,0.04,0,0,0.92,0.26,mscan,12

0,tcp,telnet,SF,125,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,109,
0.43,0.02,0,0,0,0,0.01,0.03,guess_passwd,10

4,tcp,pop_3,SF,31,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,208,0.
82,0.02,0,0,0,0,0.07,0,guess_passwd,18

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,27,0,0,0,0,1,0,1,2,60,1,0,1,
0.5,0,0,0,0,ipsweep,17

0,udp,domain_u,SF,46,77,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,188,188,0,0,0,0,1,0,0,2
55,255,1,0,0.01,0,0,0,0,0,normal,18

0,udp,domain_u,SF,45,82,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,159,160,0,0,0,0,0.99,0.
01,0.02,255,250,0.98,0.01,0,0,0,0,0,0,normal,18

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,21,0,0,0,0,1,0,1,2,42,1,0,1,
0.26,0,0,0,0,nmap,16

0,tcp,ftp_data,SF,43,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,62,0.
24,0.02,0.01,0,0.87,0.97,0,0,normal,14 *

0,udp,private,SF,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,156,146,0,0,0,0,0.94,0.01,0,
255,208,0.82,0.03,1,0,0,0,0,0,satan

0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,5,17,1,0,
1,0.12,0,0,0,0,warezclient,13

0,tcp,finger,SF,9,196,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,5,5,0,0,0,0,1,0,0,155,7,0.05,
0.06,0.01,0,0.01,0,0,0,satan,0

0,icmp,ecr_i,SF,520,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,184,184,0,0,0,0,1,0,0,255,
255,1,0,1,0,0,0,0,0,smurf,18

0,icmp,eco_i,SF,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,3,0.01
,0.02,0.01,0,0,0,0,0,satan,6

1,tcp,ftp,RSTO,0,133,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,2,2,0.5,1,0
.5,1,0,0,1,1,ipsweep,9

0,udp,domain_u,SF,34,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,0,0,0,0,1,0,1,213,22,
0.1,0.03,0.1,0,0,0,0,0,normal,21

0,icmp,ecr_i,SF,1480,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,1,25,1,0,
1,0.52,0,0,0,0,pod,17

0,tcp,pop_3,RSTO,0,36,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0,0,1,1,0.33,1,1,189,
60,0.24,0.03,0.01,0.03,0,0,0.88,0.98,mscan,15

0,tcp,ftp,SF,26,157,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,0,155,71,0.46,
0.03,0.01,0,0,0,0,0,guess_passwd,7

0,tcp,telnet,S3,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,1,0,0,255,117,0.4
6,0.02,0,0,0.53,0.37,0.01,0.01,processtable,18

0,udp,private,SF,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,254,1,
0.01,0,0,0,0,0,0,snmpguess,12

0,tcp,telnet,RSTO,124,188,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,255,
254,1,0.01,0,0,0.01,0.01,0.02,0.02,guess_passwd,10

8169,tcp,telnet,SF,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,19,0
.07,0.82,0,0,0,0,0.8,0,processtable,12

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,192,19
1,0.99,0.01,0.01,0,0,0,0,0,snmpgetattack,2

1,tcp,telnet,RSTO,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7,0,0,1,0.57,1,0,0.57,96
,87,0.34,0.05,0.01,0.02,0,0,0.9,0.7,mscan,11

0,tcp,pop_3,SF,30,217,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,2,0.0
1,0.02,0,0,0,0,0,0,guess_passwd,4

0,udp,other,SF,23,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,1,0,0.03
,0,0,0,0,0,0,multihop,9

0,tcp,telnet,SF,123,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,49,0
.19,0.02,0,0,0,0,0,0.01,0.04,guess_passwd,11

0,udp,private,SF,105,147,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,237,23
5,0.99,0.01,0,0,0,0,0,0,snmpgetattack,1

,tcp,telnet,SF,6,54,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,5,0,0.2,0.5,0.4,0.5,0.75,0.6,2
55,83,0.33,0.1,0,0,0,0,0.33,0.48,mscan,11

0,udp,private,SF,47,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,255,1,
0,0.01,0,0,0,0,0,snmpguess,15

0,tcp,telnet,SF,122,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,25,0
.1,0.02,0,0,0,0,0,0.04,guess_passwd,9

0,tcp,telnet,SF,120,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,115,
0.45,0.02,0,0,0,0,0.01,0.03,guess_passwd,11

0,icmp,ecr_i,SF,508,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,182,182,0,0,0,0,1,0,0,255,
255,1,0,1,0,0,0,0,0,smurf,18

1,tcp,smtp,SF,2599,293,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0,0,1,0,0,255,138
,0.54,0.15,0,0,0,0,0.44,0,mailbomb,11

0,udp,other,SF,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,140,1,0,0,0,0,0.01,0.01,0,8,2,0
.25,0.62,0.25,0,0,0,0.12,0,satan,10

0,icmp,eco_i,SF,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,55,0,0,0,0,1,0,1,3,47,1,0,1,0.26,0,0,0,0,saint,15

0,udp,private,SF,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0,0,1,0,0,255,255,1,0,0.02,0,0,0,0,snmpguess,16

8,tcp,ftp,SF,220,688,0,0,0,4,0,1,0,0,0,0,4,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,53,27,0.51,0.08,0.02,0.00,0.00,0.00,0.00,0.00,buffer_overflow,1

0,udp,private,SF,47,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,15

0,tcp,ftp_data,SF,12,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,7,25,1.00,0.00,1.00,0.12,0.00,0.00,0.00,0.00,warezmaster,12

0,tcp,ftp,SF,26,157,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,129,0.51,0.03,0.00,0.00,0.00,0.00,0.00,0.00,guess_passwd,9

0,udp,private,SF,46,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,snmpguess,15

282,tcp,ftp,SF,163,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,5,0.02,0.24,0.00,0.00,0.34,0.00,0.24,0.00,warezmaster,15

0,udp,other,SF,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,145,1,0.00,0.00,0.00,0.00,0.01,1.00,0.00,255,1,0.00,0.62,0.99,0.00,0.00,0.00,0.00,0.00,satan,19

282,tcp,ftp,SF,161,597,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,97,0.38,0.03,0.00,0.00,0.00,0.00,0.00,0.00,warezmaster,13

282,tcp,ftp,SF,164,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,31,0.12,0.03,0.00,0.00,0.40,0.00,0.02,0.00,warezmaster,15

,tcp,ftp,SF,162,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,17,0.07,0.02,0.00,0.00,0.14,0.00,0.25,0.00,warezmaster,12

282,tcp,ftp,SF,162,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,57,0.22,0.03,0.00,0.00,0.13,0.00,0.02,0.00,warezmaster,11

1179,udp,other,SF,146,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,33,14,0.42,0.12,1.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack,0

0,udp,private,SF,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack,1

0,udp,private,SF,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,5,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,15

0,icmp,ecr_i,SF,1008,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,303,303,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00,smurf,18

282,tcp,ftp,SF,158,597,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,72,0.28,0.03,0.00,0.00,0.05,0.00,0.02,0.00,warezmaster,12

0,udp,private,SF,49,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,255,253,0.99,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,14

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,
1.00,0.00,0.00,255,253,0.99,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpgetattack,
3

4,tcp,pop_3,SF,28,103,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.
00,0.00,0.00,255,214,0.84,0.02,0.00,0.00,0.00,0.00,0.04,0.01,guess_passwd,15

282,tcp,ftp,SF,160,597,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.
00,0.00,0.00,255,50,0.20,0.02,0.00,0.00,0.00,0.00,0.55,0.00,warezmaster,14

,udp,private,SF,41,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,12

0,icmp,eco_i,SF,30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,247,247,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00,normal,20

0,icmp,eco_i,SF,30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,227,227,1,
0,1,0,0,0,0,0,normal,20

0,udp,domain_u,SF,35,91,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,4,0,0,0,0,1,0,0,75,25
5,249,0.98,0.01,0.01,0,0,0,0,0,normal,21

0,udp,domain_u,SF,22,75,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0,0,0,0,0.5,1,0,62,5
9,0.950.03,0.02,0,0,0,0,0,normal,18

Figure 3.4: Input Dataset in Attribute-Relation File Format (ARFF)
Source: GitHub Inc (2020)

From Figure 3.4 ARFF consists of twenty-three (23) data classes. Some data classes do not have enough values or records to make them fit to train with. To fix this problem, the 23 class variables are grouped into normal type and four categories of attack. This is depicted in Table 3.1 and Figure 3.5

Table 3.1: Number of Records in each Class of the Target Variable (labels)

Classes	Number of records
Normal	67343
neptune	41214
satan	3633
ipsweep	3599
Portsweep	2931
smurf	2 646
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

In Figure 3.5, a bar chart depicts the unbalanced attack types and the pie chart depicts when these labels are grouped into normal and four attack types, DOS, PROBE, U2R and R2L

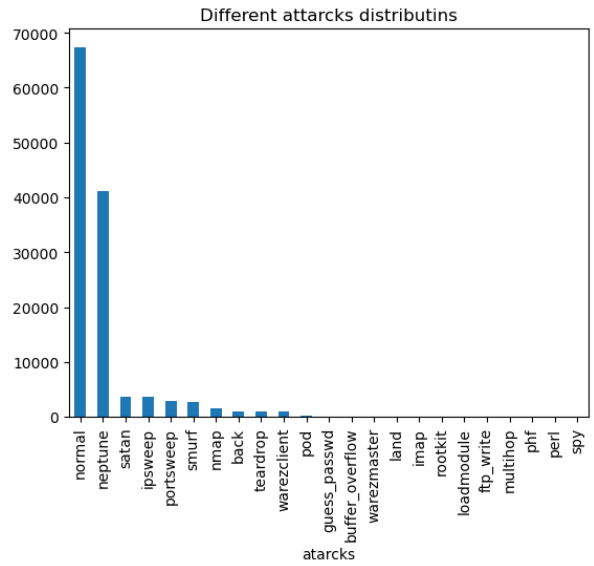
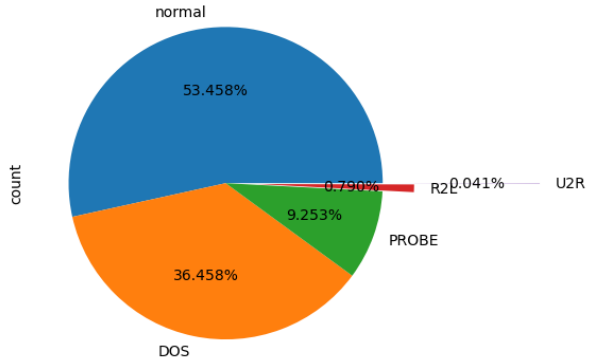
Ungrouped attacks types	Grouped attacks types using the above table
	

Figure 3.5: Bar Chart Depicting Imbalanced Attack Types, and a Pie Chart Depicting the Groupings of these Labels into Normal, DOS, PROBE, U2R and R2L Attack Types

The records of Table 3.1 are further rearranged into the four groups as depicted in Table 3.2

Table 3.2: The Grouping of NSL-KDD Attack Types

ATTACK CLASS	ATTACK NAMES
DOS	Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Udpstorm, Processtable, Worm.
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps

Source: Researcher (2024)

The data subsection of Figure 3.4 is in comma separated version (CSV) and it is classified into symbolic features, which are four (4): protocol type, services, flag, and class label; fifteen (15) float data type; nineteen (19) integer data type; and four (4) binary type , {0, 1} (Github Inc, 2020). The dataset consist of 125, 973 records (rows) in the training dataset; and 22, 544 records (rows) in the test dataset. Each record has forty two (42) columns (or attributes or criteria, or features or dimensions).

Each data record ends with the class label (type), which is either normal or anomalous (malware). From Figure 3.4, symbolic features determine the state of an application (normal or malware). Malware have affinity for icmp protocol, while normal applications use tcp or udp; Service: The following services are used by malware: private, mtp, finger, netbios_dgm, supdup, uucep_path, ftp_data, z39_50, and csnet_ns. Normal applications use the following services: http, private, smtp, ftp_data, other, telnet, and domain_u. Flag: Malware usually use S0 and REJ flags; while normal apps use SF (safe flag) flag. However, if the protocol used by an application is icmp and the

service is eco_i then the application is ipsweep malware (Probe). If the protocol used by an app is tcp and flag used is S0 or REJ, then such an app is Neptune (DOS) malware. Table 3.3 depicts the description of the forty three features in NSL-KDD dataset.

Table 3.3: Description of NSL-KDD Dataset Features

S/N	FEATURE NAME	DESCRIPTION	TYPE	VALUE TYPE
1	Duration	Length of time duration of the connection	Continuous	Integers
2	Protocol Type	Protocol used in the connection	Categorical	
3	Service	Destination network service used	Categorical	
4	Flag	Status of the connection – Normal or Error	Categorical	
5	Src Bytes	Number of data bytes transferred from source to destination in single connection	Continuous	Integers
6	Dst Bytes	Number of data bytes transferred from destination to source in single connection	Continuous	Integers
7	Land	If source and destination IP addresses and port numbers are equal then, this variable takes value 1 else 0	Binary	{ 0 , 1 }
8	Wrong Fragment	Total number of wrong fragments in this connection	Discrete	{ 0,1,3 }
9	Urgent	Number of urgent packets in this connection. Urgent packets are packets with the urgent bit activated	Discrete	Integers
10	Hot	Number of “hot“ indicators in the content such as: entering a system directory, creating programs and executing programs	Continuous	Integers
11	Num Failed Logins	Count of failed login attempts	Continuous	Integers
12	Logged In	Login Status : 1 if successfully logged in; 0 otherwise	Binary	Integers
13	Num Compromised	Number of "compromised” conditions	Continuous	Integers
14	Root Shell	1 if root shell is obtained; 0 otherwise	Binary	{ 0 , 1 }
15	Su Attempted	1 if "su root" command attempted or used; 0 otherwise	Discrete (Dataset contains ‘2’ value)	Integers
16	Num Root	Number of "root" accesses or number of operations performed as a root in the connection	Continuous	Integers
17	Num File Creations	Number of file creation operations in the connection	Continuous	Integers
18	Num Shells	Number of shell prompts	Continuous	Integers
19	Num Access Files	Number of operations on access control files	Continuous	Integers
20	Num Outbound Cmds	Number of outbound commands in an ftp session	Continuous	Integers
21	Is Hot Logins	1 if the login belongs to the "hot" list i.e., root or admin; else 0	Binary	{ 0 , 1 }

22	Is Guest Login	1 if the login is a "guest" login; 0 otherwise	Binary	{ 0 , 1 }
23	Count	Number of connections to the same destination host as the current connection in the past two seconds	Discrete	Integers
24	Srv Count	Number of connections to the same service (port number) as the current connection in the past two seconds	Discrete	Integers
25	Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)
26	Srv Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (24)	Discrete	Floats (hundredths of a decimal)
27	Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)
28	Srv Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (24)	Discrete	Floats (hundredths of a decimal)
29	Same Srv Rate	The percentage of connections that were to the same service, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)
30	Diff Srv Rate	The percentage of connections that were to different services, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)
31	Srv Diff Host Rate	The percentage of connections that were to different destination machines among the connections aggregated in srv_count (24)	Discrete	Floats (hundredths of a decimal)
32	Dst Host Count	Number of connections having the same destination host IP address	Discrete	Integers
33	Dst Host Srv Count	Number of connections having the same port number	Discrete	Integers
34	Dst Host Same Srv Rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)
35	Dst Host Diff Srv Rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)
36	Dst Host Same Src Port Rate	The percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)
37	Dst Host Srv Diff Host Rate	The percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)
38	Dst Host Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2	Discrete	Floats (hundredths

		or s3, among the connections aggregated in dst_host_count (32)		of a decimal)
39	Dst Host Srv Serror Rate	The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)
40	Dst Host Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)
41	Dst Host Srv Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)
42	Class	Classification of the traffic input	Categorical	Strings
43	Difficulty Level	Difficulty level	Discrete	Integers

Source: Saperito (2019)

From Table 3.3, there are forty-three features distributed as follows:

i. Input features are 41

ii. Label feature is 1

iii. Score or Severity of the traffic input is 1

The features are categorized into four:

i. Intrinsic features, 1 – 9

ii. Content features, 10 – 22

iii. Time_based features, 23 – 31

iv. Host_based features, 32 - 41

Intrinsic features can be derived from the packet header, without looking into the payload. They hold the basic information about the packet. The category contains features 1 - 9.

Content features hold information about the original packets, they are sent in multiple pieces rather than one. With this information, the system can access the payload. This category contains features 10 – 22.

Time-based features hold the analysis of the traffic input over a two-second window and contains information like “how many connections it attempted to make to the same host”. These features are mostly counts and rates rather than information about the contents of the traffic. The category contains features 23 – 31.

Host_based features are similar to time-based features, except that instead of analyzing over a two –second window, it analyzes over a series of connections made (how many requests made to the same host over x-number of connections). The features are designed to access attacks, which spans longer than a 2-second window time span. This category contains features 32 – 41.

The feature types can be further broken into:

i. Categorical features are four (2, 3, 4, 42)

ii. Binary features are six (7, 12, 14, 20, 21, and 22)

iii. Discrete features are twenty-two (8, 9, 15, 23 – 41, and 43)

iv. Continuous features are ten in number (1, 5, 6, 10, 11, 13, 16, 17, 18 and 19)

Protocol and service types describe the connection, while flag type describe the status of the connection, and whether a flag was raised or not. Each value in flag represents a status the connection had. Table 3.4 depicts the status of each flag in a connection.

Table 3.4: Status of Each Flag in NSL-KDD Dataset

S/N	Flag	Description
1	SF	Normal establishment and termination. It is the same as State S1. However, for S1 there will be no byte count in the summary, while SF has byte counts.
2	REJ	Connection attempt rejected
3	S0	Connection attempt seen, no reply
4	S1	Connection established, not terminated
5	S2	Connection established and close attempt by originator seen (but no reply from the responder)
6	S3	Connection established and close attempt by responder seen (but no reply from the originator

7	RSTO	Connection reset by the originator
8	RSTR	Connection reset by the responder
9	OTH	No SYN seen, just midstream traffic (a partial connection that was not later closed)
10	RSTOSO	Originator sends a SYN followed by a RST, no SYN-ACk from the responder.
11	SH	Originator sent a SYN followed by a FIN, no SYN-ACK from the responder (hence the connection was half open)
12	SHR	Responder sent a SYN- ACK followed by a FIN, however, no SYN from the originator. (This flag is not in NSL-KDD dataset)

Source: Saperito (2019); SYN: Synchronization, ACK: Acknowledgement; RST: Reset, to indicate that the originator will neither accept nor send data; FIN: Specifies that no more data will be sent by the sender

3.5.1 Corroborating NSL-KDD Dataset Using Significant Features in ARFF File

Some columns or features in ARFF file are more significant than others. For instance, ambiguities are found in some records with same values for all the forty-one features in both training and test datasets. Such redundant and ambiguous features are sifted out during dimensionality reduction using PCA preprocessing. The significant features are depicted in Table 3.5

Table 3.5: Significant Features

S/N	Attribute Names	Description
1	Src_bytes	This is the number of data bytes that is transferred from source to destination in a single connection.
2	Count	This is the number of connections to the same destination host, as the current connection in the past two seconds
3	Srv_count	This is the number of connections to the same service (port number), as the current connection in the past two seconds.
4	Logged_in	This indicates the log in status: 1 if successfully logged in, and 0 otherwise.
5	Num_compromised	This is the number of compromised conditions, especially network systems infected and turned into zombies, then added into the botnet family. Another example is the "SQL injection" where malicious code is embedded into a normal application to corrupt it.
6	Dst_host_count	This is the number of connections having the same destination host IP address
7	Num_outbound_cmd	This is the number of outbound commands issued by command and control (C&C) servers in a file transfer (ftp) protocol session to steal data, customer contact list, and carry out nefarious acts like SEND_SMS, READ_CONTACT, WRITE_CALL_LOG.

Source: Github Inc. (2020)

All the attributes in @ATTRIBUTE statement of ARFF file are open to both normal applications and malware, and each record in the @DATA subsection is a defined data type (normal or any of the attack types).

The significant features are used to corroborate the dataset. The models used to validate the dataset include min-max normalization, PCA process, and KNN. The following were computed:

i. Computation of compromised systems using smurf (DOS), ipsweep (probe), warezclient (R2L) and buffer_overflow (U2R) attack types. The number of systems compromised within two seconds attack time ranged from 0.76 to 0.99. This burtressed the min-max normalization technique that reduced data to a range of [0,1] for better performance of the trained models.

ii. Computation of principal components used to reduce the dimensions of the training dataset. This was achieved by using normal data type and ipsweep, a probe attack type, to build a covariance matrix in two dimensions. The features used in the computation were “count” and “srv_count” both were integer types. “count” indicates the number of connections to the destination host; and “srv_count” indicates the number of connections to the port numbers. This process highlights how Eigen vectors and Eigen values were computed and used to derive principal components.

iii. Computation of the nearest neighbors using Euclidean distance measure. Features used in the computation were taken from ARFF file (Figure 3.4); “src_byte” is the initial byte sent to the destination host from the source; while “dst_byte” constitutes the bytes sent from destination host to the source. The computations demonstrate the reason why ensemble learning by Random Forest was used to update the misclassified points in the base

learning models, especially as the training was done sequentially, starting with K-NN and the corrected sample_weights form a new set of data, which is used to train DT in the next iteration.Test objects will be assigned following plurality voting system by the base models. Where there is a tie, the predictions are sorted and the first one taken as the accepted prediction. Mix values of different features in different propotions of k were also used to compute the nearest neighbors. In it, the target object is assigned the class with the majority vote.

All the attributes in @ATTRIBUTE statement of ARFF file are open to both normal applications and malware, and each record in the @DATA subsection is a defined data type (normal or any of the attack types).

3.5.1.1 Computation of Compromised Systems During an Attack Process

To demonstrate the use of mini-max normalization, significant features from ARFF file are selected and used to compute compromised systems during an attack. Only attack data types are used in the computation. The attack types include smurf (DOS) attack, ipsweep (probe attack), warezclient (R2L) attack, and buffer_overflow (U2R) using min_max normalization whose formula appears in Equation 3.1

$$U_i = \frac{V_i - X_1}{X_2 - X_1} (Y_2 - Y_1) + Y_1 \qquad \text{Equation 3.1}$$

Where

X_1, X_2: are the minimum and maximum boundaries of a feature

Y_1, Y_2: are the new scale at which normalization is done, for example a range of [0,1]

V_i: the value of the attribute as indicated in the @DATA section of ARFF file.

U_i: the computed value of the normalized attribute to fall within the range.

To apply this model on the NSL-KDD dataset, features from the ARFF file were used from the four attack types as contained in Table 3.6.

These parameters are applied in Equation 3.1 where

X_1: is the src_byte feature, and is the initial number of bytes sent to the destination host.

X_2: is the count feature, which is the number of times the destination host is connected

Y_1: is the initial logged_in feature, which is usually zero (0)

Y_2: is the successfully logged in status, which is one (1)

V_i: is the destination host count with the same IP address

U_i: is the number of compromised systems in a connection session of not more than two Seconds.

Table 3.6: Features used to compute compromised systems (Min-Max normalization)

Src_byte	Count	Dst_host_count	Logged_in	
X_1	X_2	V_i	Y_1	Y_2
smurf (DOS Attack)				
520	184	255	0	1
ipsweep, Probe attack type				
8	1	2	0	1
warezclient, R2L attack type				
334	1	5	0	1
buffer_overflow, U2R attack				
220	1	53	0	1

Using smurf, a DOS attack type, with the following extracted values from Figure 3.5, as represented in Table 3.6

X_1: = 520

X_2 = 184

Y_1: = 0

Y_2 = 1,

$V_i = 255$,

U_i: is the number of compromised systems being computed.

Therefore,

$$Ui = \frac{255 - 520}{184 - 520}(1 - 0) + 0 = \frac{-265}{-336} = \mathbf{0.79}$$

That is, the number of systems compromised and added to the botnet family, within the two seconds DOS attack window is 0.79, which is within the expected range [0, 1].

Using ipsweep, a probe attack type, the number of compromised systems computed using the following features from Figure 3.4:

X_1: = 8;

$X_2 = 1$,

Y_1: = 0,

$Y_2 = 1$,

$V_i = 2.0$,

$$Ui = \frac{2 - 8}{1 - 8}(1 - 0) + 0 = \frac{-6}{-7} = 0.86$$

This also is within the normalized range, 0.86.

To compute the number of local systems compromised by a remote attacker (R2L), warezclient attack type, the following features from Figure 3.4 file were used, as depicted in Table 3.6.

X_1: = 334;

$X_2 = 1$,

Y_1: = 0,

$Y_2 = 1$,

$V_i = 5$,

$$Ui = \frac{5 - 334}{1 - 334}(1 - 0) + 0 = \frac{-329}{-333} = 0.99$$

The compromised system in this case is 0.99, and falls within the range [0, 1].

Using the privileged attacker, buffer overflow (internal attacker), which is user to root (U2R) attack type; the number of systems that could be compromised within the two seconds window were computed. The parameters used were also extracted from Figure 3.5 (ARFF file), as depicted in Table 3.6.

X1 = 220

X2 = 1

Y1 = 0

Y2 = 1

Vi = 53

$$Ui = \frac{53 - 220}{1 - 220}(1-0) + 0 = \frac{-167}{-219} = \mathbf{0.76}$$

Out of the four attack methods, the rate of compromise by an internal attacker is lower than the other three; warezclient (R2L) attack type had the maximum hit of 0.99.

3.5.1.2 Computation of Principal Components (PCs) Using PCA

Similarly, to reduce the level of ambiguity and other redundant features such as outliers, noise, etc. computation of principal component using normal and ipsweep (probe attack type) features was demonstrated using PCA in two-dimensional linear process. PCA reduces dimension of the features by computing Eigen vectors with large Eigen values, given a threshold (>= 70%) to eliminate redundant features, outliers or noise, and select acceptable principal components. An *m x n* covariance matrix **A,** with elements being the features of the training dataset, an identity matrix **I** and a column vector **V** were used in the computation. From the characteristic polynomial, the determinant of the said polynomial is derived, called characteristic equation, as shown in Equation 3.2.

$$AV = \lambda IV$$

$$(A - \lambda I)V = 0$$

$$\text{Det}\,(A - \lambda I) = 0 \qquad \text{Equation 3.2}$$

Where

A: is the m x n covariance matrix

V: is the number of rows or objects (a column vector)

I: is the identity matrix

λ: is a scalar called Eigen value

m x n: m being the observations (rows) and n is the features (columns)

Det (A - λI) = 0: indicates that the matrix used is a singular matrix. The elements used for matrix **A** are normal type and ipsweep (probe attack type). The features used from the ARFF file for the above data types were "count" and "srv_count".

Top row: (normal data)

2: "count" (integer value) (column 23)

2: "srv_count" (integer value) (column 24)

Second row (ipsweep: probe attack)

1: "count" int. (column 23)

1: "srv_count" int. (column 24)

These values are extracted from the @DATA subsection of the ARFF file. They represent normal data and ipsweep attack type in Figure 3.6

0,tcp,ftp_data,SF,43,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,62,0.24,0.02,0.01,0,0.87,0.97,0,0,normal,14 *

1,tcp,ftp,RSTO,0,133,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,2,2,0.5,1,0.5,1,0,0,1,1,ipsweep,9 *

Figure 3.6: ARFF Records From Where Count and Srv_count Values were Randomly Extracted (Columns 23 and 24 Respectively)

$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix}$$

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \lambda I = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$$

$$\text{Det } \mathbf{(A - \lambda I) = 0}$$

$$\text{Det}\left(\begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}\right) = 0$$

$$\text{Det}\begin{bmatrix} 2 - \lambda & 2 \\ 1 & 1 - \lambda \end{bmatrix} = 0$$

$$(2 - \lambda)(1 - \lambda) - 2 \text{ x } 1$$

$$(2 - \lambda)(1 - \lambda) - 2 = 0$$

$$2 - 2\lambda - \lambda + \lambda^2 - 2 = 0$$

$$= \lambda^2 - 3\lambda = 0$$

$$= \lambda(\lambda - 3) = 0$$

$$\lambda = 0$$

$$\lambda - 3 = 0$$

$$\lambda = 3$$

Compute the associated Eigen vectors.

Let the associated Eigen vector be $\mathbf{V} = \begin{bmatrix} Y \\ Z \end{bmatrix}$

Therefore, for $\lambda = 0$

$$(A - \lambda I)\, V = 0$$

$$\begin{bmatrix} 2 - \lambda & 2 \\ 1 & 1 - \lambda \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} = 0$$

$$\begin{bmatrix} 2 - 0 & 2 \\ 1 & 1 - 0 \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} = 0$$

$$\begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} = 0$$

$$2Y + 2Z = 0$$

$$Y + Z = 0$$

Let Y = 1

Then $1 + Z = 0$

$$Z = -1$$

Hence, $V_0 = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$

For $\lambda = 3$

$$\begin{bmatrix} 2 - 3 & 2 \\ 1 & 1 - 3 \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} = 0$$

$$\begin{bmatrix} -1 & 2 \\ 1 & -2 \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} = 0$$

$$-Y + 2Z = 0$$

Let Y = 1

$$-1 + 2Z = 0$$

$$2Z = 1$$

$$Z = ½$$

$$V_3 = \begin{bmatrix} 1 \\ ½ \end{bmatrix}$$

The threshold is set to >= 70%, and the spectrum of A is σ (**A**) = (0+3) = 3

$$\frac{3}{3} x \frac{100}{1} = 100\%$$

Therefore, $\mathbf{V_3}$ with Eigen value 3 produced the first principal component (PC).

The pre-processed data of NSL-KDD dataset contained in ARFF file (Figure 3.4) will be analyzed using production rules and Inference Engine to determine how data types were derived; and how IRS uses production rules (contained in knowledge base) to determine the counter measure to use during an attack.

Principal Component Analysis (PCA) Algorithm

PCA algorithm reduces the dimension of data features by computing the principal components. The steps used include:

Input parameter:

A: this is Covariance matrix

I: this is identity matrix

ⲗ : Lambda, a constant called Eigen value

V: this is the Eigen vector

i. From the characteristic polynomial, form the characteristic equation., which is the determinant as illustrated in Equation 3.3

$$Det (A - ⲗI) = 0 \qquad \text{Equation 3.3}$$

ii. Expand the determinant using row factor form

iii. Solve the resulting quadratic equation to get the Eigen values of the covariant matrix A

iv. Find the corresponding Eigen vectors V, to the Eigen values

v. Sort the Eigen vectors in ascending order, according to their corresponding Eigen values

vi. Set a threshold to use, say ≥ 70%, of Eigen values

vii. Select Eigen Vectors with the highest Eigen values within the threshold range

viii. The selected Eigen Vectors represent the principal components (PC). The first PC is assigned the Eigen Vector with the highest Eigen value, in that order.

3.5.1.3 Classification of Target Object Using K- Nearest Neighbor (KNN)

The distance measure used in this research work is the Euclidean distance measure; and optimal parameter k = 5 points represents the nearest neighbors with the shortest distance to the target point. The target object is assigned the class with majority votes among the nearest neighbors. The model is illustrated in Equation 3.4

$$D_{Euc(l}(p, q) = \sqrt{\Sigma_{i=1}^{n} (p_i - q_i)^2} \qquad \text{Equation 3.4}$$

Where p and q are points in n-space.

The value of k used in this work is k = 5.

The computations that follow illustrate the possibility of all values of k being from the same class. To illustrate the computation of nearest neighbors, Euclidean distance measure is used, which determines the nearest neighbors of the training dataset; data used in this illustration are gotten from ARFF file, Figure 3.4. Src_byte is the initial byte sent from the source to the destination host in the current layer, and dst_byte is the bytes sent to the source from the destination host. The Euclidean distance measure is an extension of Pythagoras theorem thus:

$$D_{Eucl}(xi, yi) = \sqrt{(x1 - y1)^2 + (x2 - y2)^2 + , ..., + (xn - yn)^2}$$

$$= \sqrt{\Sigma_{i=1}^{n} (xi - yi)^2}$$

Where

X_i: is the initial source byte (src_byte) sent to the destination host

Y_i: is the bytes sent from the destination host (dst_byte)

D (xi, yi): is the distance measure using Euclidean distance formula.

Table 3.7 depicts the selected values of src_byte and dst_byte, from columns 5 and six of normal records (as contained in the @ATTRIBUTE subsection) of ARFF.

Table 3.7: Attribute Extracted From Normal Data Records of ARFF File

Src_byte	Dst_byte
34	0
5	381
19	0
22	75
30	0

Computing nearest neighbors of the same class using Euclidean distance measure, with k = 5; that is, five different points in the cluster (all of them being normal data type). From Table 3.7, the following bytes were extracted from normal (benign) records and used in computing the nearest distance:

$$D_{Eucl}(x_5,y_5) = \sqrt{(34-0)^2+(5-381)^2+(19-0)^2+(22-75)^2+(30-0)^2}$$

$$= \sqrt{(34)^2+(-376)^2+(19)^2+(-53)^2+(30)^2}$$

$$\sqrt{1156+141{,}376+361+2{,}809+900}$$

$$= \sqrt{146{,}602}$$

$$D(x_5, y_5) = 382.89 \text{ bytes}$$

Where

X_5: is the src_byte from source to destination host

Y_5: is the dst_byte from the destination host to source

D_{Eucl} (x_5, y_5): is the computed result using Euclidean distance measure.

Using guesspassword (R2L) attack type in five different points (k = 5), with the same features: Src_byte and dst_byte values from the records. From Table 3.8 the following values are extracted randomly from ARFF file in Figure 3.4

Table 3.8: Attributes Extracted From Guesspassword Attack Type of ARFF File

Src_byte	Dst_byte
26	93
28	103
32	93
30	93
123	178

$D_{Eucl}(x_5,y_5)=$

$$\sqrt{(26-93)^2+(28-103)^2+(32-93)^2+(30-93)^2+(123-178)^2}$$

$$=\sqrt{(-67)^2+(-75)^2+(-61)^2+(-63)^2+(-55)^}$$

$$=\sqrt{4{,}489+5{,}625+5{,}721+3{,}969+3{,}025}$$

$$=\sqrt{22{,}829}$$

D (x_5,y_5) = 151.10 bytes.

In a similar way, nearest neighbors were computed using src-byte and dst-byte values from mscan (probe) attack type records as depicted in Table 3.9.

Table 3.9: Attributes Extracted From Mscan Attack Type of ARFF File

Src_byte	Dst_byte
24	715
0	44
0	36
0	15

6 54

$$D_{Eucl}(x_5, y)_5 =$$

$$\sqrt{(24-715)^2+(0-44)^2+(0-36)^2+(0-15)^2+(6-54)^2}$$

$$=$$

$$\sqrt{(-691)^2+(-44)^2+(-36)^2+(-15)^2+(-48)^2}$$

$$=\sqrt{477{,}481+1{,}936+1{,}296+225+2{,}304}$$

$$=\sqrt{483{,}242}$$

$D_{Eucl}(x_5, y_5) = 695.16$ bytes

From literature, the class with the shortest distance forms the nearest neighbors. In which case, guess_password, which is a remote to local attack type, is the nearest neighbor with a smaller value of 151.10 bytes, against normal features with distance value of 382.29 bytes, and mscan (probe), with a value of 695.16 bytes. The drawback of this illustration, when members of the nearest neighbors are all of the same class, is that the test target will be assigned to the class, as there are no other features of the opposite class to compare. KNN is biased to the majority class.

Although rare, the painted scenario is possible and would not give the desired result if KNN is faced with one class nearest neighbors; bearing in mind that we are dealing with a binary problem.

Situations where K=5 with the ratio of 3:2 and 4:1 are also considered.

Compute the nearest neighbors using Euclidean distance measure with K=5, using normal data type with 3 points and snmpgetattack, a R2L attack type with 2 points in the cluster of nearest neighbors. The data features extracted are depicted in Table 3.10.

Table 3.10: Attributes Extracted from Normal Data Records and Snmpgetattack Type of ARFF File; Ratio of 3:2

Src_byte	Dst_byte
Normal class labels	
46	86
45	82
52	54
Snmpgetattck class label	
105	146
105	0

$$D_{Eucl}(x_5,y_5) = \sqrt{(46-86)^2+(45-82)^2+(52-54)^2+(105-146)^2+(105-0)^2}$$

$$=\sqrt{(-40)^2+(-37)^2+(-2)^2+(-41)^2+(105)^2}$$

$$=\sqrt{1{,}600+1{,}369+4+1{,}681+11{,}025}$$

$$=\sqrt{15{,}679}$$

$$D(x_5, y_5) = 125.22 \text{ bytes}$$

Where

X_5: is the src-byte of normal data type with 3 points, and snmpgetattack (R2L) with 2 points respectively.

Y_5: is the dst-byte of normal data type with 3 points, and snmpgetattack (R2L) data type with 2 points respectively.

$D_{Eucl}(x_5, y_5)$: is the distance measure computed using Euclidean measure

Now we compute the nearest neighbors using normal data type with 3 points and processtable (DOS) attack type with 2 points in the cluster of K=5; and src_byte and dst_byte as features. Table 3.11 illustrates the random records extracted from Figure 3.4.

Table 3.11: Attributes extracted from Normal data records and Processtable of ARFF file; in a ratio of 3:2

Src_byte	Dst_byte
Normal class labels	
46	77
35	91
43	0
Processtable class labels	
0	15
0	44

$$D_{Eucl} = (x_3, y_2) = \sqrt{(46 - 77)^2 + (35 - 91)^2 + (43 - 0)^2 + (0 - 15)^2 + (0 - 44)^2}$$

$$= \sqrt{(-31)^2 + (-56)^2 + (43)^2 + (-15)^2 + (-44)^2}$$

$$= \sqrt{961 + 3136 + 1849 + 225 + 1936}$$

$$= \sqrt{8{,}107}$$

$$D\ (x_3, y_2) = 90.04 \text{ bytes}$$

Where

X_5 : is the src-byte of normal data type with 3 points, and processtable (DOS) attack type with 2 points respectively.

Y_5: is the dst-byte of normal data type with 3 points and processtable (DOS) data type with 2 points respectively.

$D_{Eucl}\ (x_5, y_5)$: is the computed distance in bytes

The next example combines attack types: smurf (DOS) having 3 points in the cluster and satan (probe) data types using 2 points in the cluster, with features used being src_byte and dst_byte. Table 3.12 depicts the extractions from the labels in Figure 3.5.

Table 3.12: Attributes extracted from smurf and satan data records of ARFF file, in a ratio of 3:2

Src_byte	Dst_byte
Smurf	
508	0
520	0
1008	0
Satan class labels	
20	0
9	196

$$D_{Eucl}(x_5, y_5) =$$

$$\sqrt{(508-0)^2+(520-0)^2+(1008-0)^2+(20-0)^2+(9-196)^2}$$

$$=\sqrt{(508)^2+(520)^2+(1008)^2+(20)^2+(-187)^2}$$

$$=\sqrt{258{,}064+270{,}400+1{,}016{,}064+400+34{,}969}$$

$$=\sqrt{1{,}579{,}897}$$

$$D_{Eucl}(x_5, y_5) = 1{,}256.94 \text{ bytes}$$

Where

X_5: is src_byte of smurf (DOS) attack data type with 3 points, and Satan (probe) attack type with 2 points respectively

Y_5: is dst_byte of Smurf (DOS) attack type with 3 points, and Satan (probe) attack data type with 2 points in the cluster respectively

$D_{Eucl}(x_5, y_5)$: is the computed result using Euclidean distance measure

From the computations, it is observed that Normal/processtable relationship has the shortest distance of 90.04 bytes. From this set, the Normal type has the majority vote of 3. The target object is then assigned to Normal.

The combination with the shortest distance forms the nearest neighbors. In this case, the combination of normal data type and processtable (DOS) data type constitutes the nearest neighbors, with a distance (D_{Eucl} (x_5, y_5) being 90.04 bytes. The test object will be assigned normal class since it forms the majority vote. The src_byte (x-values) and dst_byte (y-values) are randomly chosen from the data subsection of ARFF file of NSL-KDD dataset. The classification of the target object is depicted in Figure 3.7

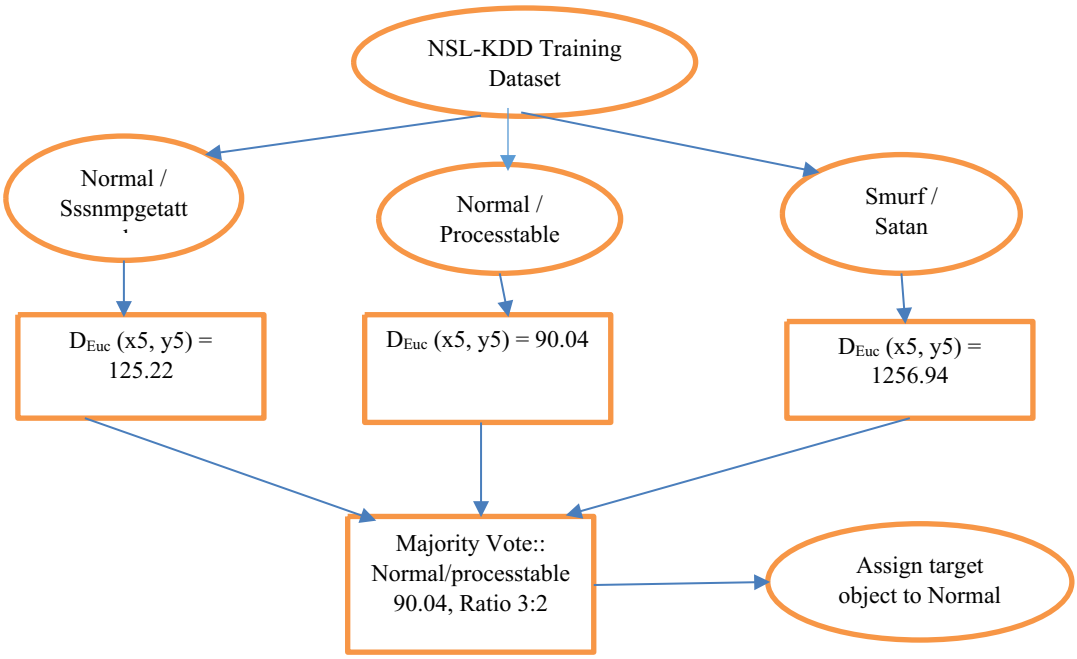


Figure 3.7: Analysis and Classification of Target Object Using KNN

Algorithm of KNN

The main steps used to classify the target object in KNN include:

i. Determine the number of nearest neighbors (the k value)

ii. Compute the distance between the target point and all training samples using Euclidean distance measure

iii. Sort the distances and determine the nearest neighbors based on the k^{th} minimum distance

iv. Assemble the classes of the nearest neighbors

v. Assign the target (test) object to the class with majority votes.

3.5.2 Data Preprocessing

One-Hot Encoder transforms Categorical Variables

To process the categorical features in the dataset, encoding was employed to transform these features into numerical representations, thereby creating additional features. Specifically, Label Encoding was applied to convert the 'protocol_type', 'service', and 'flag' columns into numerical values. Following this, One-Hot Encoding was utilized to further encode these categorical features, generating a binary column for each category within 'protocol_type', 'service', and 'flag'. Figure 3.8 depicts the dataset before encoding.

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...
0	5	tcp	smtp	SF	2429	475	0	0	0	0	...
1	0	udp	domain_u	SF	45	134	0	0	0	0	...
2	0	udp	domain_u	SF	45	80	0	0	0	0	...
3	1979	udp	other	SF	145	105	0	0	0	0	...
4	14462	tcp	other	RSTR	1	0	0	0	0	0	...

Figure 3.8: Categorical variables before encoding

The resultant encoded features were then concatenated with the original dataset as a new feature set, which increased the dimension of the dataset to 126 columns, and the

	protocol_type_tcp	protocol_type_udp	flag_REJ	flag_RSTO	flag_RSTOS0	flag_RSTR	flag_S0	flag_S1	flag_S2	flag_S3	flag_SF	flag_SH
0	1	0	0	0	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	0	0	1	0
2	1	0	0	0	0	0	1	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	1	0
4	1	0	0	0	0	0	0	0	0	0	1	0

initial categorical columns were subsequently removed to finalize the feature engineering process. Figure 3.9 depicts the dataset after encoding process.

Figure 3.9: The dataset after one-hot-encoding

On the other hand, the target variables “attack types” were manually encoded such that each attack group are mapped to a numerical value {0: ‘DOS’, 1: ‘PROBE’, 2: ‘R2L’, 3: ‘U2R’, and 4: ‘NORMAL’}.

Balancing the Dataset

The unbalanced class dataset underwent further processing to achieve balance, which is crucial to avoid bias in model performance. Using the Synthetic Minority Over-sampling Technique (SMOTE) from the `imbalance` library, the minority classes were oversampled to match the number of instances in the majority class. Initially, the class distribution showed a significant imbalance, with the 'normal' and 'DOS' classes having a much higher count compared to 'PROBE', 'R2L', and 'U2R'. After applying SMOTE, the dataset was balanced, resulting in an equal number of instances (67,342) for each class. This balanced dataset is crucial for training a model that performs well across all classes. Figure 3.10 provides a visual representation of the class distribution before and after the balancing process.

UNBALANCED ATTACKS GROUPS	BALANCED ATTACKS GROUPS

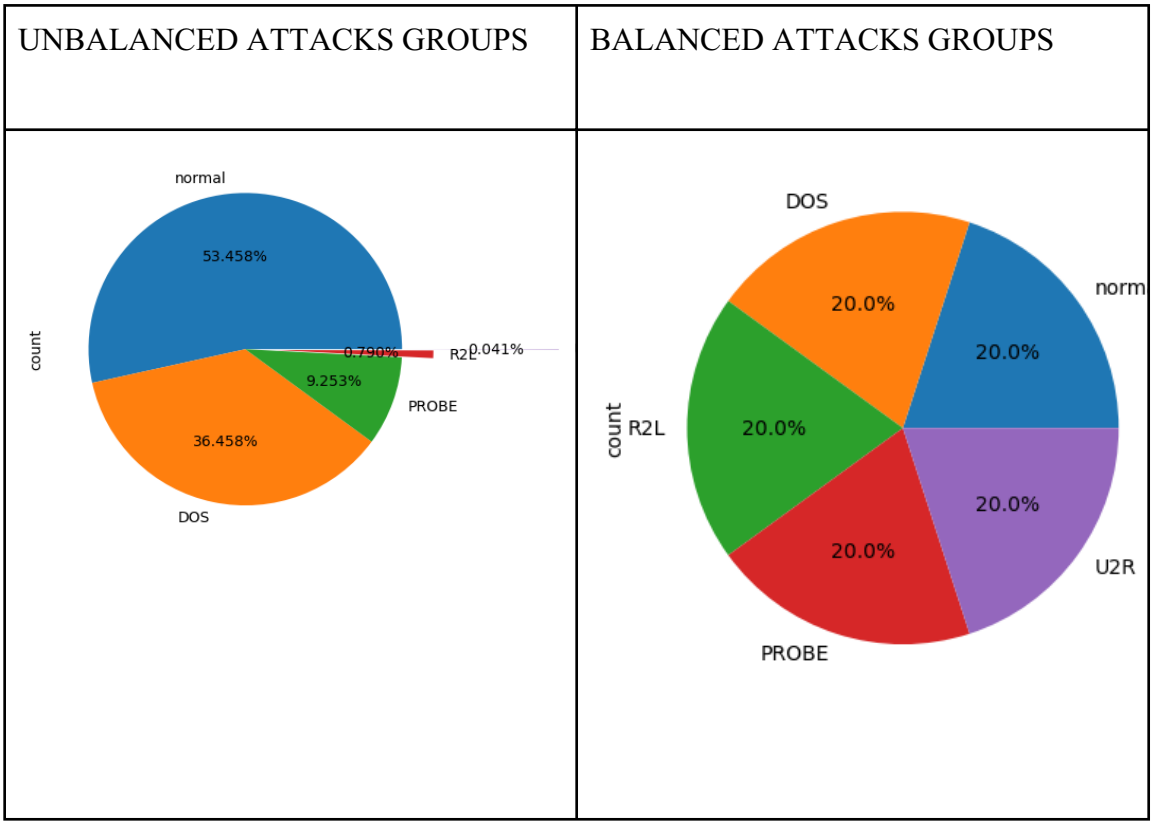


Figure 3.10: Pie Charts Showing Unbalanced and Balanced Attack Types
Source: Researcher (2024)

Principal Component Analysis (PCA)

Criteria for selecting the number of components

Given the high-dimensional dataset, dimensionality reduction was applied using Principal Component Analysis (PCA). Initially, the data was standardized to ensure each feature contributed equally to the analysis. The standardized data was then transformed using PCA to identify the principal components. The explained variance ratio was analyzed to determine the number of components required to capture at least 95% of the total variance (95% being the threshold or cut off point). A plot of the cumulative explained variance against the number of principal components was generated to visualize this relationship. Based on this plot, the number of components that explained at least 95% of the variance was selected. This dimensionality reduction step is essential to enhance model efficiency and performance by reducing the complexity of the dataset while retaining most of the information.

Figure 3.11 illustrates the explained variance against the number of principal components, highlighting the point where 95% of the variance is captured.

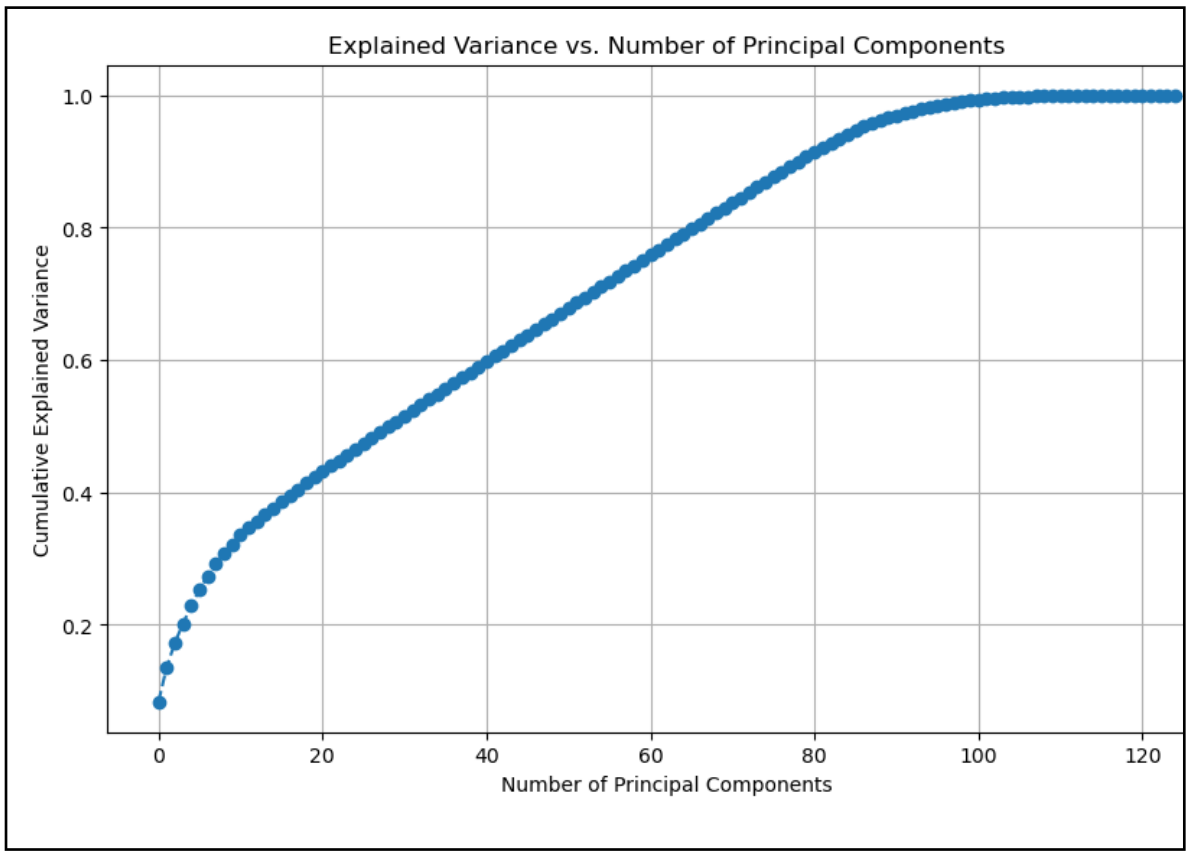


Figure 3.11: Explained variance against the principal components
Source: Researcher (2024)

Data Segmentation

After processing the dataset through sampling and balancing using SMOTE, the number of data points was increased to 336,710, ensuring equal representation for each class. Categorical encoding expanded the columns from 42 to 125, and PCA was applied to reduce the dimensions to 88 columns, retaining 95% of the variance. Due to computational constraints, a subset of 100,000 balanced data points was selected. The dataset was then split into 80% for training and 20% for testing, ensuring the model was trained on a large portion of the data to learn patterns effectively, while the testing set

provided independent validation to assess the model's accuracy and generalization capabilities. This approach ensures robust evaluation and effective predictions on new data.

3.5.3 Model Training

K-Nearest Neighbor (KNN) Classifier

Following the data segmentation strategy, 80% of the dataset was used for training, while the remaining 20% was reserved for testing. The KNN classifier was initialized with 5 neighbors and utilized the Minkowski distance metric, equivalent to the Euclidean distance measure with $p = 2$. After training the model, predictions were made on the validation dataset. These predictions were evaluated using a custom `evaluate_model` function, with performance visualized through a confusion matrix and a detailed classification report, highlighting metrics such as precision, recall, and F1-score.

Random Forest Classifier (RF)

To train and evaluate a Random Forest model, it was initialized with 500 estimators to ensure robust and stable model performance by aggregating the predictions from multiple decision trees, thus reducing overfitting. Entropy was used to measure the quality of splits, focusing on maximizing the information gain at each split. The model was trained on the training dataset (80%), and predictions were made on the test dataset (20%). To assess the model's performance, evaluate_model() function was used, and printed the classification report, and visualized the results using a confusion matrix

Naive Bayes Classifier (GNB)

To training and evaluating a Naive Bayes model, Gaussian Naive Bayes classifier was used due to its effectiveness in handling continuous data and its assumption of a normal

distribution for the features. The model was trained on the training dataset (80%), and predictions were made on the test dataset (20%).

Decision Tree Classifier (J45)

To train a Decision Tree model, entropy was used as the criterion to measure the quality of splits, focusing on maximizing the information gain at each split. The model was trained on the training dataset (80%), and predictions were made on the test dataset (20%).

Logistic Regression (LR) Classifier

To train Logistic Regression model, a maximum of 1000 points were initialized and iterated to ensure convergence. The model was trained on the training dataset, which comprised 80% of the total data, and predictions were made on the test dataset, which comprised the remaining 20%.

Ensemble Learning (Voting Classifier)

The ensemble learning approach employed in this implementation leverages the strengths of multiple individual classifiers to enhance predictive performance. The voting classifier is the core of this strategy, combining five distinct models: K-Nearest Neighbors (KNN), Random Forest (RF), Naive Bayes (GNB), Decision Tree (J45) and Logistic Regression (LR). Soft voting is utilized, where the predicted class probabilities from each model are averaged, and the class with the highest average probability is selected. This ensemble method ensures a more balanced and robust prediction. The weights assigned to each model—slightly favoring KNN and Decision Tree—reflect their relative contributions to the ensemble, fine-tuning the overall performance. The process involves defining the ensemble model with these estimators, training it on the dataset, predicting outcomes on the validation set, and evaluating its performance using key metrics such as accuracy, precision, recall, F1-Score, and ROC-AUC

3.5.4 Algorithm Used to Develop the Framework

To develop the framework that will be used to detect the menace of malware threats, the following are considered:

i. Obtain the data to train the models, which is NSL-KDD dataset, from Kaggle.

ii. Balance the data types (labels) using SMOTE technique

iii. Pre-process the balanced dataset using mini-max normalization, principal component analysis (PCA), One_hot_encoder(), Label_encoder().

iv. Split the dataset into training dataset (80%) and test dataset (20%) using train_test_split() function, in python programming language

v. Train base classifiers such as KNN, NB, DT, and LR in parallel using Random ForestClassifier() function

vi. Predict the labels using each trained classifier with model.predict() function

vii. Aggregate the predictions of the trained models using votingClassifier() function in sklearn.ensemble module to form a consensus classifier called soft vote.

viii. Train the soft vote classifier and use it to predict test dataset, and produce results.

   a. Compare the predictions of soft vote with the actual labels using confusion matrix

   b. Compute the standard metrics such as accuracy, precision, recall and f1-score, and area under the curve (AUC).

ix. Send reported alerts in the form of Normal and attack types such as DOS, Probe, R2L and U2R to the user via the Hospital.

x. Display results on the desktop or laptop or smartphone used.

3.6 Expert System

In artificial intelligence, an expert system is a computer system emulating the decision-making ability of a human expert. Expert systems are designed to solve complex problems by reasoning through bodies of knowledge, represented mainly as if–then rules rather than through conventional procedural programming code. Expert systems have four main components:

i. Data base which consists of ARFF dataset

ii. Knowledge base (Production rules)

iii. Inference Engine (represented by AIRS)

iv. User Interface

Figure 3.12 depicts an expert system.

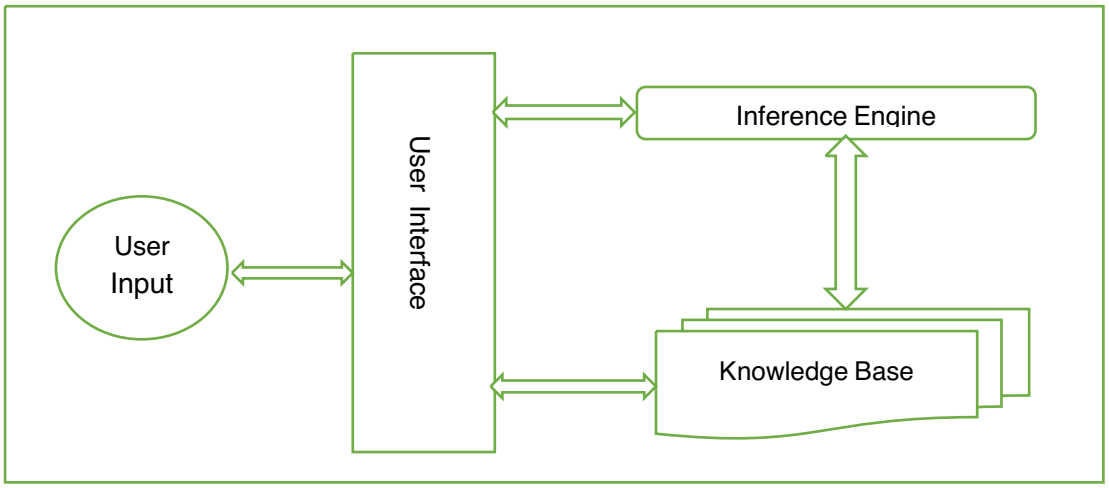


Figure 3.12: An expert system illustrating four basic components
Source: Researcher (2024)

3.6.1 Knowledge Base (Production Rules)

The knowledge base comprises of production rules represented in the form of IF..THEN statements. The features that determine if an application is normal or malware are symbolic features such as protocols, services and flags. They are represented in the

knowledge base using production rules. The protocols are tcp, udp and icmp; there are many service types such as aol, auth, bgp, and so on; so also are flags which include OTH, REJ, RSTO, S0, S1, S2, S3.

The rules, which are IF …THEN statements, logically relate information in the IF part to that of the THEN part using:

IF {Condition} THEN {Application} rules.

Some of the production rules are depicted in Table 3.13 comprising thirty-four rules.

Table 3:13: Production Rules Used to Determine the Class of Applications

S/N	Production Rules
I	IF protocol = tcp AND service = http AND flag = SF THN Application = normal; OR
Ii	IF protocol = udp AND service = domain_u AND flag = SF THEN App = normal, OR
Iii	IF protocol = tcp AND service = pop_3 AND flag = SF THEN App = guess- password (R2L) OR
Iv	IF protocol = tcp AND service = telnet AND flag = RSTO THEN App = guesspassword (R2L),
V	IF protocol = tcp AND service = ftp AND flag = RSTO THEN App = ipsweep (Probe attack type)
Vi	IF protocol = udp AND service = other AND flag = SF THEN App = snmpgetattack (R2L)
Vii	IF protocol = tcp AND service = telnet AND flag = S3 THEN App = processtable (DOS attack type)
Viii	IF protocol = tcp AND service = smtp AND flag = SF THEN App = mailbomb (R2L attack type)
Ix	IF protocol = tcp AND service = image4 AND flag = RSTO THEN App = mscan (Probe attack type) OR
X	IF protocol = tcp AND service = pop_3 AND flag = RSTO THEN App = mscan (Probe attack type) OR
Xi	xi. IF protocol = tcp AND service = telnet AND flag = RSTO THEN App = mscan (Probe attack type).
	% Policy rules for Neptune (DOS attack type)
Xii	IF protocol = tcp AND service = private AND flag = REJ THEN App = Neptune (DOS attack type) OR
Xiii	IF protocol = tcp AND service = private AND flag = S0 THEN App = Neptune OR
Xiv	IF protocol = tcp AND service = remote_job AND flag = S0 THEN App = Neptune (DOS attack type) OR

Xv	IF protocol = tcp AND service = name AND flag = S0 THEN App = Neptune OR
Xvi	IF protocol = tcp AND service = netbios_ns AND flag = S0 THEN App = Neptune OR
Xvii	IF protocol = tcp AND service = mtp AND flag = S0 THEN App = Neptune OR
Xviii	IF protocol = tcp AND service = finger AND flag = S0 THEN App = Neptune OR
Xix	IF protocol = tcp AND service = supdup AND flag = S0 THEN App = Neptune OR
Xx	IF protocol = tcp AND service = uucp_path AND flag = S0 THEN App = Neptune OR
Xxi	IF protocol = tcp AND service = z39_50 AND flag = S0 THEN App = Neptune OR
Xxii	IF protocol = tcp AND service = csnet_ns AND flag = S0 THEN App = Neptune OR
Xxiii	IF protocol = tcp AND service = efs AND flag = S0 THEN App = Neptune (DOS)
	% Policy rules for data types sharing the same symbolic features
Xxiv	IF protocol = udp AND service = private AND flag = SF THEN App IN {normal, teardrop, snmpguess, snmpgetattack, satan}
Xxv	IF protocol = tcp AND service = ftp_data AND flag = SF THEN App IN {normal, warezclient, warezmaster}
Xxvi	IF protocol = tcp AND service = finger AND flag = SF THEN App IN {normal, Satan}
Xxvii	IF protocol = tcp AND service = telnet AND flag = SF THEN App IN {processtable, guess_password, mscan}
Xxviii	IF protocol = udp AND service = other AND flag = SF THEN App IN {Satan, snmpgetattack, multihop}
Xxix	IF protocol = tcp AND service = ftp AND flag = SF THEN App IN {warezmaster, buffer_overflow, guesspassword}
Xxx	IF protocol = icmp AND service = ceo_i AND flag = SF THEN App IN {Satan, ipsweep, saint}
Xxxi	IF protocol = icmp AND service = ecr_i AND flag = SF THEN App IN {smurf, pod}
	% Policy rules for AIRS (Automated Intrusion Response System) % ON {Event} IF {Condition} THEN {Action} % Aggressive action policy
xxxii	ON ANOMALY DETECTION IF role != IN {SA, DBA,CEO} and Source IP 192.168.0.0/16 and

	Object type = Table and Object name IN {Server, Host, Network} and Command IN {READ_SMS, WRITE_SMS, SEND_SMS, MODIFY_PHONE_STATE} Hardware Apps IN {GPS, GPRS, CAMERA, CONTACT LIST} THEN Abort and PRINT {Action = Abort}
	% Policy rule for conservative action, where the target is not so important to the Organization
Xxxiii	ON ANOMALY DETECTION IF Confidence metric LOW and Target Object NOT CRITICAL THEN NOP and PRINT {Action = ignore}
	% This policy re-authenticates the privileged user (inside attacker)
Xxxiv	ON ANOMALY DETECTION IF request IN {Contact list, media filter, log file, GPS, internet filter} and INTENT IN {connectivity_change, uninstall_shortcut, view_phone_state} THEN put on_hold and CONFIRM {second authentication} ON SUCCESS NOP and PRINT {Action = Continue} ON FAILURE Abort and Revoke PRINT {Action = Abort, Revoke}

These rules are traversed when the inference engine receives facts from the user through the user interface or anomalous report from IDS.

3.6.2 Inference Engine (IE)

Inference engine is a device used by Expert system to reason by matching facts supplied by the user, through the user interface, with the rules formulated and stored in the knowledge base and then draws conclusion.

It has two reasoning processes:

i. Backward chaining (modus tolens), which starts reasoning from the solution or goal, and works backward to find facts that support the goal; tests some rules that are relevant to the problem at hand; it is bottom-up

and goal-driven. It is suitable for problems that start from a hypothesis, such as medical diagnosis of a patient.

ii. Forward chaining (modus ponens) starts from the set of facts and moves towards the conclusion or solution.

In this research work, forward chaining will be used to process the facts that match with the rules. It will be used because it is suitable for problems that start with data collection such as planning, monitoring and control; it starts with initial facts, tests all the rules, and it is data-driven. Figure 3.14 illustrates the sequence of events in forward chaining process as modified from Al-Ajlan (2015).

From Figure 3.13 the sequence is mimicked as follows:

Referring to rule number **xxxiii** in the knowledge base**,** the anomalous application is requesting for dangerous permissions and IE scans the rules and found that the request is dangerous, and aborts the application. “Abort” will be reported to the user.

Secondly, assuming IE receives this information from the ARFF file,

protocol = tcp, service = telnet and flag = RSTO

it will scan the rule base beginning from the first rule, which will not match, it checks the next rule, which will still not match; and then the next rule will not match. It is rule number **iv** that matches and guesspassword (R2L) will be added to the working memory (WM). Once the rule has fired, then IE goes back to the WM for another input from ARFF file until there are no more inputs from the file. If all the 33 rules do not match the set of variables in a particular record, IE will then return to the user, through the user interface, and ask for information about the set of values presented.

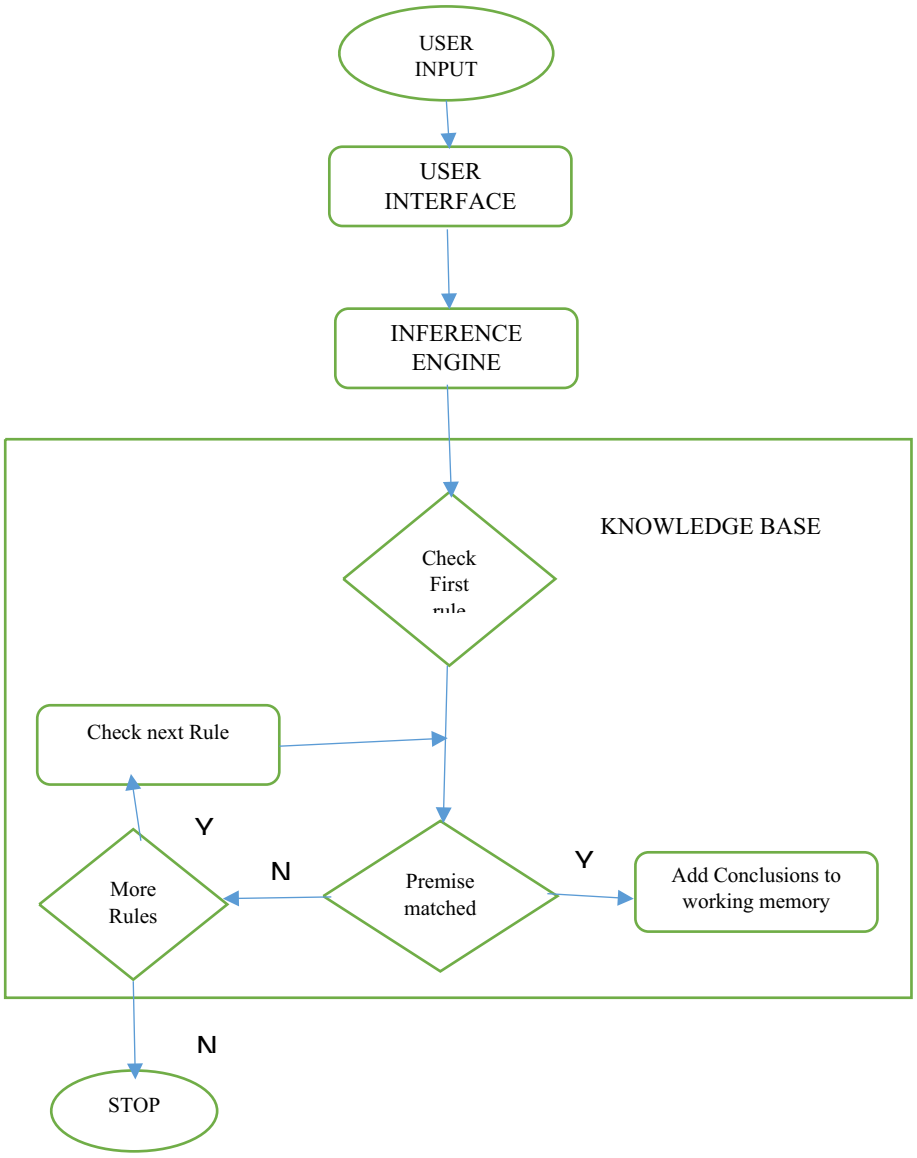


Figure 3. 13: Forward chaining inference process.
Source: Researcher (2024)

3.6.3 The User Interface

The user interface enables the communication between the user and the Inference engine, to perform specific tasks. It is also required to translate and interpret the input for the application of rules. It is depicted in Figure 3.14. From Figure 3.14, the problem to be solved is stated at the top of the user interface; the user first requests to use the

system by selecting the system from the menu. The system then requests for user details, and the user enters his or her details through the user interface, which will be validated by the system. If the user's details are not correct, he or she will be given only two more chances to input the correct details or access to the interface will be denied. The user in turn enters the facts of the problem in the space below for use by the inference engine to match with the rules in the knowledge base. The results of the system are communicated to the user through the *record label* and *action taken by IE* boxes.

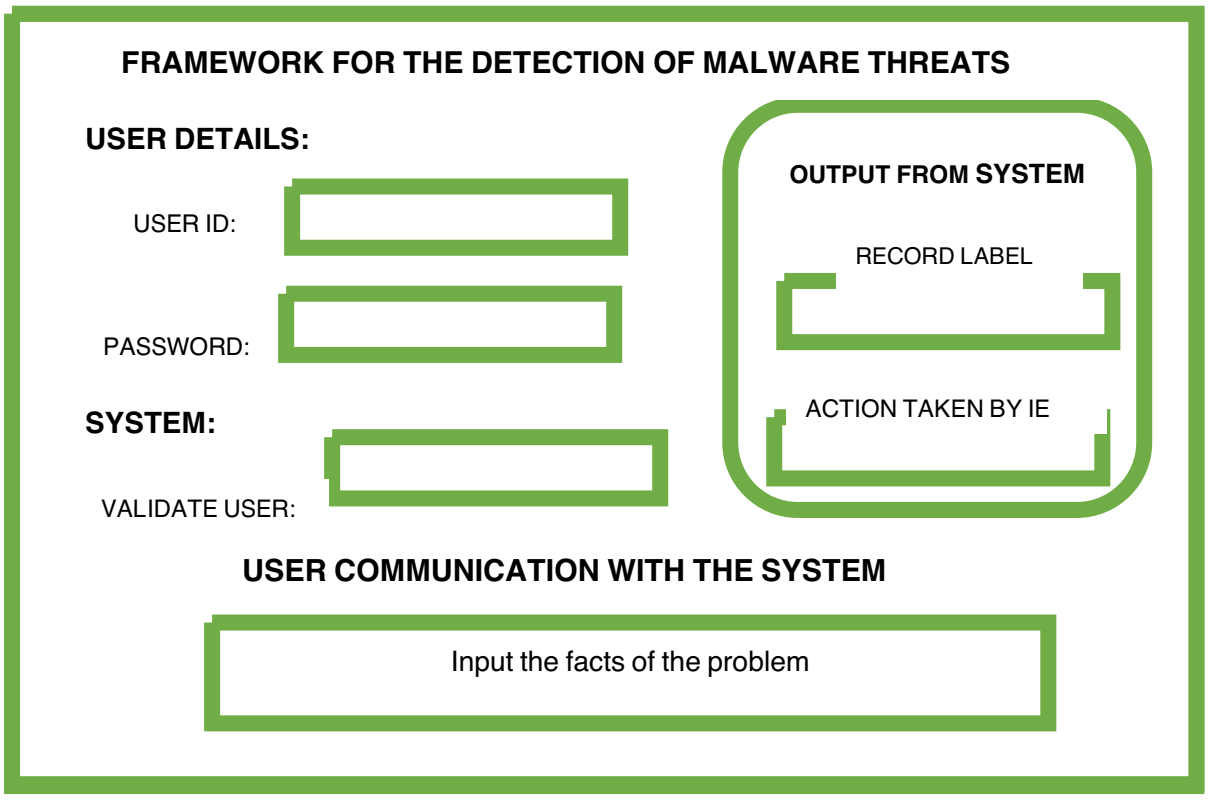


Figure 3.14: The User Interface for Communication Between User and Inference Engine

3.7 System Design

The objective of this research work is to design a Framework that will monitor applications seeking to use the EHR system to either install third party applications or to mischievously disrupt the operations and steal vital information after gaining access to the system. Malware developers usually request for more permissions to be

mischievous when the need arises, especially making use of dangerous permissions. For the Framework to be up and doing, Ensemble learning, piloted by Random Forest, is used to train base (weak) models in parallel, and use a voting technique to ascertain the cooperation of the separate models that predicted labels independently.

The system design is implemented using unified modeling language (UML). In this research work, four system design components such as user interface (UI), use case diagram, class diagram and sequence diagram will be designed using UML. They include:

(i) Use case diagram (dynamic)

(ii) Class diagram (static)

(iii) Sequence diagram (dynamic)

(iv) User interface

These components interact with the system through the user interface.

3.7.1 Use Case Diagram

A use case diagram specifies the interactions of external objects (Actors), which are either human or applications or systems, with the system being developed. Actors used in this work are depicted in Table 3.14.

Table 3.14: Actors Definition

S/N	ACTOR NAME	DESCRIPTION
1.	User	This actor uses the laptop to link with the system by logging in and provide user details, through the user interface. The user may decide to cancel the process.
2.	Administrator	This actor is a privileged user (the hospital) that links the user with the system, after

		verifying the user's details. Admin also gives the user privilege to update his or her details and provides the enabling environment for logging out.
3.	The CSP Server	This actor is a repository where updates of the user details are stored and, confirmation of the user details are made when required.

Assumptions

i. It is assumed that the user has subscribed with at least one network operator to get access to the internet

ii. The laptop used in this work is assumed to use Windows operating system

iii. Given that the analysis is dynamic using CCT, data collected are uploaded to the cloud for processing

Figure 3.15 depicts use case diagram.

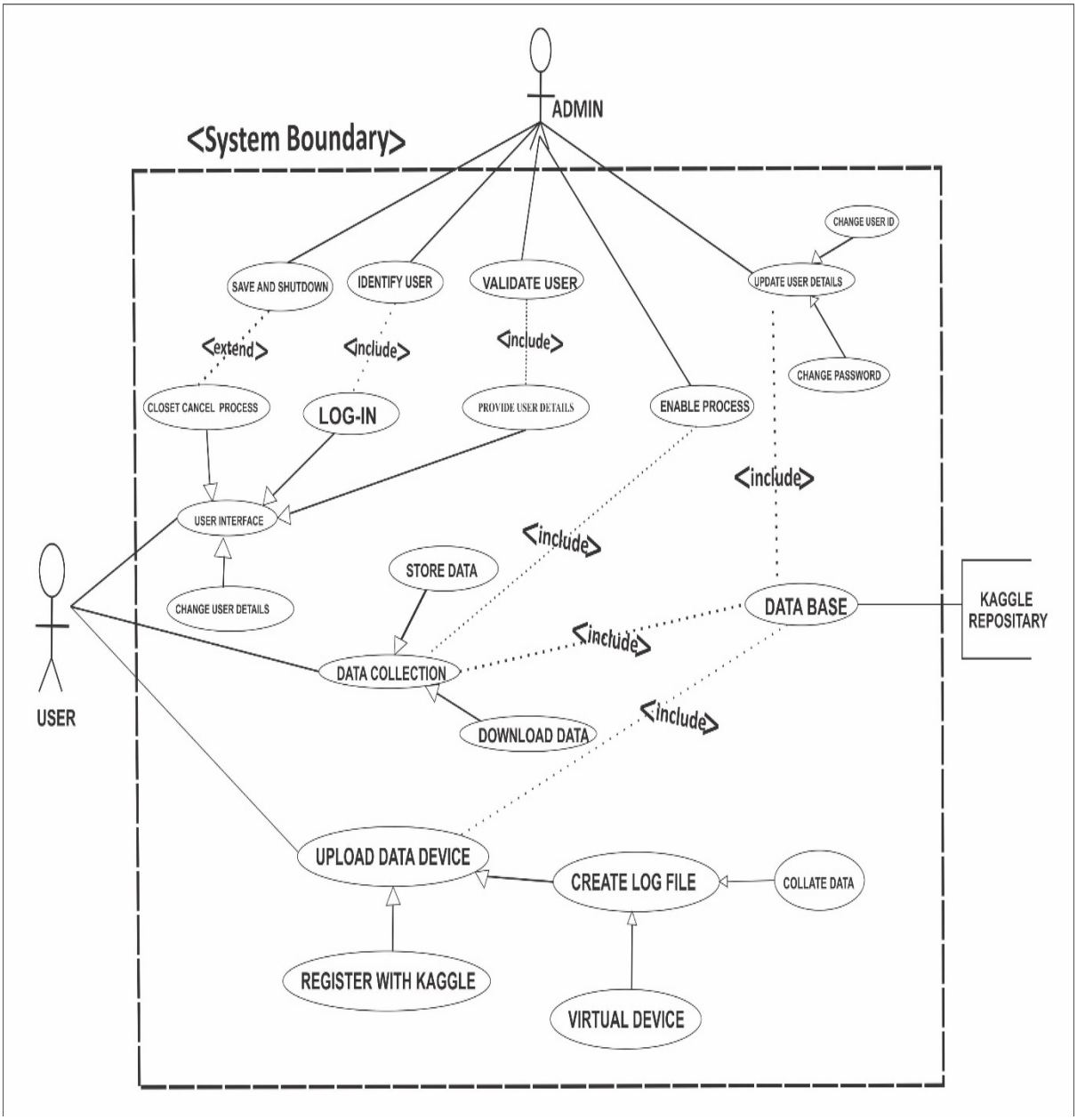


Figure 3.15: Use Case Diagram
Source: Researcher (2024)

Table 3.15, Table 3.16, and Table 3.17 depict use case definitions

Table 3.15: Use Case Definition 1

USE CASE NAME	CODE	DEFINITIONS
User Interface	UC001	The user accesses the system through the user interface by providing identity details as he or she logs_in.
Actors		User, Administrator and CSP Server repository

Events in Use Case Definition 1

EVENT	ACTOR	SYSTEM
	1) This use case is initiated when the user navigates to the E-Hospital platform.	2) The system requests for identification.
	3) The user logs in to the platform by providing the following details (i) user ID (ii) Password	4) The system validates the user. 5) Displays the E-Hospital platform with menus. 6) From the menu, the customer has the option of Editing the user ID or password or cancel the process or activate any process like loading data to the system memory.
	7) However, the user can decide to cancel the transactions.	8) The system terminates the process and signs out the user, then shuts down.

Table 3.16: Use Case Definition 2

USE CASE NAME	USE CASE CODE	DESCRIPTION
Data collection	UC002	The user collects the data to use and train/test models from CSP server repository and stores in the device.
Actors involved.	User, Admin, CSP Server repository.	

Events in Use Case Definition 2

Customer	System
Step 1: From the menu, the user chooses to download HER from CSP Server repository	2). The system enables the process.
	3) CSP effects the transfer of the dataset through the internet
Step 4: The user then stores the received dataset in the device memory	6) The system signs out the user
5) The user concludes the process.	

Table 3.17: Use Case Definition 3

USE CASE NAME	CODE	DESCRIPTION
Upload data	UC003	Given the constraints of the device, the user then registers with CSP using the address: https://www.kaggle.com in order to access kaggle's console and use it to process transactions in the cloud
Actors		User, Admin, and CSP Server repository.

Events in Use Case Definition 3

ACTOR	SYSTEM
Step 1: This use case starts with the user registering with Kaggle online.	2) The system provides enabling environment
	3) Kaggle asks for user identity.
step 4: The user provides details such as user ID and password.	
	5) Kaggle registers the user and provide him or her access to its console
Step 6:	
The user opens a log file and collates both data and virtual image of the device.	7) The system provides the enabling environment
Step 8) The user uploads the log file to the proxy server	
	10) The proxy server in turn forwards the log file to the cloud for processing.
Step 9) The user saves a copy of the log file in the device	

3.7.2 Class Diagram

A class diagram provides the structural (static) view of the system.

Classes are represented by boxes, which are subdivided into three parts:

- The class name
- The attributes
- The method (behavior)

In most cases, the attribute subsection contains private features, which are manipulated by the methods of the class only; while the methods are mostly public or protected, where they can be called by other classes or sub classes.

The visibility elements mentioned above are summarized below

- Private - used by the class methods only

+ Public - can be called by other classes' methods

Protected – called by the class and sub classes only

Mobility values indicate the number of objects each class uses or participate in a session. They are illustrated in Table 3.18.

Table 3.18: Multiplicity Symbols and Meanings

Symbols	Meaning
0	None
1	One
M	An integer value
0..1	Zero or one
0..*	Zero or more
1..*	One or more
*	Any non-negative integer (Zero or more)

Figure 3.16 depicts the class diagram of user interface process

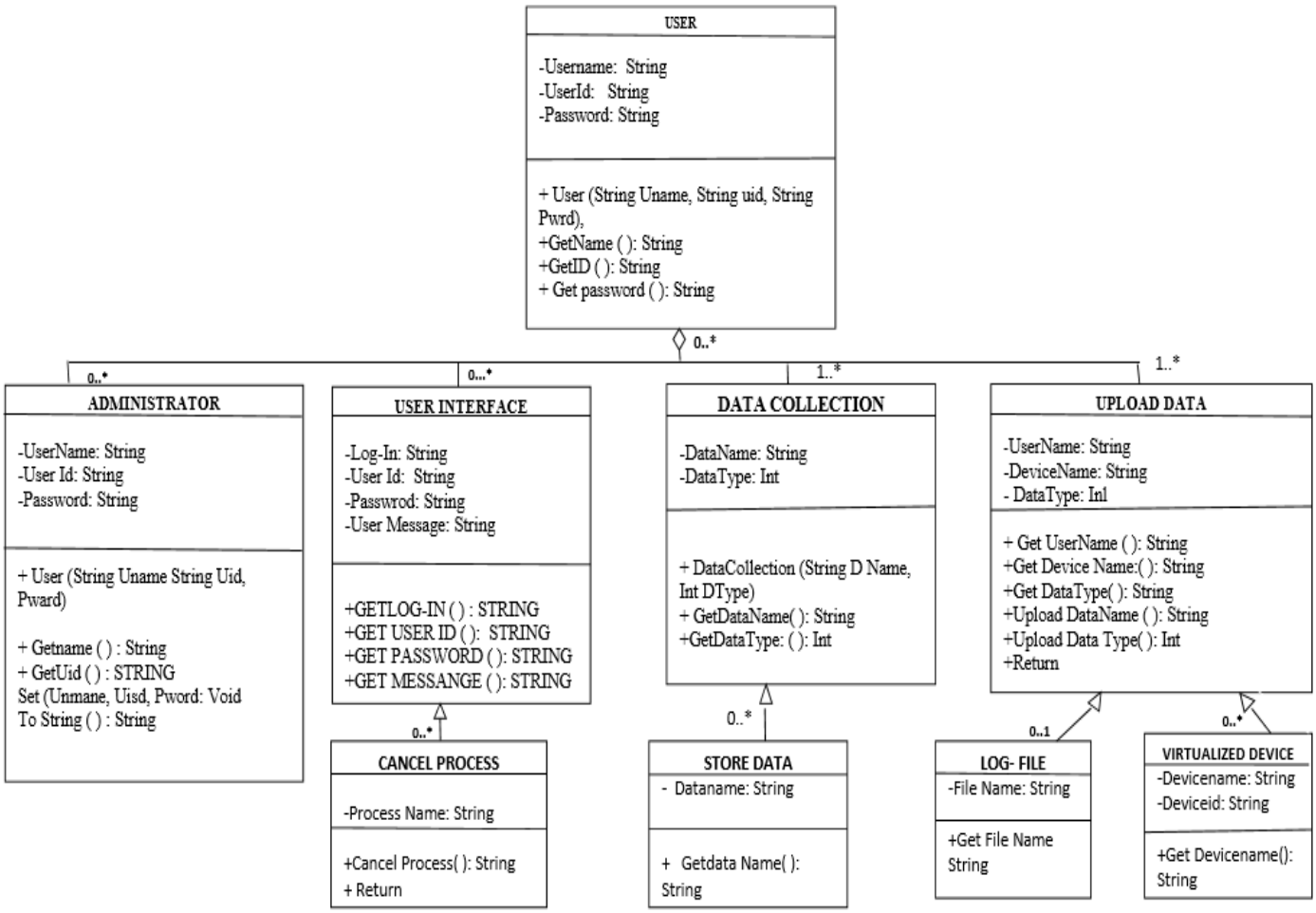


Figure 3.16: Class Diagram
Source: Researcher (2024)

Table 3.19 depicts the process relationships in the Class diagram

Table 3.19: Process Relationship in the Class Diagram

CLASS	CLAUSES
User	A single user can aggregate 0 or more Transactions.
Administrator	This is a surrogate to the system, but as well a user and can access one or more processes.
User Interface	This is the gateway between the user and the system. It accesses one user at a time. It has a child subclass, cancel process, which inherits the features of the parent.
Data Collection	Data collection is related with the system and database, therefore one or more processes can be accessed. It has a child subclass, store process, which inherits its features and methods (that are either public or protected) but can over-write any method of the parent.
Upload Data	This process is aggregated to the user and relates with the admin and database where data is downloaded from and updated. It has two child processes, log-file and virtualize device
Cancel Process	This child class is a subclass to user interface and can inherit its features and methods, but has a right to over-write any method. It accesses zero or more processes.
Store Data	This subclass is a child class of Data collection use case, and can inherit all the methods and features of the parent class. It can access zero or more objects.
Log-file	This process is a sub class to upload Data process, and can inherit its features and methods. It has zero to one mobility.
Virtualize Device	This process is a sub class to upload Data process, and can inherit its features and methods. It has zero to one mobility.

3.7.3 Sequence Diagram

A sequence diagram models the interactions between objects in a single use case. In this research work, the sequence diagram of user interface process is illustrated in

Figure 3.17. It illustrates how the user, system admin and CSP Server repository interact with each other to carry out a function.

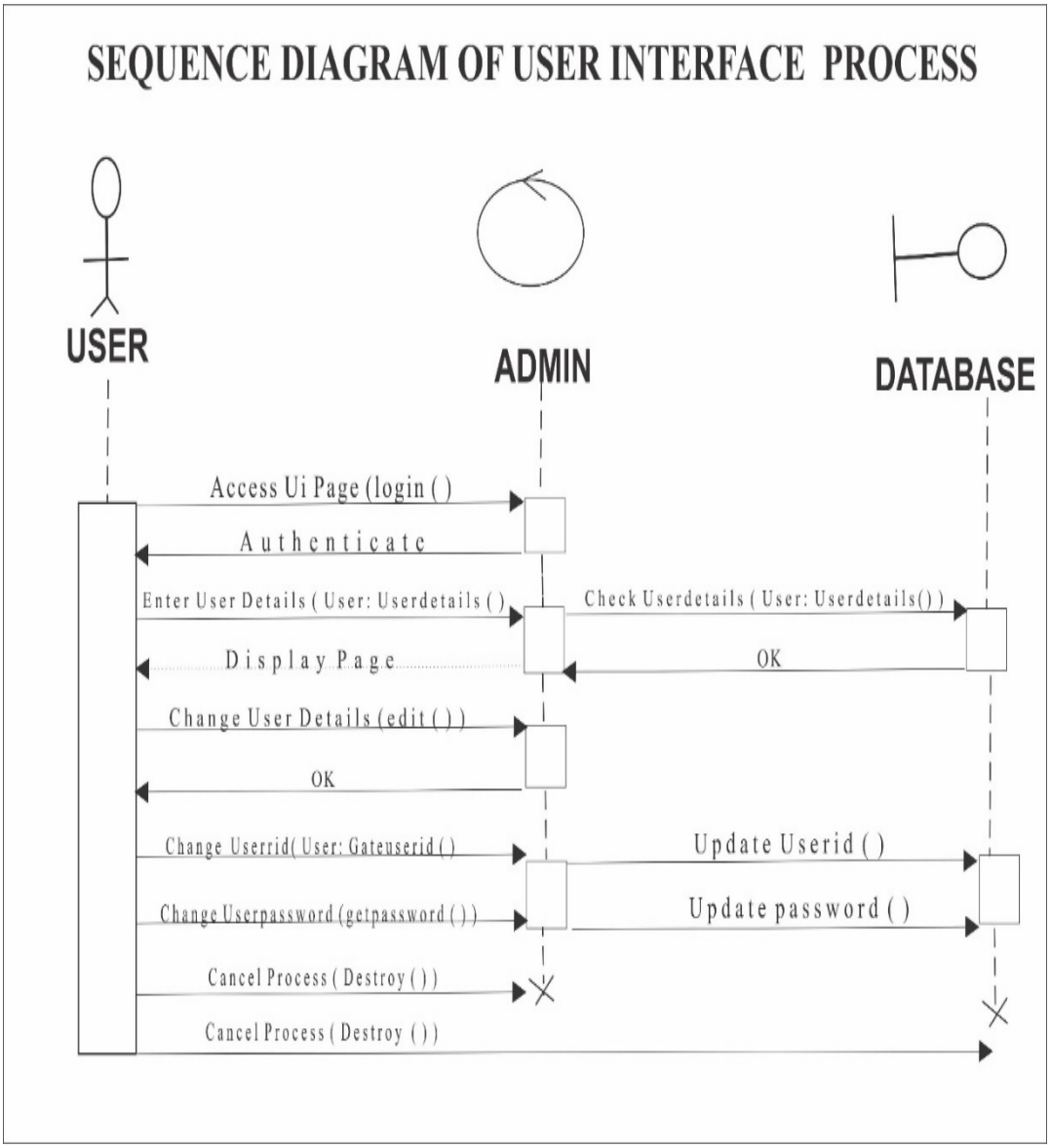


Figure 3. 17: Sequence Diagram
Source: Researcher (2024).

Sequence Diagram Details on User Interface Process

The following are the steps or interactions between the user and the system

i. The user initiates access to the user interface (UI)

ii. The system requests for user identity

iii. The user provides user details:

- UserID ( ): string
- Password ( ): string

iv. The system verifies user details from the database

checkUserDetail ( )

v. User decides to change his or her details:

getUserID ( ): string

getPassword ( ): string

vi. Administrator updates the database

updateDatabase ( )

vii. User decides to cancel the process:

destroy ( )

viii. The system updates the database:

updateDatabase ( )

CHAPTER FOUR
RESULTS AND DISCUSSION

4.1 Preamble

In this Chapter, the results, analysis of the results using confusion matrix and discussion of the results are presented. This research work leverages on the security of the healthcare system and patient's privacy in the cloud, and designs and implements a framework to analyze and detect unfriendly applications that would undermine security and abuse the privacy of patients' records, using machine learning tools. Given the sophistication of malware threats, research community approves the combination of classifiers to produce a robust classifier to detect the stealthy malware threats and safeguard the electronic health records of patients. Ensemble learning technique, bagging, using random forest as the training algorithm is adopted in this research work. The base classifiers include KNN, DT, NB and LR. The predictions of these models are aggregated using votingclassifier() to produce a consensus classifier (soft vote model), which is trained and used to analyze the test dataset and produce results.

The models are further evaluated using cross validation techniques to assess their performances using the following cross validation parameters: accuracy, precision, recall, f1-score and area under the curve (AUC). The soft vote Classifier demonstrates exceptional performance, with metrics indicating near-perfect predictive capabilities. The model achieves an accuracy of 0.99788, precision of 0.997879, recall of 0.997885, F1-Score of 0.997880, and a ROC-AUC score of 0.99996. The confusion matrix further supports these results, showing minimal misclassifications and indicating that the majority of instances are correctly classified, using TP, FP, TN and FN rates. The AUC-ROC scores for each class include—normal (1.0), DOS (0.99995), PROBE (0.99995),

R2L (1.0), and U2R (0.99991). The generated values of these parameters form part of this chapter. They will be used to compare with the results of similar works in literature.

4.2 System evaluation

The primary objective of this thesis is to leverage machine learning models to efficiently detect and classify network attacks. The system architecture begins with the NSL-KDD dataset, followed by data extraction and initial analysis. Manual experimentation is conducted to explore features, and data preparation includes label encoding, normalization, and PCA for dimensionality reduction. Five machine learning models: K-Nearest Neighbors (KNN), Random Forest Classifier (RF), Decision Tree (DT) Classifier, Naive Bayes (NB) Classifier and Logistic regression (LR) are trained on the characteristics of normal applications, and to use that knowledge to classify malware threat applications from normal ones. Each model was evaluated based on its ability to identify patterns and anomalies in network traffic. The ultimate goal was to create an ensemble model using these classifiers with soft voting, allowing each model to contribute to the final decision by averaging their predicted probabilities. More predictive power was assigned to the best-performing models within the ensemble to enhance overall accuracy and efficiency. The predicted outputs (results) were compared with the expected labels using confusion matrix, with metrics such as TP, TN, FP and FN, being a supervised learning problem.

The models were assessed using cross validation metrics such as accuracy, precision, recall, F1-Score, and area under the (receiver operating characteristics) curve (AUC-ROC), with particular attention to AUC-ROC scores for each attack class. Additionally, Principal Component Analysis (PCA) was applied for dimensionality reduction, and SMOTE was used to balance the dataset, ensuring robust and unbiased model performance. This comprehensive approach aims to develop a highly accurate and

reliable machine learning model for detecting and classifying network attacks, thereby enhancing cyber security measures, and protecting electronic health records system for improved morbidity and mortality rates of healthcare system. The framework will assist decision makers in planning and funding electronic healthcare system, which has relative advantages of electronic health records (EHR) over the traditional paper system still mostly used in Nigerian health systems.

Some advantages of EHS over the paper-based system include

i. Easy collaboration and data sharing within and between hospital staff
ii. Mobility
iii. Cost reduction on ICT services
iv. "Pay for what you use", rather than owning expensive infrastructure and equipment
v. Scalability
vi. Business continuity
vii. Flexibility
viii. Strategic values.

4.3 Results

The results are derived from classified test dataset consisting of actual and predicted data classes such as: NORMAL, DOS, PROBE, U2R and R2L. As a supervised learning problem, the actual values of this test dataset are known (Github Inc., 2020). Subsequently, PCA was applied with the chosen number of components, and the resulting principal components were used to create a new DataFrame with records and columns as depicted in Table 4.1. Given the new dataFrame, the original features have been replaced with the derived principal component PC1 TO PC90 including the labels.

The predicted labels of the soft vote model will be matched with the expected (or actual or known) class types for analysis by confusion matrix.

The extract presented in Table 4.1 has only twenty records out of 125,973 records, for want of space. However, some misclassifications are found such as: in lines 290 Normal label was misclassified as U2R; in line 3530, Normal label was misclassified as PROBE; in line 3770, Normal label was misclassified as DOS. Similarly, in line 4132, Normal label was misclassified as DOS respectively. Observe that extracts from the results sheet are presented in sets of eight columns and ten rows each with labels as "class" and "predicted" as depicted in the last group of columns in the first ten (10) rows or records in Table 4.1. The last group of columns in each set contains the class or actual and predicted labels. This is a convenient way of presenting the ninety columns, of the set of records extracted, including label column.

Table 4.1: Extracts of Results Presented in Groups of Records and Columns

SN	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
5188	-1.43341	0.596405	-0.44467	-1.14996	2.509284	1.120042	-1.56106	0.849499
110012	1.208092	0.588724	0.065598	-2.68224	3.190255	1.539098	-2.00343	1.376987
289664	4.76409	-4.17956	-0.4102	0.516971	0.221604	0.315511	-0.40474	0.2378
11733	-3.80554	0.220176	-6.16195	0.148521	-1.61079	0.749816	-0.13229	-0.60043
112761	4.891938	-4.20786	-0.46565	0.473168	0.239561	0.282929	-0.35332	0.18615
91777	-1.45234	-0.159	1.378551	-0.01186	0.491553	-0.4992	0.621893	-0.73715
241358	-2.60999	0.093091	0.456637	1.193181	0.238186	-1.14077	0.08406	1.828881
18724	-1.99626	-0.54995	-6.02778	-4.54614	0.817659	0.947084	1.188159	-4.17533
37521	-0.59901	0.710094	0.353775	-0.04459	-0.22459	0.124985	0.222975	-0.80542

PC9	PC10	PC11	PC12	PC13	PC14	PC15	PC16
0.064919	-0.36444	0.981542	-0.09704	0.566431	-1.09105	-0.30802	-1.8841
0.27099	-1.39076	0.896866	-0.73486	0.056956	0.346718	0.048024	0.775836
0.55475	-0.44988	-0.40682	-0.66058	0.008154	0.413063	0.305829	0.972755
0.506227	-1.29716	0.044736	0.301274	-0.06884	-0.01851	-0.0492	-0.14783
0.370871	-0.41412	-0.39389	-0.65759	0.039806	0.403971	0.2998	0.940998
0.197757	-1.81841	-1.07062	-2.35901	1.654002	-0.73425	-0.37695	-1.69693
-0.35664	1.213269	0.734716	-0.20698	0.142829	-0.01658	0.121971	0.250291
-3.16717	8.379447	-5.16169	-2.3807	0.301356	0.105053	0.057338	0.442796
0.485391	-2.60376	-0.95341	-2.17076	1.730463	-0.94782	-0.43489	-1.98836

PC17	PC18	PC19	PC20	PC21	PC22	PC23	PC24
-0.10106	-0.21132	0.521873	-1.12022	0.028723	-0.14176	-0.23805	-0.04097
0.451745	-0.62111	-0.14419	-0.1341	0.058582	0.01413	0.108368	-0.06636
0.277142	-0.23485	-0.17009	-0.07654	0.10105	-0.00301	-0.04474	-0.08857
-0.21819	-0.02033	0.109183	0.029106	0.036225	0.056989	0.027328	-0.08436
0.250505	-0.23763	-0.15985	-0.08607	0.11368	0.012623	-0.04428	-0.12272
-0.06101	-0.68544	-0.83662	4.697652	-1.31443	-0.97666	-0.29589	0.162373
-0.09655	0.033966	-0.03281	0.147765	-0.03313	0.142869	0.183885	-0.06653
0.67874	0.099866	-0.19613	0.368079	-0.18241	-0.07104	-0.03548	-0.16956
-0.10621	-0.77626	-0.86778	5.51174	-1.51989	-1.06084	-0.31204	0.106846

PC25	PC26	PC27	PC28	PC29	PC30	PC31	PC32
-0.24601	0.274055	-0.02524	-0.31038	-0.5044	0.464046	0.248291	0.274289
0.006556	0.029092	0.01416	-0.03842	-0.03363	0.093504	-0.02029	0.013292
-0.10364	0.024803	-0.00698	-0.03687	-0.07626	0.071422	0.110993	0.108795
-0.03596	0.087156	0.024951	-0.01998	-0.07275	0.032027	0.064439	0.056664
-0.13227	0.02922	-0.01486	-0.03268	-0.0861	0.095472	0.118817	0.132343
0.637566	0.206886	-0.00399	0.119879	0.321275	-0.32297	0.428508	-0.64849
0.171206	0.108496	0.065695	-0.08492	-0.01409	0.028414	-0.06793	-0.19687
-0.07961	0.190489	-0.01736	-0.32991	-0.43887	0.237063	0.279467	0.299154
0.716157	0.290157	-0.0125	0.08961	0.370457	-0.38355	0.43724	-0.65779

PC33	PC34	PC35	PC36	PC37	PC38	PC39	PC40
0.168795	0.196434	0.180756	0.124714	-0.03128	0.276966	-0.21958	0.164281
-0.03607	-0.02418	-0.01627	-0.01714	0.014987	0.003005	0.017857	0.001184
-0.02033	0.052084	0.02941	-0.0094	0.009499	0.022714	-0.03871	-0.01296
-0.03152	0.056972	0.015423	0.008694	0.02103	0.046384	-0.00665	0.026593
-0.02636	0.055578	0.021791	-0.00643	0.008927	0.028292	-0.04374	-0.00978
-0.09869	-0.27961	-0.02805	0.31	0.020309	-0.23659	0.029365	-0.26532
-0.09783	-0.16139	-0.04988	-0.05709	0.059904	-0.14054	0.056844	-0.05366
-0.02813	0.127981	0.308976	0.054738	-0.03959	0.247038	-0.0705	0.01809
-0.14986	-0.31848	-0.0607	0.353178	0.040466	-0.20971	0.032624	-0.28506

PC41	PC42	PC43	PC44	PC45	PC46	PC47	PC48
0.271315	0.298299	-0.21565	0.161457	-0.11932	0.092712	0.019399	0.01545
0.01087	0.000783	0.083221	-0.00975	-0.01198	0.015618	0.004404	0.00741
-0.0519	0.014324	0.176439	0.007044	-0.01006	-0.0017	-0.0012	0.006924
0.072205	0.069764	0.052549	-0.01969	0.025628	0.005628	-0.00473	0.003232
-0.03571	0.001406	0.18288	-0.00516	-0.01508	0.000759	-0.00132	0.007851
0.170807	-0.025	0.047128	0.05009	-0.11704	-0.08584	0.006675	0.007316
0.010414	-0.0402	0.038305	-0.02805	-0.02097	0.036438	0.009034	-0.00609
0.126591	0.353941	-0.34637	0.132045	0.196084	-0.2348	-0.13636	-0.00628
0.232716	0.003749	0.022029	0.034817	-0.11401	-0.09436	0.010222	0.001772

PC49	PC50	PC51	PC52	PC53	PC54	PC55	PC56
0.013883	-0.01929	-0.09379	0.033215	0.048525	-0.0909	-0.05882	-0.10556
0.002983	-0.0018	0.006215	0.014866	-0.01186	-0.01966	-0.01573	-0.01319
-0.00822	-0.00056	-0.01273	-0.00421	0.021861	-0.00565	0.002936	0.046428
0.00501	-0.00183	-0.00391	0.025313	-0.01366	-0.01784	0.004542	-0.00928
-0.00818	-0.00074	-0.0133	0.005196	0.025573	-0.00801	0.004914	0.039688
-0.05436	-0.0034	-0.00966	-0.0073	-0.24796	0.175255	-0.0329	-0.18207
0.001456	0.00142	0.034947	-0.00052	-0.02172	-0.0286	-0.00334	-0.02392
0.013532	0.032552	-0.03622	-0.14356	-0.03896	-0.00622	-0.03715	0.006501
-0.04338	-0.00754	-0.00837	0.043197	-0.31472	0.191177	-0.02769	-0.20914

PC57	PC58	PC59	PC60	PC61	PC62	PC63	PC64
-0.20741	-0.00829	-0.19157	-0.03852	-0.02104	-0.05698	-0.06674	0.026465
0.002367	-0.01418	0.000238	-0.0067	-0.00202	-0.0009	0.010973	0.003148
0.049624	-0.01655	0.013147	-0.00464	-0.01317	-0.03436	0.027717	0.037506
-0.00951	-0.01931	-0.02858	-0.01658	0.002502	0.014562	0.004828	0.017342
0.045607	-0.02449	0.011386	-0.00897	-0.00885	-0.02353	0.03172	0.038085
0.008792	-0.00169	0.052061	0.03718	-0.05403	0.061843	0.022992	-0.02295
0.020733	0.003774	0.004646	0.008505	0.010498	0.0128	0.020946	-0.00428
-0.00617	0.034188	-0.00812	-0.00269	-0.03655	-0.13021	-0.02872	0.019974
-0.00162	0.003402	0.026731	0.060524	-0.05699	0.105273	0.012538	-0.0362

PC65	PC66	PC67	PC68	PC69	PC70	PC71	PC72
-0.03984	0.131631	-0.431	-0.09089	0.145141	-0.04534	-0.02046	-0.05586
0.039914	0.005963	0.04663	0.017481	-0.04266	0.025823	-0.00114	0.009557
0.086355	-0.02088	0.115364	0.046035	-0.08443	0.030532	0.0008	0.000616
0.070286	0.009329	-0.01818	0.005931	0.017778	0.007867	-0.01791	-0.00064
0.093789	-0.0201	0.118175	0.050006	-0.08391	0.034083	0.001436	0.002349
0.134086	-0.19176	0.102926	0.074559	-0.05577	-0.00405	0.007944	0.012066
0.022551	-0.01414	0.021962	0.00581	0.013413	0.00299	-0.01391	0.005417
-0.003	-0.0029	-0.01204	-0.0191	-0.06942	-0.00806	-0.01469	-0.00198
0.114101	-0.18693	0.076515	0.073868	-0.01479	-0.01284	0.00777	0.007949

PC73	PC74	PC75	PC76	PC77	PC78	PC79	PC80
-0.00547	0.232443	-0.01008	0.022562	-0.32387	-0.23442	1.0158	0.187921
-0.00674	-0.05749	0.000146	0.004887	0.102742	0.022052	-0.25433	-0.10798
-0.00571	-0.11614	-0.00223	0.01973	0.178398	0.029836	-0.51407	-0.15209
0.016878	0.00158	0.000748	-0.01906	-0.02431	0.004828	-0.01447	-0.0347
-0.00636	-0.11437	0.000229	0.015693	0.172527	0.034322	-0.52383	-0.15292
0.071227	-0.00677	-0.00635	-0.05522	-0.09755	0.019701	-0.12645	0.03126
0.013893	-0.00031	-0.00818	-0.02002	-0.03588	0.014368	0.013182	0.000854
-0.01928	-0.05049	-0.00889	0.037035	0.107675	-0.02587	-0.10588	-0.05275
0.099113	0.030275	-0.01287	-0.07317	-0.18672	0.016234	0.001142	0.066488

PC81	PC82	PC83	PC84	PC85	PC86	PC87	PC88	class	Predicted_
0.042053	-0.24301	-0.06745	0.331163	-0.51075	0.21299	-0.04361	1.33562	normal	normal
-0.05209	-0.6451	0.110284	0.252581	-0.59601	0.269111	0.228987	1.851491	DOS	DOS
-0.05794	-0.81644	0.267012	0.230452	-1.01476	0.258604	0.561143	-0.09836	U2R	U2R
-0.0068	-0.04942	-0.01854	0.010824	-0.13111	0.027481	-0.0334	-0.12842	DOS	DOS
-0.06084	-0.81933	0.265786	0.247251	-1.06038	0.265955	0.583791	-0.2435	normal	normal
-0.01655	-0.05213	-0.04446	-0.04408	0.168358	-0.02896	0.016122	-0.00182	normal	normal
-0.00846	-0.05918	0.0466	0.036568	-0.05721	-0.00211	0.010228	0.008848	R2L	R2L
-0.03255	-0.58336	0.127527	0.312746	-0.14551	0.064929	0.105298	0.175536	DOS	DOS
0.004521	0.308484	-0.03548	-0.19775	0.293882	-0.10385	-0.08376	0.216704	PROBE	PROBE

Rows 11 to 20 begin here.

93402	-2.00847	-0.32392	1.344167	-0.87271	0.447848	1.599749	6.407752	0.151383
313978	-0.38632	0.051189	-0.43239	-3.18599	3.636681	1.018746	-1.24182	0.588119
103367	-1.24601	-0.03116	2.347593	1.742339	1.105891	1.548678	-1.07418	-1.5806
165945	4.917583	-4.20257	-0.43597	0.513373	0.215858	0.327084	-0.41618	0.242556
1011	-3.76339	0.337634	-6.36657	0.88681	-1.86574	1.122255	-0.72474	-0.09987
124722	-2.23267	0.229103	0.52391	0.947295	0.169123	-1.04238	0.053781	1.841169
323856	-2.49873	0.16529	0.395036	1.059523	0.269006	-1.2546	0.221194	1.788027
80180	5.283799	3.648818	-0.69371	-0.54698	-1.47147	-1.30429	1.479508	-0.43894
213785	4.961381	-4.19194	-0.42442	0.542991	0.203266	0.31619	-0.42019	0.275789
58917	-2.39428	0.447401	-4.31487	0.243806	-1.26502	1.160017	-0.92368	0.214541

0.35398	-1.47465	-1.28346	0.110311	-0.13034	-1.33129	-3.19277	0.916946
0.638574	-0.3355	0.333779	-0.69904	-0.02105	0.520495	0.258204	1.033567
-0.78028	-0.34204	-1.09888	0.490023	0.244657	-0.01601	0.099692	0.035288
0.408203	-0.43496	-0.37666	-0.6552	0.043991	0.397146	0.298004	0.943952
0.369163	-1.36886	0.56621	0.408148	-0.03373	-0.01574	0.001392	-0.03282
-0.46042	0.956657	0.574489	-0.28286	0.054551	-0.07059	0.023165	0.119043
-0.66519	1.028897	0.765497	-0.27148	0.158634	0.017976	0.06354	0.176903
-1.67514	0.87093	1.259652	3.035217	-0.34689	0.340888	0.179362	0.233459
0.36633	-0.50455	-0.33665	-0.6536	0.055098	0.393583	0.288392	0.927868
-0.02462	-0.43128	-0.00513	-0.00431	-0.04612	-0.20231	-0.19726	-0.27521

0.153092	-0.23475	-0.98576	-0.42282	0.263631	0.060319	-1.11357	-1.25695
0.145004	-0.17851	-0.11793	-0.09077	0.080737	0.097974	0.082656	-0.16237
0.045358	0.085788	-0.18157	-0.14968	0.03723	-0.42814	-0.60339	0.272412
0.26282	-0.25309	-0.16593	-0.07795	0.106687	0.005631	-0.04429	-0.11247
-0.21975	0.075456	0.018066	0.064638	0.028902	0.061766	-0.00929	-0.10038
0.012262	-0.05259	0.000687	0.184746	-0.03645	0.141669	0.200046	-0.06361
-0.04569	-0.00608	-0.02805	0.049776	0.002043	0.160566	0.211681	-0.06683
0.093463	-2.2807	0.008045	1.28981	-0.99265	0.832372	-0.51714	-2.44368
-1.54025	0.131775	-1.16587	-2.30575	0.121295	-0.86996	0.712183	-0.14682
0.268222	-0.27245	-0.16203	-0.09826	0.113133	0.00763	-0.0108	-0.11549
-0.08556	-0.03612	-0.02118	-0.01622	0.01284	-0.01876	0.013612	-0.01809
0.146527	-0.17978	0.114311	-0.19166	0.111577	-0.0317	0.029222	0.091286
0.205514	0.457891	0.056146	0.189711	-0.30063	0.398788	-0.16563	0.101584
-0.02374	0.052512	0.021414	-0.00677	0.00961	0.025791	-0.04316	-0.00977
-0.03166	0.064361	0.019374	0.011485	-0.00382	0.07905	-0.01994	0.024745
-0.06263	-0.12352	-0.03014	-0.0697	0.057388	-0.12623	0.064509	-0.03331
-0.01764	-0.05478	-0.02815	-0.04879	0.008547	-0.27139	0.064239	-0.02573
-0.02394	0.068375	-0.00623	-0.07852	0.065632	-0.27626	0.076251	-0.00815
-0.02457	0.055729	-0.02492	0.188487	0.344249	-0.4265	0.204472	-0.36668
-0.12854	0.078628	-0.01588	-0.03625	-0.03125	0.090752	-0.14736	0.085293
-0.08251	0.023081	-0.02282	-0.01289	-0.03774	0.015703	0.044594	0.14307
0.146737	0.097958	0.072015	-0.07619	-0.04275	0.061436	-0.06552	-0.13375
0.162442	0.104216	0.074473	-0.09043	-0.02655	0.035923	-0.06256	-0.18076
-0.55583	-3.34314	-3.53173	0.512504	-0.45322	-0.80259	0.860898	0.834917
-0.12668	0.028635	-0.01222	-0.03314	-0.08414	0.097722	0.115323	0.127821
-0.02785	0.038211	0.072612	-0.0034	-0.09908	0.098686	0.000984	0.006457

-0.21911	-0.1208	-0.12267	-0.19848	0.137261	-0.00879	0.054087	-0.03533
0.005398	-0.03082	0.091014	-0.03039	-0.00612	0.013275	-0.01447	0.006794
-0.13109	0.032458	-0.00696	0.123562	-0.02103	-0.08404	-0.01186	-0.00047
-0.03749	0.003809	0.176913	-0.0023	-0.01445	8.58E-05	-0.00055	0.007613
0.047531	0.049819	0.03394	-0.02055	0.023987	0.003689	-0.00475	-0.00099
0.004018	-0.01919	0.043685	-0.01523	-0.02473	0.048657	0.015654	-0.00861
0.004515	-0.03169	0.090561	-0.03491	-0.01244	0.035188	0.003218	-0.00514
-4.11546	2.286793	-0.79452	0.46845	-0.90705	-1.06202	0.106818	0.06151
-0.03348	0.002546	0.180782	-0.00472	-0.0156	0.001054	-0.00058	0.008003
0.048156	0.059805	-0.0358	0.025884	0.004993	0.01582	0.028077	-0.00075

0.037657	-0.00362	0.381661	0.152621	-0.0248	-0.20366	-0.1477	-0.01733
-0.00724	-0.00034	0.002899	0.013572	-0.00729	-0.02671	-0.0047	0.0429
-0.00015	0.00199	-0.10171	0.030897	0.005475	0.078736	0.002423	0.117059
-0.00758	-0.00072	-0.01263	0.003918	0.024816	-0.00709	0.003982	0.038784
0.008924	-0.00132	-0.00443	0.021693	0.010594	-0.01045	0.017029	-0.00907
0.004519	0.002258	0.035904	-0.01284	-0.01069	-0.03319	-0.02008	-0.06247
-0.00048	0.002131	0.032971	-0.00513	-0.02122	-0.03591	0.000983	-0.01544
-0.00977	0.074983	-0.24582	-0.51039	0.514885	0.062606	0.184317	0.926697
-0.00769	-0.00078	-0.01277	0.005792	0.025346	-0.0083	0.004302	0.036923
0.0057	-0.00593	0.002877	0.005908	0.015957	-0.01358	-0.03957	-0.0483

-0.06009	-0.02573	-0.15096	-0.13476	0.173026	0.183017	0.18029	-0.02283
0.025217	-0.03159	-0.00113	-0.01325	-0.00474	0.002313	0.034295	0.021302
-0.07748	-0.01535	-0.00569	0.015461	-0.0455	-0.00998	-0.03547	0.01522
0.045992	-0.02185	0.01106	-0.0075	-0.01009	-0.02555	0.02961	0.037248
-0.00978	-0.01027	-0.02201	-0.01259	0.004296	0.011988	-0.00485	0.007953
0.022331	0.011637	0.016669	0.006118	0.009504	0.002145	0.011669	-0.013
0.017217	-0.00671	0.008083	-0.00452	0.015392	0.009748	0.027902	0.001388
4.790084	1.381317	2.174723	5.862019	-1.84729	2.715497	4.720275	-3.13074
0.043542	-0.02406	0.010331	-0.00919	-0.00868	-0.02345	0.030458	0.037634
-0.03103	0.013779	-0.01511	-0.0112	0.0024	-0.01357	-0.02411	-0.00469

-0.04117	0.32673	0.185074	0.078244	-0.13912	0.129997	0.013557	0.065301
0.089144	-0.00487	0.086259	0.030044	-0.03967	0.024926	-0.01046	0.008455
0.024208	0.017749	-0.02315	0.033763	-0.04966	0.015726	0.035364	-0.01856
0.088557	-0.01921	0.113988	0.047821	-0.08247	0.032677	0.001496	0.001561
0.00921	0.014574	-0.01343	-0.00321	0.013434	0.000731	-0.00789	0.000474
0.000387	-0.01399	0.022538	0.001467	-0.00207	0.006419	-0.00379	0.00828
0.042752	-0.0187	0.050749	0.012703	-0.00411	0.009233	-0.01652	0.012102
-2.81353	-2.9511	-0.48083	-0.15682	-0.50776	2.736617	0.304367	2.813478
0.091431	-0.01901	0.114039	0.048765	-0.08268	0.033812	0.001348	0.002249
0.017142	0.027395	-0.11264	-0.02454	0.036674	0.004331	-0.01253	-0.00844

-0.01261	0.038651	-0.28163	-0.01561	0.01736	0.388389	0.310078	-0.02983
0.0076	-0.07861	-0.00363	-0.00235	0.093733	0.029018	-0.35292	-0.11713
-0.01438	0.011729	0.017141	0.055743	0.116237	-0.03273	-0.20904	-0.00894
-0.00668	-0.11259	-7.22E-05	0.016875	0.171387	0.031702	-0.51221	-0.15051
0.003255	0.001342	0.002986	-0.00921	-0.01358	0.008073	-0.02173	-0.01084
-0.00076	-0.00218	-0.00675	-0.00367	0.000275	0.014535	0.021011	-0.00393
0.013505	-0.02175	-0.00887	-0.02367	-0.00996	0.037569	-0.07926	-0.0244
-0.81805	-4.28553	0.676899	-2.15346	-3.02031	-0.29966	2.457507	0.497404
-0.00681	-0.1121	0.000444	0.015691	0.170377	0.033434	-0.51425	-0.15186
-0.00941	0.037057	0.004995	-0.00193	-0.00671	-0.0523	0.216915	-0.02044

-0.03452	-0.16805	-2.01771	0.07269	-1.15699	-1.31997	-0.3013	0.648507	DOS	DOS
-0.03231	-0.58415	0.235833	0.311408	-0.78918	0.318157	0.247363	1.591094	U2R	U2R
0.072983	0.497691	-0.0434	-0.31022	0.35957	-0.15293	-0.18622	0.305336	DOS	DOS
-0.0592	-0.81376	0.262462	0.239644	-1.04296	0.264488	0.574554	-0.16736	PROBE	PROBE
0.004634	0.040834	-0.01029	-0.05564	-0.06109	0.003275	0.002845	-0.08558	DOS	DOS
-0.02859	-0.14757	-0.00588	0.056306	0.113992	-0.0383	0.031037	-0.11301	normal	normal
-0.02309	-0.14755	0.03957	0.054862	-0.15854	0.002336	0.038658	-0.36615	U2R	U2R
0.028088	1.125286	-0.06559	0.115585	1.159934	-0.1756	-0.32748	1.88739	normal	normal
-0.06111	-0.82355	0.258799	0.24499	-1.05705	0.266951	0.578012	-0.19794	R2L	R2L
-0.03331	-0.27421	-0.14254	0.038104	0.17542	0.019401	-0.09956	0.567177	normal	normal

To identify some errors, results from record number 285 to 295 are presented

13945	-2.10323	0.177106	2.059507	2.392953	1.219139	1.178649	-0.87307
234566	-2.63189	0.10808	0.392037	1.252919	0.210205	-1.1163	0.047431
113719	-2.60087	0.087072	0.460003	1.157997	0.2432	-1.14781	0.101184
79142	-2.36426	-0.20152	0.872232	-0.58232	0.227524	-2.07327	2.230025
58257	5.23576	3.711878	-0.57757	-0.24909	-0.97075	-0.98179	1.21632
111248	4.927342	3.622376	-0.63448	-0.51914	-1.55735	-1.42311	1.549533
21558	-2.09045	-0.31265	0.589255	-1.14842	-0.00693	-1.90453	2.28428
171535	-2.55955	0.040225	0.581654	1.023269	0.301822	-1.21703	0.201799
306268	4.909196	-4.62792	-0.20931	1.158004	0.191078	0.472318	-0.3153
297121	-2.09538	0.172523	2.066069	2.38133	1.213815	1.151924	-0.89299
221242	-1.27312	-0.1049	2.78226	-2.53718	-3.66848	1.172802	-0.84484

-1.50167	-0.60677	-0.18494	-0.50834	-0.15621	0.230596	0.059001	0.275354
1.860407	-0.37148	1.198877	0.780212	-0.19057	0.135332	-0.00753	0.131932
1.811502	-0.35504	1.215509	0.71424	-0.20833	0.145221	-0.01773	0.121028
-0.84962	0.2771	-1.13968	-0.8941	-0.0026	-0.24044	0.082192	-0.01317
-0.61331	-1.22902	0.826633	0.358493	-0.79506	-6.58068	1.298993	-0.82874
-0.28996	-1.10421	0.615719	1.295456	3.288967	-0.86332	0.175222	0.596372
-1.05749	0.448669	-1.61048	-1.12872	0.063899	-0.08816	-0.00583	-0.05512
1.732695	-0.31427	1.229674	0.611214	-0.24025	0.153375	-0.03291	0.100905
-1.02259	0.967952	-0.25511	0.33835	-1.8807	1.946544	-1.03999	0.542577
-1.52594	-0.60724	-0.19659	-0.51279	-0.15417	0.2319	0.054434	0.264388
0.690165	1.417019	0.466252	-0.91575	0.425308	0.484023	-0.13049	-0.01393

0.091237	-0.20453	0.556766	-0.23823	-0.44241	0.155492	-0.62161	-0.98656
0.276095	-0.09949	0.062849	-0.03937	0.15122	-0.03137	0.151212	0.181942
0.247267	-0.09576	0.030582	-0.02922	0.147256	-0.03284	0.143178	0.185126
0.01671	0.200535	-0.19264	0.405194	-0.85656	0.226949	0.106451	0.055293
-2.28912	-2.15418	1.590885	-0.0194	0.598815	-0.87003	-0.31953	-0.46069
1.148244	0.140647	-1.12464	-0.58695	0.741788	-1.13195	-0.82095	-2.19884
-0.14802	0.200764	-0.29888	0.437731	-0.61265	0.16812	0.068047	0.057062
0.195806	-0.0912	-0.02506	-0.01031	0.135388	-0.03368	0.129121	0.190595
0.316558	-0.76748	-0.79476	-0.50943	0.32329	0.374875	-0.53143	0.172729
-0.07364	0.16167	0.098624	0.058289	-0.08479	-0.01178	0.026981	-0.0708
-0.06621	0.17253	0.109628	0.066755	-0.08537	-0.01425	0.028553	-0.06837
-0.04723	-0.12218	0.018766	0.010938	0.056303	-0.14263	0.194158	0.030021
0.555277	0.927152	-0.63031	-0.29332	0.760726	1.804878	-1.672	-0.82098
1.956799	-0.36339	0.422216	1.741814	-0.27967	0.092756	-0.81731	-0.58228
-0.0403	-0.07249	0.04062	0.019906	0.046745	-0.10431	0.160923	0.024785
-0.05374	0.190226	0.130364	0.082443	-0.08615	-0.02126	0.03364	-0.06198
-2.23623	3.35377	0.323412	1.98849	0.075058	0.749068	0.353691	-2.84132
0.314358	-0.7769	-0.80009	-0.51165	0.308792	0.377526	-0.53295	0.162437
-0.00607	-0.09494	-0.00826	0.018335	-0.0039	0.111338	-0.02215	-0.02019

0.610441	0.332633	0.751681	0.150465	0.331131	-0.46933	0.660542	-0.27342
-0.17812	-0.09881	-0.15844	-0.04648	-0.05858	0.055503	-0.13444	0.056302
-0.19861	-0.09814	-0.16171	-0.05025	-0.0577	0.060671	-0.14139	0.057368
0.173275	0.056165	0.089285	0.023818	-0.01155	0.028444	0.052717	-0.01261
-1.00108	-0.80749	-1.16982	-0.8124	0.776668	-0.31765	-0.41004	-0.35623
-2.69629	1.220407	-2.80272	3.507898	0.293266	-0.62336	0.303718	-0.12946
0.139032	0.025541	0.078287	0.001978	-0.01408	0.044442	0.044779	-0.00308
-0.23492	-0.09621	-0.16661	-0.05628	-0.05591	0.070201	-0.15398	0.059336
-0.30498	-2.67633	-1.54451	-1.07761	-2.89343	-4.3108	2.830411	-0.82815
0.608178	0.328718	0.752463	0.147098	0.325594	-0.4702	0.658788	-0.27179
-0.01307	-0.06601	0.049236	-0.07588	-0.09182	-0.01823	-0.00161	-0.00098

0.163669	-0.18682	0.052112	-0.01185	0.187399	-0.02485	-0.13988	-0.01893
-0.05443	0.004837	-0.0416	0.039858	-0.02802	-0.02065	0.037053	0.00824
-0.05379	0.011076	-0.03961	0.039529	-0.02817	-0.02122	0.037004	0.008748
0.070066	0.167171	0.094075	-0.07026	-0.02283	0.018942	0.009735	-0.00353
-0.10743	0.537104	-0.68046	-0.52695	-0.54985	-0.18905	-0.36877	0.037238
-0.82223	-6.2511	9.118171	0.017024	-2.19525	-3.31419	-5.86201	1.00617
0.057556	0.193555	0.117575	-0.0392	-0.03041	0.010829	0.023682	-0.01539
-0.05227	0.023023	-0.03565	0.038215	-0.02829	-0.02189	0.036465	0.01017
2.365137	3.799115	6.593063	2.184718	3.623008	-2.30523	-1.3108	-0.23073
0.163228	-0.18849	0.051097	-0.01205	0.186976	-0.02449	-0.14012	-0.01844
-0.0606	-0.00589	-0.02981	0.055667	-0.05188	0.034172	-0.03363	-0.01693

0.015341	-0.01031	-0.00044	-0.16451	0.033341	0.019526	0.122754	0.017018
-0.00677	0.002057	0.001825	0.034804	-0.00274	-0.01721	-0.02851	-0.00168
-0.00617	0.001386	0.001427	0.03501	-0.00022	-0.02238	-0.02912	-0.00385
0.005886	0.012859	-0.00113	-0.01314	0.002859	0.0345	-0.0432	-0.01258
0.000183	0.044791	0.001912	0.032271	0.456743	-0.50593	0.497977	0.405524
0.034888	0.075328	0.157105	-0.25239	2.504697	-2.72507	2.15486	0.003834
0.000917	0.013517	-0.00191	-0.01261	0.023844	0.005256	-0.05136	-0.01735
-0.00475	9.69E-05	0.000712	0.035249	0.00377	-0.03114	-0.03007	-0.00748
0.145117	-0.19679	0.06525	-0.93432	-1.46739	0.92577	-3.87809	1.558001
0.015205	-0.01013	-0.00046	-0.16349	0.033836	0.019536	0.123223	0.017196
0.002829	0.000481	0.005213	0.007212	0.006692	0.01764	-0.0759	-0.03816

0.215973	-0.12505	-0.03975	-0.0095	0.018266	-0.06163	-0.01623	-0.03104
-0.02462	0.022715	0.00508	0.007001	0.009094	0.010935	0.011336	0.020106
-0.02431	0.020852	0.003501	0.004466	0.008428	0.010536	0.012985	0.021158
-0.15784	0.00086	-0.00595	-0.02258	-0.01783	0.015962	0.006623	-0.01079
0.147676	0.112903	0.089722	-0.30027	0.086224	0.237236	0.324512	0.148228
-2.0984	-4.08055	4.032051	-1.02729	-2.43555	-0.56662	-0.56089	-1.43806
-0.16899	-0.00526	-0.01532	-0.03709	-0.01981	0.01351	0.020922	-0.01034
-0.02425	0.01746	0.000552	2.54E-05	0.007067	0.009887	0.015818	0.023213
4.501136	4.111309	0.347039	1.086104	0.235119	-0.34403	-2.25829	2.565339
0.216246	-0.12522	-0.03956	-0.00954	0.018065	-0.06127	-0.01581	-0.03083
0.009136	-0.01151	-0.00674	0.001412	0.026583	0.046356	0.039762	0.027143

0.029604	0.107687	0.010271	-0.0263	0.067469	-0.06831	0.031695	0.043259
-0.00567	0.013955	-0.01375	0.027062	0.0054	0.00956	0.002725	-0.01178
-0.0041	0.023727	-0.0141	0.022144	0.006072	0.013328	0.003239	-0.01404
0.003787	0.004793	0.028036	-0.05203	-0.0011	0.00044	0.006126	0.01001
0.168124	-0.30662	0.004617	-0.16555	-0.0486	0.501406	-0.27609	0.048862
1.637229	-0.32131	-0.59973	1.021982	0.174058	-0.01427	0.634604	0.273802
0.008696	0.018084	0.028426	-0.05114	0.002396	0.00925	0.006003	0.007355
-0.0012	0.042397	-0.01513	0.013195	0.007527	0.020048	0.00432	-0.01814
0.468443	-0.7272	-0.40087	1.371332	0.514365	-0.28446	0.720522	-1.59168
0.029466	0.106557	0.010833	-0.02579	0.06748	-0.06873	0.031812	0.043446
0.015309	-0.01554	-0.00594	0.040799	0.030405	0.018224	0.018242	0.048773

-0.02351	-0.01668	0.009195	0.024688	0.065615	0.170761	-0.05813	-0.32255
0.006145	0.011207	-0.00268	-0.00773	-0.01754	-0.02951	0.016725	0.001814
0.005456	0.014228	-0.00033	-0.00819	-0.02016	-0.03608	0.014637	0.012816
0.00021	-0.01497	0.044569	0.005618	0.01538	-0.02191	-0.00778	0.147473
-0.06001	0.391849	0.179115	-0.13489	-0.30492	-1.10713	-0.10563	1.336603
0.744756	0.034422	0.754027	0.012426	-0.00794	-0.12008	0.876843	0.598302
0.000796	-0.00094	0.05096	0.002897	0.010661	-0.05169	0.005677	0.151055
0.004208	0.019576	0.004028	-0.00903	-0.02493	-0.04777	0.010583	0.033033
-0.49607	-0.41443	-0.5042	0.179255	0.479736	1.652984	-0.55448	-1.14812
-0.02346	-0.01686	0.009005	0.023992	0.065593	0.171059	-0.05726	-0.32243
0.006872	-0.00567	0.014121	0.015603	-0.02491	-0.05274	0.024598	-0.02981

-0.03079	0.079522	0.55564	-0.01073	-0.22497	0.341536	-0.11896	-0.12891	0.00708	normal	normal
0.002539	-0.00788	-0.05171	0.050727	0.03004	-0.04354	-0.00839	0.01935	-0.021	R2L	R2L
0.000224	-0.00866	-0.06074	0.04721	0.03738	-0.05932	-0.00189	0.00828	0.007375	DOS	DOS
0.0258	-0.01978	-0.1158	-0.12727	0.153186	-0.07586	0.053775	0.033606	-0.04925	DOS	DOS
0.716844	0.87198	6.61661	0.89756	-4.25718	-6.36996	1.39164	0.387119	1.80654	Normal	Normal
0.**307746**	0.118571	1.**903448**	-0.09946	0.**198819**	1.**496094**	-0.**11647**	-0.57824	1.**750175**	**Normal**	**U2R**
0.023333	-0.01152	-0.04006	-0.11395	0.11133	-0.14312	0.053542	-0.03307	0.006546	DOS	DOS
-0.00405	-0.01095	-0.08156	0.038541	0.05426	-0.08706	0.010178	-0.00855	0.051269	PROBE	PROBE
0.373686	0.360647	2.960279	-0.21389	-0.53043	1.661562	0.056836	-0.42663	-0.15791	U2R	U2R
-0.03059	0.079575	0.556624	-0.0144	-0.22765	0.34051	-0.12246	-0.12739	0.006204	U2R	U2R
0.06521	-0.00409	0.142647	0.229809	-0.08622	-0.2365	-0.04055	0.311207	-1.34343	R2L	R2L

Please observe that record number 290 (Bolden) was misclassified, with Normal predicted as U2R

4.4 Confusion Matrix

Confusion matrix depicts the relationship between the actual (or expected) values and the predicted values. The confusion matrices depicted in this work are multi-class confusion matrices.

K-Nearest Neighbor (KNN)

Figure 4.1 depicts the confusion matrix of KNN. These predictions were evaluated using a custom `evaluate model` function, with performance visualized through a confusion matrix and a detailed classification report, highlighting metrics such as precision, recall, and F1-score.

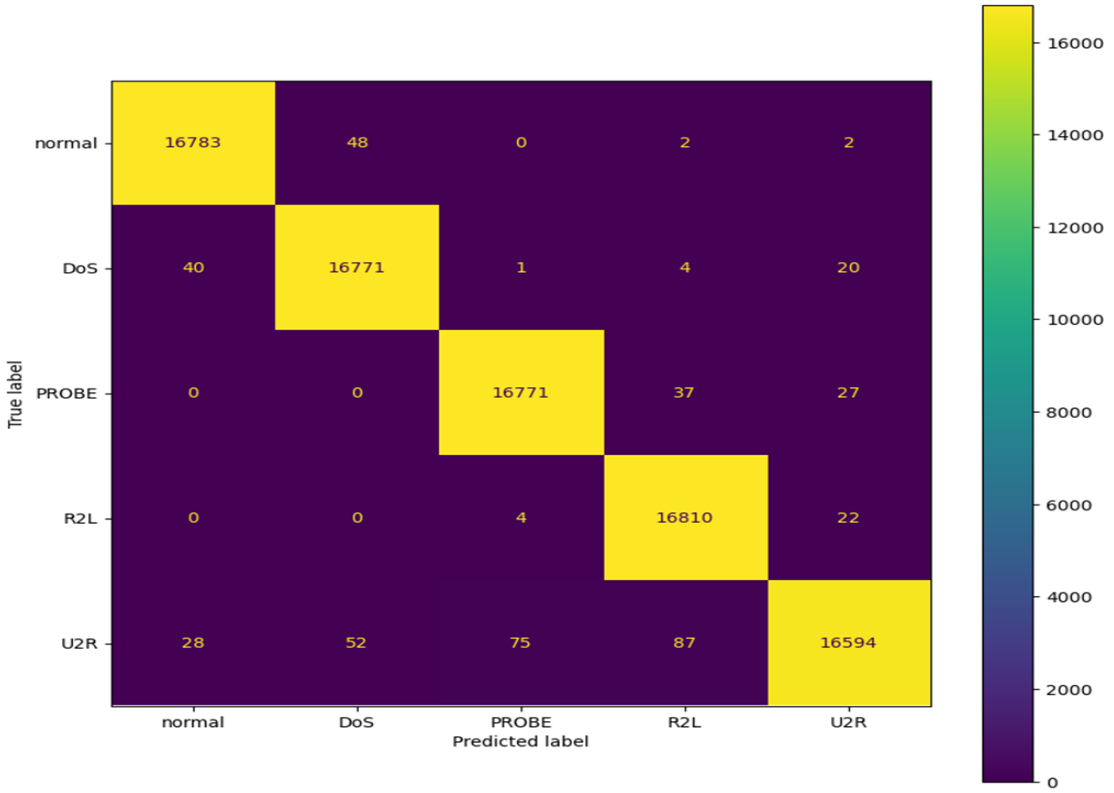


Figure 4.1: Confusion Matrix for KNN
Source; Researcher (20240

.

From Figure 4.1, KNN predicted 16783 TP records as normal; 16771 TP records as DOS, 16771 TP records as PROBE, 16610 TP records as R2L, and 16594 TP records as U2R. Observe that all TP values lie in the main diagonal of the confusion matrix. However, the model misclassified 48 normal records as DOS, 2 normal records as R2L and U2R respectively. KNN also misclassified 40 DOS records as normal, one DOS record as PROBE, 4 DOS records as R2L, and 20 DOS records as U2R. Similarly, the model misclassified 37 PROBE records as R2L and 27 PROBE records as U2R. It also misclassified 4 R2L records as PROBE and 22 U2R records as R2L. Finally, the model also misclassified 28 normal records as U2R, 52 DOS records as U2R, 75 PROBE records as U2R and 87 R2L records as U2R. . These predictions were evaluated using a custom `evaluate model` function, with performance visualized through cross validation and a detailed classification report, highlighting metrics such as accuracy,

precision, recall, F1-score, and area under the receiver operating curve AUC-ROC) curve, true positive rate (TPR), false positive rate (FPR), true negative rate (TNR), and false negative rate (FNR). In malware detection problem, TP refers to the number of normal executives predicted as normal; TN refers to the number of malicious records predicted as malware, FP refers to the number of true positive records predicted as malicious; while FN refers to malicious records predicted as normal records.

TPR is defined as the number of true positives divided by the total number of true positives divided by total number of malicious executive files, as depicted in Equation 4.1.

$$\text{TPR} = \frac{TP}{TP+ FN} \qquad \text{Equation 4.1}$$

FPR is defined as the number of false positive divided by total number of benign executive files, such as depicted in Equation 4.2.

$$\text{FPR} = \frac{FP}{FP+ TN} \qquad \text{Equation 4.2}$$

Precision is defined as true positive numbers divided by the sum of positive number and false positive numbers, as depicted in Equation 4.3

$$\text{Precision} = \frac{TP}{TP+ FP} \qquad \text{Equation 4.3}$$

Recall is defined as true positive numbers divided buy the sum of true positive numbers and false positive numbers, as depicted in Equation 4.4

$$\text{Recall} = \frac{TP}{TP+ FN} \qquad \text{Equation 4.4}$$

Accuracy is defined as the sum of the true positive numbers and true negative numbers divided by the total number of instances, as depicted in Equation

$$\text{Accuracy} = \frac{TP+ TN}{TP+ FP+ FN+ TN} \qquad \text{Equation 4.5}$$

The KNN model demonstrated strong performance with an accuracy of 0.98816, precision of 0.988158, recall of 0.988170, F1-Score of 0.988152, and an overall AUC-ROC score of 0.99740. Table 4.2 depicts KNN performance metrics

Table 4.2: KNN Performance Metrics

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
KNN	0.98816	0.988158	0.988170	0.988152	0.9991

Source: Researcher (2024)

The AUC-ROC scores for individual attack classes were also impressive, with normal at 0.9980, DOS at 0.9974, PROBE at 0.9982, R2L at 0.9986, and U2R at 0.9949.

The confusion matrix further highlights the model's effectiveness, showing minimal misclassifications across all classes. The AUC-ROC scores for individual attack classes were also impressive, with normal at 0.9980, DOS at 0.9974, PROBE at 0.9982, R2L at 0.9986, and U2R at 0.9949. Figure 4.2 depicts the AUC-ROC plot. Specifically, the AUC curve establishes the relationship between false negatives and false positives. While ROC curve is obtained by plotting the TPR against FPR (Santos *et al.,* 2013)

These results indicate that the KNN model is highly capable of accurately detecting and distinguishing between different types of network attacks, making it a reliable choice for network intrusion detection.

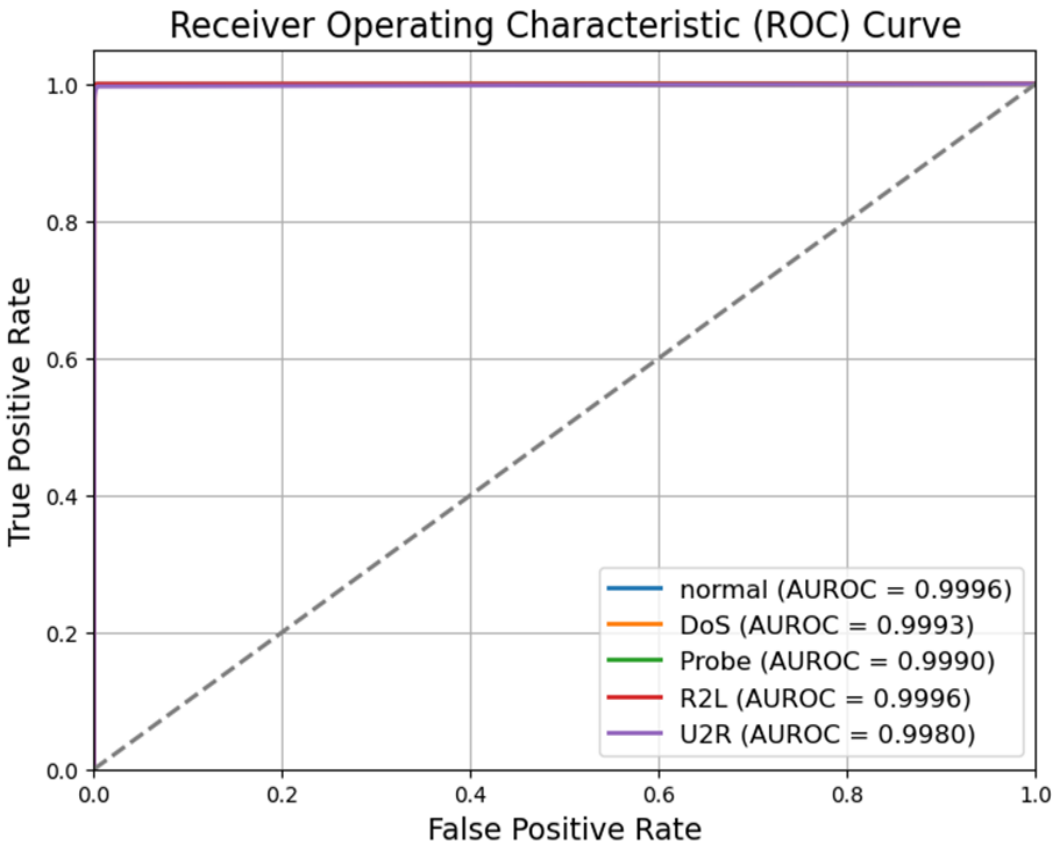


Figure 4.2: AUC-ROC Curve of KNN Classifier
Source: Researcher (2024)

Random Forest Classification

To train and evaluate a Random Forest model, it was initialized with 500 estimators to ensure robust and stable model performance by aggregating the predictions from multiple decision trees, thus reducing over fitting. Entropy was used as the criterion to measure the quality of splits, focusing on maximizing the information gain at each split. The model was trained on the training dataset (80%), and predictions were made on the test dataset (20%).

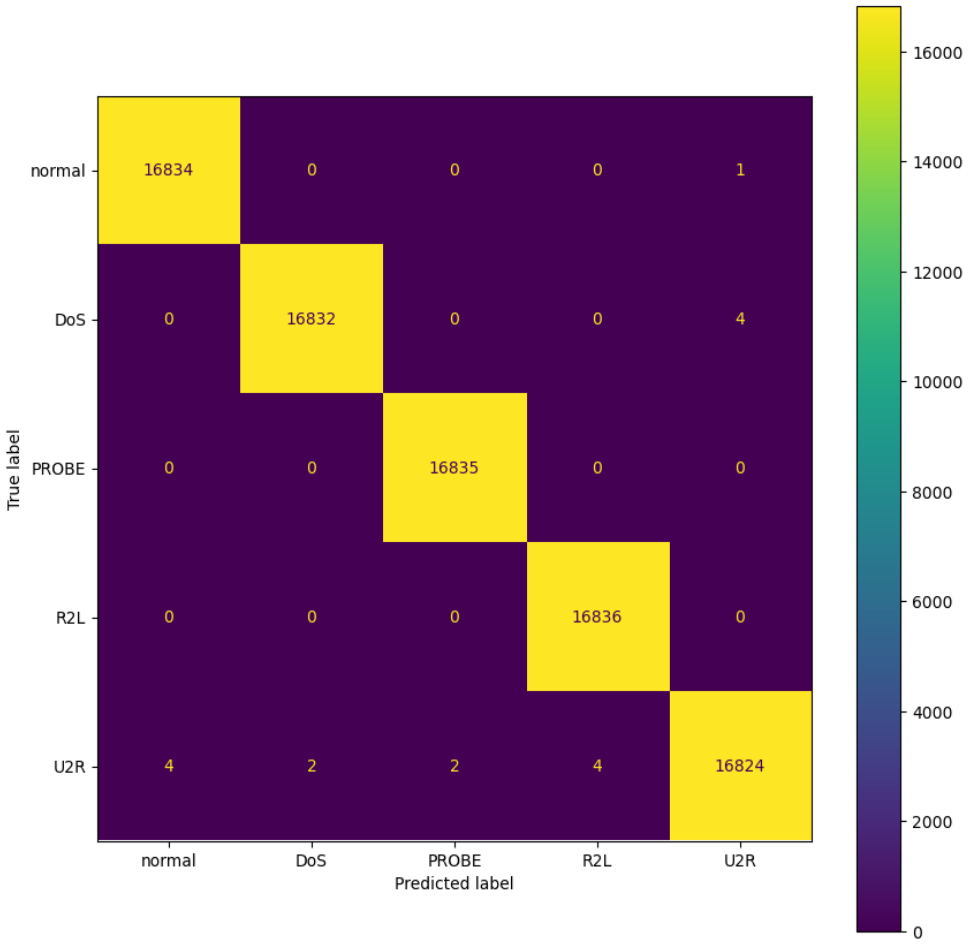


Figure 4.3: Confusion Matrix for Random Forest
Source: Researcher (2024)

Figure 4.3 shows that RF correctly predicted 16834 TP records as Normal, 16832 TP records as DOS, 16835 TP records as PROBE, 16836 TP records as R2L, and 16824 TP records as U2L with all the TP records displayed in the main diagonal of the confusion matrix. However, the model misclassified 1 normal record as U2R, 1 U2R record as DOS, 4 Normal records U2R, 2 DOS records as U2R, 2 PROBE records as U2R and 4 R2L records as U2R respectively. To assess the model's performance, a custom evaluate_model` function was used, and confusion matrix used to compare the predictions with the actual or expected values. Table 4.3 depicts the performance metrics for RF.

Table 4.3: Performance metrics for RF Classifier

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest Classifier	0.99932	0.999322	0.988170	0.999319	1.0

Source: Researcher (2024)

The Random Forest Classifier demonstrated excellent performance metrics, achieving an accuracy of 0.99932, precision of 0.999322, recall of 0.988170, F1-Score of 0.999319, and a perfect overall ROC-AUC score of 1.0. The AUC-ROC scores for each attack class were outstanding with 1.0 in all the classes, as depicted in Figure 4.4 Specifically, the AUC curve establishes the relationship between false negatives and false positives. While ROC curve is obtained by plotting the TPR against FPR (Santos *et al.,* 2013)

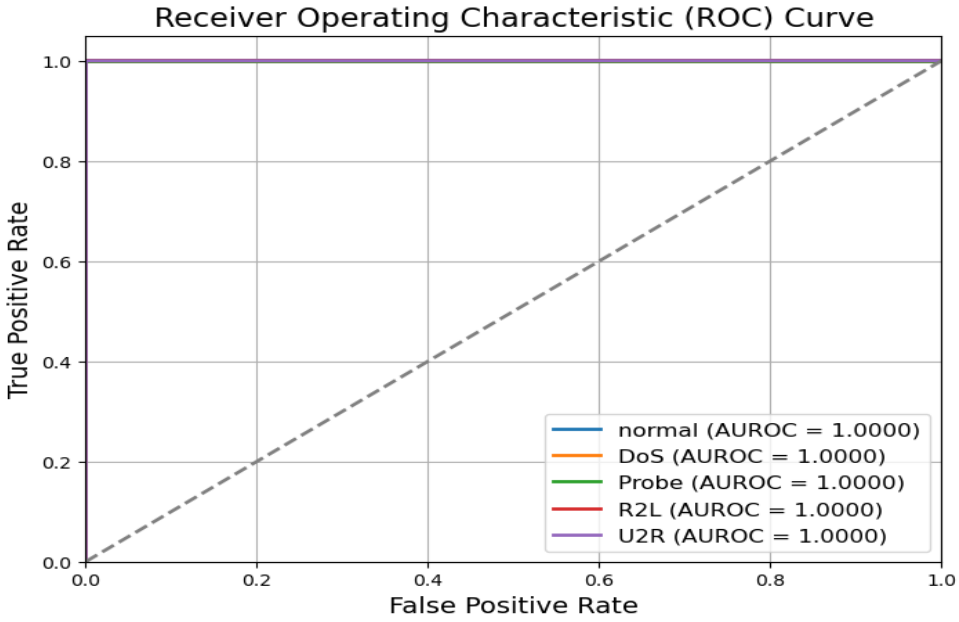


Figure 4.4: AUC-ROC operating curve of Random Forest

Source: Researcher (2024)

Despite these high scores, there were still some misclassifications evident in the confusion matrix. These misclassifications highlight that while the Random Forest

Classifier is highly effective at distinguishing between various types of network attacks, some errors do persist, particularly with the 'DOS' and 'U2R' attack classes.

Naïve Bayes Classifier

Training and evaluating a Naive Bayes model, I utilized the Gaussian Naive Bayes classifier due to its effectiveness in handling continuous data and its assumption of a normal distribution for the features. The model was trained on the training dataset (80%), and predictions were made on the validation dataset (20%).

Figure 4.5 depicts the confusion matrix of Gaussian Naïve Bayes classifier.

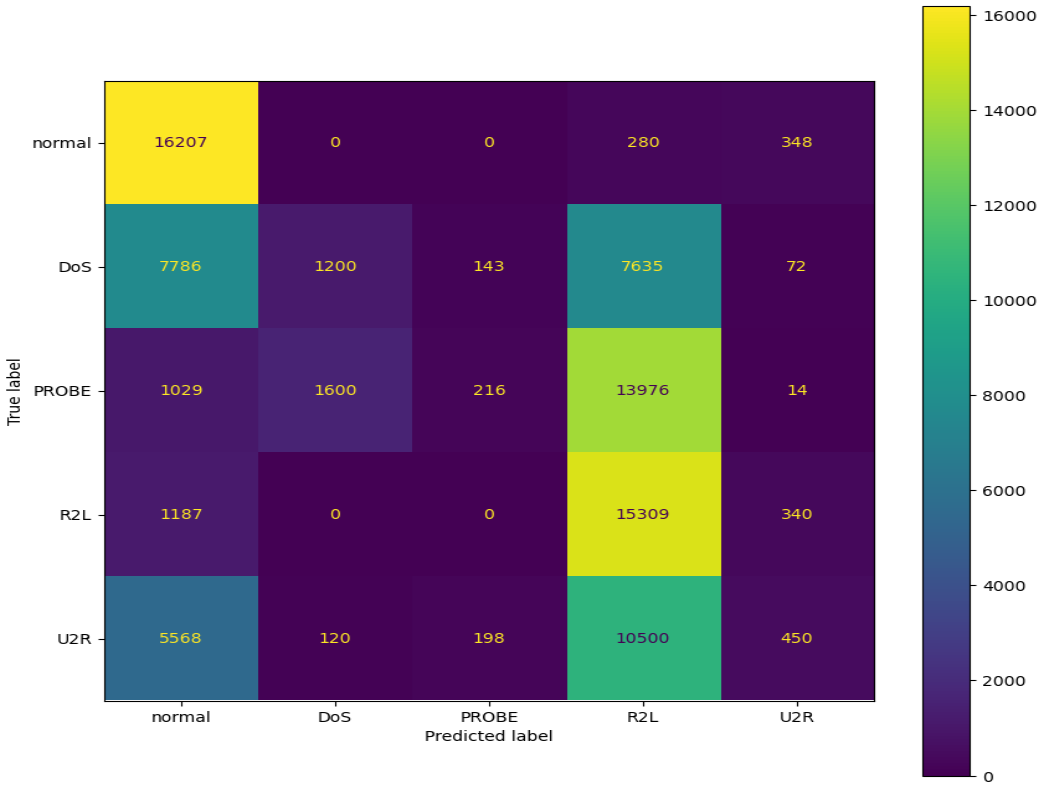


Figure 4.5: Confusion Matrix of Naïve Bayes classifier
Source: Researcher (2024)

Figure 4.5 depicts that Naïve Bayes classifier correctly predicted 16207 TP records of Normal type, 1200 TP records of DOS, 216 TP records of PROBE attack type, 15309 TP records of R2L attack type, and 450 TP records of U2R attack type. However, the model misclassified 7700 DOS records as Normal type, 1020 records of PROBE as

Normal, 1187 records of R2L type as Normal and 5568 records of U2R records as Normal. Similarly, the model misclassified 1000 records of PROBE type as DOS and 120 records of U2R as DOS. The model also misclassified 143 records of DOS as PROBE and 198 records of U2R as PROBE attack type. The model also misclassified 280 records of Normal type as R2L, 7635 records of DOS types as R2L, 13973 records of PROBE attack types as R2L, 1050 records of U2R records as R2L. Again, the model misclassified 148 records of Normal type as U2R, 72 records of DOS attack type as U2R, 14 records of PROBE attack type as U2R and 340 records of R2L attack type as U2R attack type.

To assess the model's performance, custom evaluate model function was used and the parameters derived are displayed in Table 4.4.

Table 4.4: Naïve Bayes Model Performance Evaluation results

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Naive Bayes Classifier	0.39624	0.410736	0.396438	0.268998	0.7419

The Naive Bayes Classifier displayed significantly lower performance metrics compared to other models, with an accuracy of 0.39624, precision of 0.410736, recall of 0.396438, F1-Score of 0.268998, and an overall AUC-ROC score of 0.7419. The AUC-ROC scores for individual attack classes were: class 0 (Normal) at 0.9032, class 1 (DoS) at 0.4746, class 2 (PROBE) at 0.6181, class 3 (R2L) at 0.8777, and class 4 (U2R) at 0.8448 as depicted in Figure 4.7. Figure 4.7 compares the false positive rate score with that true positive rate and derived the area under the curve depicted. Specifically, the AUC curve establishes the relationship between false negatives and

false positives. While ROC curve is obtained by plotting the TPR against FPR (Santos *et al.,* 2013).

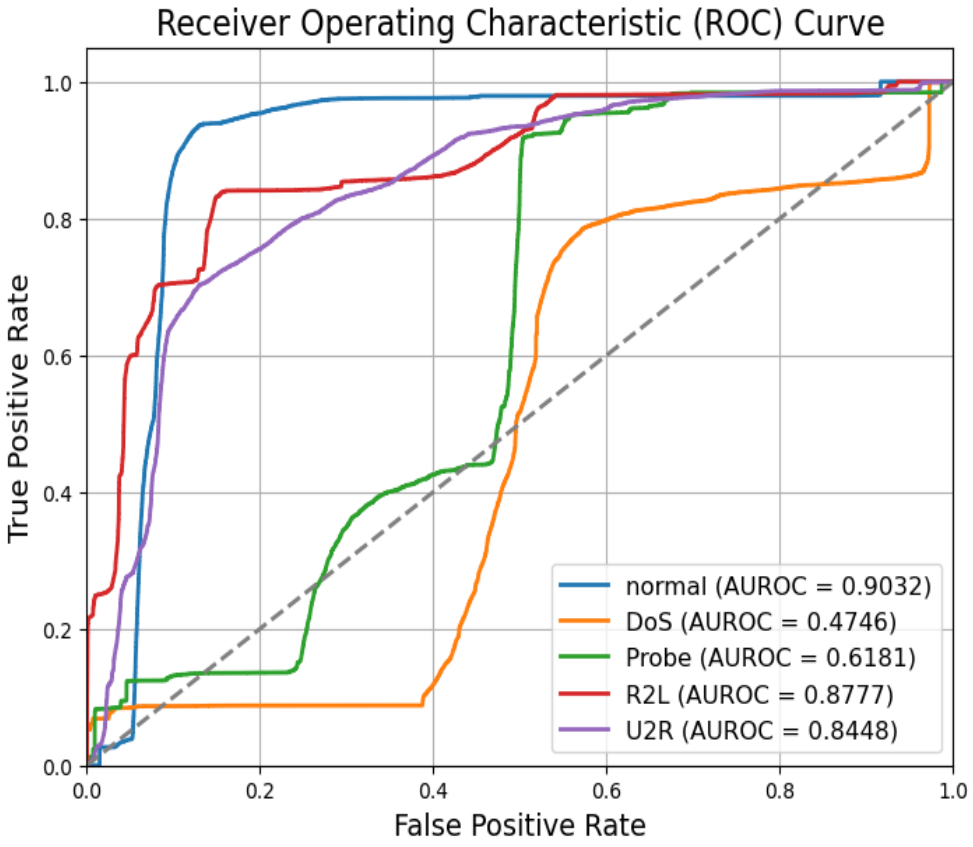


Figure 4.6: Comparison of FPR and TPR of Naïve Bayes Classifier
Source: Researcher (2024)

The AUC-ROC curves highlight the classifier's challenges in correctly identifying the different types of network attacks. The high number of misclassifications in Figure 4.5, especially among 'DoS', 'PROBE', and 'U2R' attacks, suggest that the Naive Bayes Classifier struggled with the complexity of the dataset and the variability within attack classes. These results indicate that, while Naive Bayes provides some insight, it may not be the best choice for accurately classifying network intrusions in this context.

Decision Tree Classifier

To train and evaluate a Decision Tree (DT) model, entropy was used to initialize the it as the criterion to measure the quality of splits, focusing on maximizing the information gain at each split. The model was trained on the training dataset (80%), and predictions were made on the test dataset (20%). Figure 4.8 depicts confusion matrix of DT.

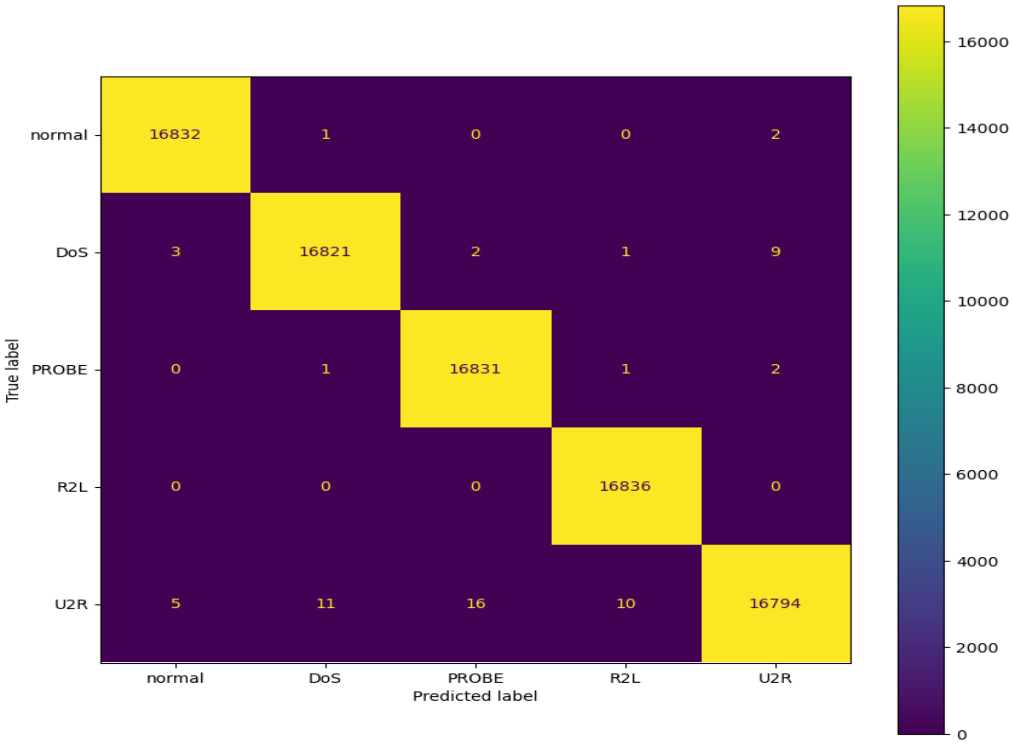


Figure 4.7: Confusion Matrix of Decision Tree Classifier.
Source: Researcher (2024)

The confusion matrix depicts 16832 TP records of Normal type, 16821 TP records of DOS, 16831 TP records of PROBE attack type, 16836 TP records of R2L attack type, and 10794 TP records of U2R attack type. The model, however, misclassified 3 records of DOS as Normal type and 5 records of U2R as Normal type. It also misclassified 1 record of Normal type as DOS attack type, 1 record of PROBE attack as DOS, and 11 records of U2R attack type as DOS attack type. Similarly, the model misclassified, the

model misclassified 2 records of DOS as PROBE attack type and 10 records of U2R attack as PROBE attack type. The model also misclassified 1 record of DOS attack type as R2L, 1 record of PROBE attack type as R2L, and 10 records of U2R attack type as R2L. In the same vein, the model misclassified 2 records of Normal type as U2R, 9 records of DOS attack type as U2R and 2 records of PROBE attack type as U2R. The low number of false positives and false negatives across all classes further underscores the model's reliability and accuracy in classifying network intrusion types, making it a robust choice for this task.

To assess the model's performance, k-fold cross validation, with k=10 was used, paving the way for splitting the dataset into 10 different sets, with 90% used to train and the remaining 10% used to assess the model. The resulting performance parameters are recorded in Table 4.5 as accuracy, precision, recall, F1-score and AUC of ROC

Table 4.5: Performance Metrics of Decision Tree

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Decision Tree Classifier	0.99804	0.410736	0.998042	0.998041	0.9988

The Decision Tree Classifier demonstrated impressive performance metrics, achieving an accuracy of 0.99804, precision of 0.410736, recall of 0.998042, F1-Score of 0.998041 and an overall UAC-ROC score of 0.9988 as depicted in Table 4.6. The AUC-ROC scores for individual attack classes were: class 0 (Normal) at 0.9999, class 1 (DoS) at 0.9995, class 2 (PROBE) at 0.9997, class 3 (R2L) at 0.9999, and class 4 (U2R) at 0.9987 as depicted in Figure 4.9. Specifically, the AUC curve establishes the relationship between false negatives and false positives. While ROC curve is obtained by plotting the TPR against FPR (Santos *et al.,* 2013).

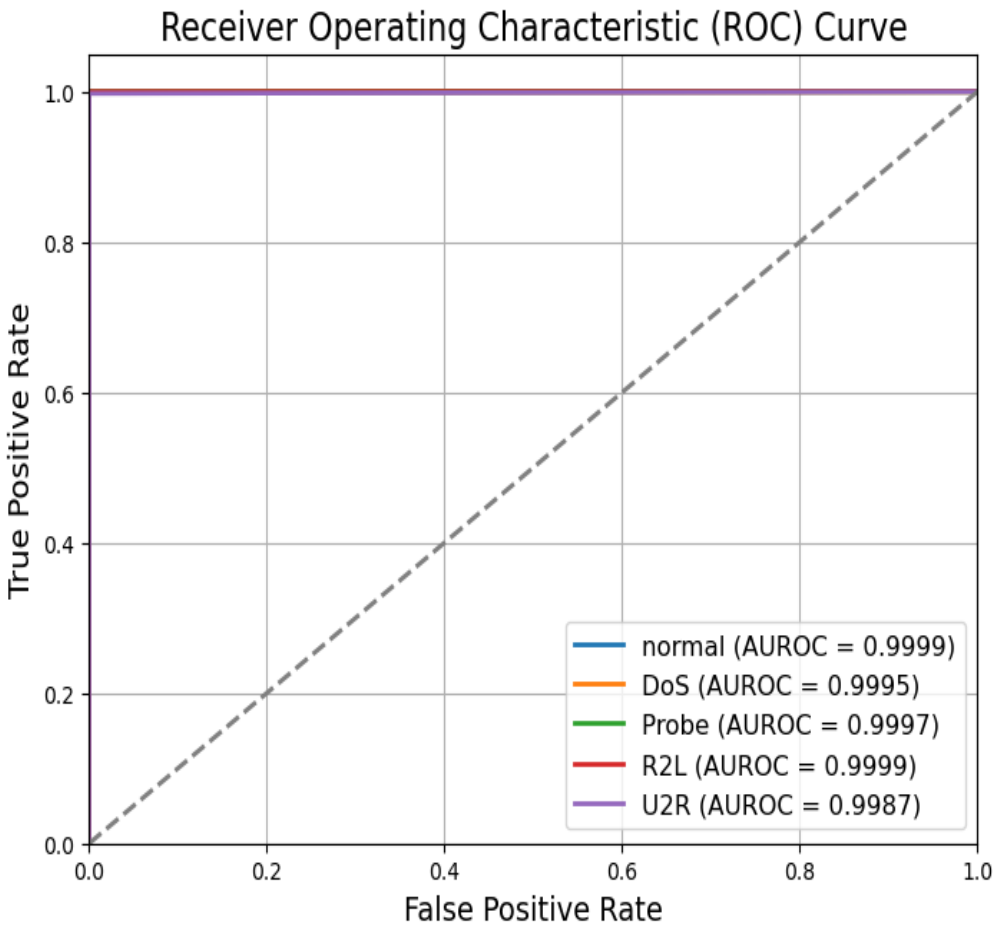


Figure 4.8: Area under the ROC Curve for Decision Tree
Source: Researcher (2024)

Logistic Regression (LR) Classifier

Training and evaluating the Logistic Regression model, the classifier was initialized with a maximum of 1000 iterations to ensure convergence. The model was trained on the training dataset, which comprised 80% of the total data, and predictions were made on the test dataset, which comprised the remaining 20%. Figure 4.9 depicts confusion matrix of LR

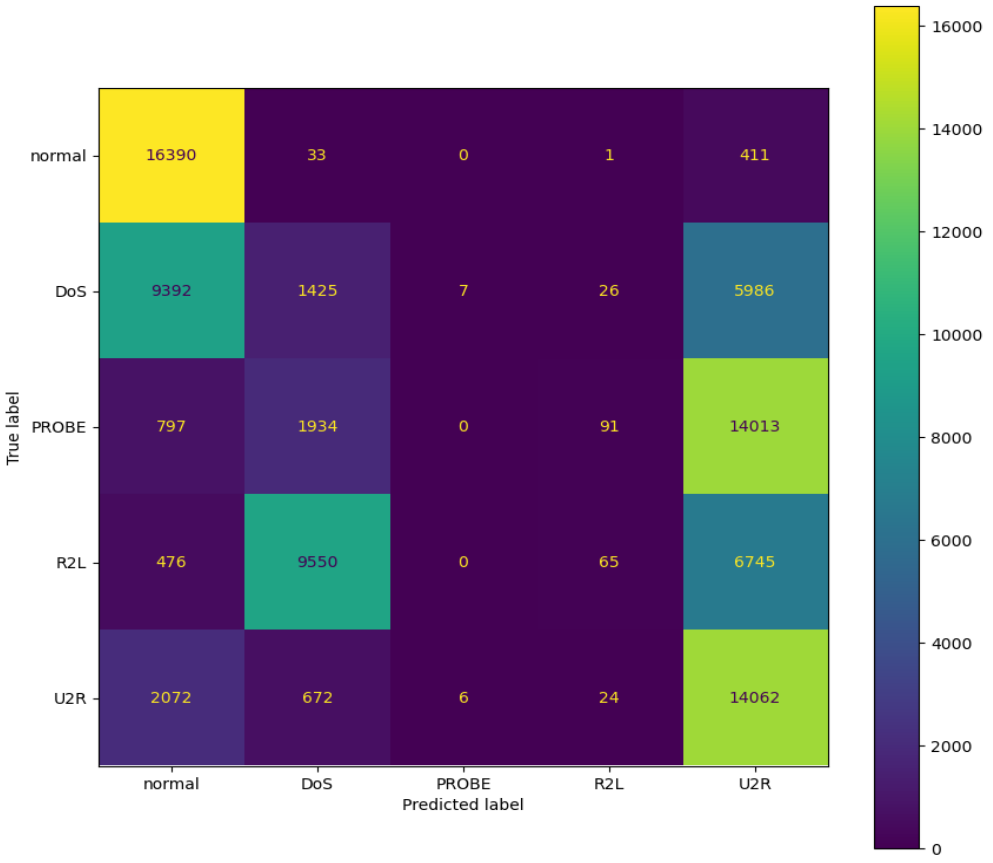


Figure 4.9: Confusion Matrix for Logistic Regression Classifier
Source: Researcher (2024)

From Figure 4.9, Logistic Regression correctly predicted 16390 TP records of Normal type, 1425 TP records of DOS, 6 5 TP records of R2L and 14062 TP records of U2R attack type. However, the model misclassified 9392 records of DOS as Normal, 797 records of PROBE as Normal, 476 records of R2L as Normal and 2072 records of U2R as Normal type. The model also misclassified 33 records of Normal as DOS, 1934 records of PROBE as DOS, 9550 records of R2L as DOS, and 672 records of U2R as DOS. Similarly, the model misclassified 7 records of DOS as PROBE attack type; and 1 record of Normal type as R2L, 26 records of DOS type as R2L, 91 records of PROBE type as R2L attack type, and 24 records of U2R attack types as R2L. Also, the model misclassified 411 records of Normal type as U2R attack type, 5986 records of DOS

attack type as U2R attack type, 1413 records of PROBE attack type as U2R and 6745 records of R2L attack type as U2R attack type.

To assess the model's performance, k-fold cross validation, with k=10 was used, paving the way for splitting the dataset into 10 different sets, with 90% used to train and the remaining 10% used to assess the model. The resulting performance parameters are recorded in Table 4.6 as accuracy, precision, recall, F1-score and AUC of ROC.

Table 4.6: Performance parameters for Logistic Regression

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC**
Logistic Regression	0.37945	0.26451	0.379460	0.259775	0.7421

In evaluating the Logistic Regression model using the NSL-KDD dataset, the performance metrics highlight the model's strengths and weaknesses. The model achieves a moderate overall accuracy of 37.945%, with low recall and precision resulting in an F1-Score of 25.9775% and a ROC of 0.7421.

The AUC-ROC scores for individual attack classes were: class 0 (Normal) at 0.9330, class 1 (DoS) at 0.2893, class 2 (PROBE) at 0.6684, class 3 (R2L) at 0.9354, and class 4 (U2R) at 0.8845 as depicted in Figure 4.10 Specifically, the AUC curve establishes the relationship between false negatives and false positives. While ROC curve is obtained by plotting the TPR against FPR (Santos *et al.,* 2013).

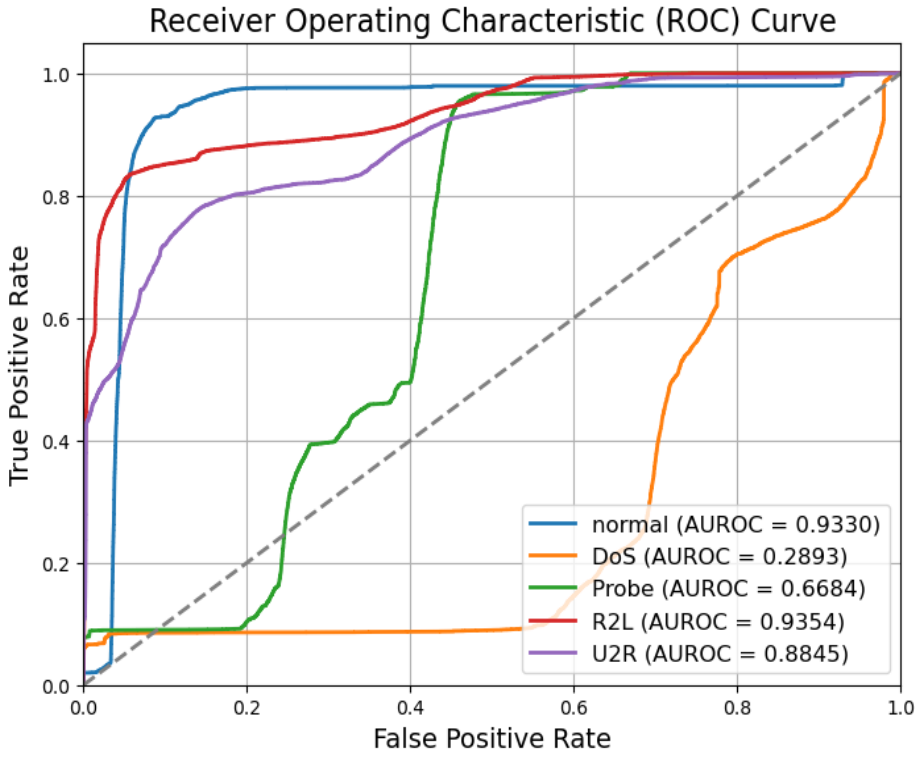


Figure 4.10: Area Under the ROC Curve of Logistic Regression
Source: Researcher (2024).

4.5 Ensemble Learning (Soft Vote) Classifier

The ensemble learning approach employed in this implementation leverages the strengths of multiple individual classifiers to enhance predictive performance. The voting classifier is the core of this strategy, combining five distinct models: K-Nearest Neighbors (KNN), Random Forest (RF), Naive Bayes (GNB), Decision Tree (DT) and Logistic Regression (LR). Soft voting is utilized, where the predicted class probabilities from each model are averaged, and the class with the highest average probability is selected. This ensemble method ensures a more balanced and robust prediction. The weights assigned to each model—slightly favoring KNN and Decision Tree—reflect their relative contributions to the ensemble, fine-tuning the overall performance. The process involves defining the ensemble model with these estimators, training it on the dataset, predicting outcomes using test dataset.

Figure 4.12 depicts the confusion matrix of the soft vote classifier, comparing the existing or expected labels with the predicted ones, and generating TP, FP, TN and FN values of the comparison.

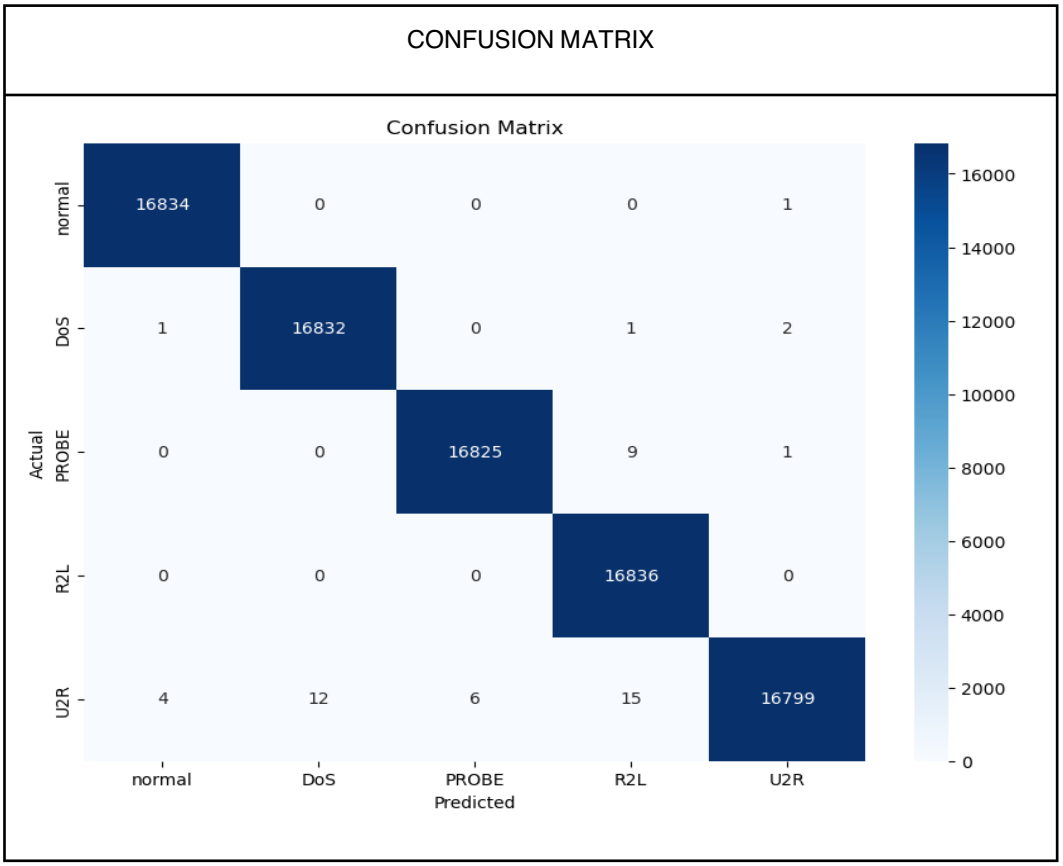


Figure 4.11: Confusion Matrix of Soft vote classifier
Source: Researcher (2024)

From Figure 4.12, soft vote classifier predicts 16834 TP records of Normal type, 16832 TP records of DOS, 16825 TP records of PROBE, 16836 TP records of R2L attack type, and 16799 TP records of U2R attack type. Although the misclassified records seem negligible, the following misclassifications were observed. 1 DOS record was misclassified as Normal and also 4 U2R records were misclassified as Normal. Similarly, 12 U2R records were misclassified as DOS attack type, 6 U2R records were also misclassified as PROBE attack type. Similarly, 1 DOS record was misclassified as R2L attack type, 9 PROBE records were misclassified as R2L attack type, and 15 U2R

records were misclassified as R2L attack type. The model also misclassified 1Normal record as U2R, 2 DOS records as U2R, and 1 PROBE record as U2R attack type.

Its performance was evaluated using k-fold cross validation with resulting metrics such as accuracy, precision, recall, F1-Score, and AUC-ROC as depicted in Table 4.7.

Table 4.7: Performance parameters of Soft Vote Classifier

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Ensemble Learning (Voting Classifier)	0.99788	0.99788	0.997889	0.99788	0.99996

Source: Researcher (2024)

The Voting Classifier demonstrates exceptional performance, with metrics indicating near-perfect predictive capabilities. The model achieves an accuracy of 0.99788, precision of 0.997879, recall of 0.997885, F1-Score of 0.997880, and a ROC-AUC score of 0.99996. The confusion matrix further supports these results, showing minimal misclassifications and indicating that the majority of instances are correctly classified. The AUC-ROC scores for each class—normal (1.00000), DOS (1.00000), PROBE (1.00000), R2L (1.00000), and U2R (0.99999)—highlight the model's excellent capability to distinguish between different attack types and normal traffic. The values are depicted in Figure 4.13

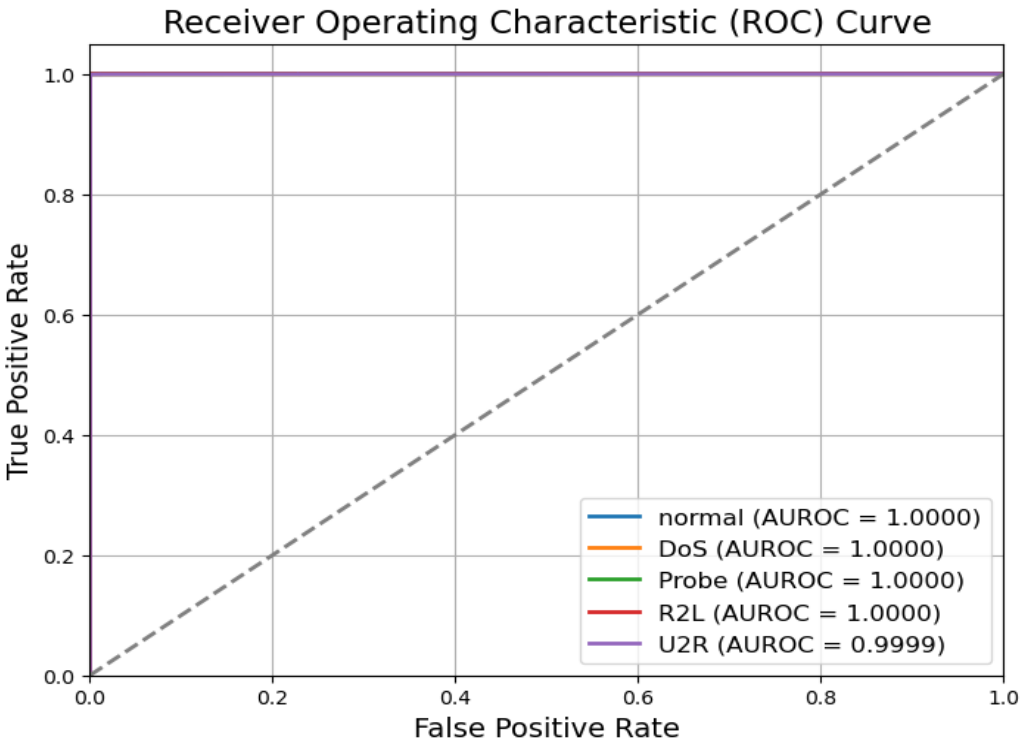


Figure 4.12: Area Under the ROC Curve of Soft Vote Classifier
Source: Researcher (2024)

This robustness in classification is critical for network intrusion detection, ensuring high precision and recall across all categories. The ensemble's ability to integrate the predictions from various models, balancing their strengths and weaknesses, leads to a superior overall performance, making it an ideal choice for this task. The strategic weighting of the models further enhances its effectiveness, providing a highly accurate and reliable solution for detecting network intrusions.

4.6 Discussion

In this section, the various models are analyzed to classify malware threats. Secondly, this research work is compared with two other research works in literature that used more than one dataset in training, analyzing and testing the models. These datasets are tagged as legacy dataset NSL-KDD dataset, UNSW-NB15 and ECML/PKDD2007 Dataset; while the modern datasets are CSE-CIC-IDS2018, CIC-IDS2017 and CSIC-

HTTP 2010. The models trained and analyzed include Random Forest (RF), Decision Tree (DT), K-Nearest Neighbor (KNN), Naïve Bayes (NB) and Logistic Regression (LR). The models were evaluated using the following cross-validation parameters Accuracy, Precision, Recall, F1-score, and Area under the ROC curve (AUC-ROC). Also used to plot the ROC curve are FPR and FNR values. Table 4.8 depicts the bench mark that contains analyzed values of the models trained and tested in this research work. The values contained in Table 4.9, Table 4.10, Table 4.11 and Table 4.12 are respectively compared and ranked with the bench mark values.

4.6.1 Comparative Analysis of the Models in this research work

The comparative analysis of different models reveals significant differences in their performance metrics as depicted in Table 4.8. The Random Forest Classifier outperforms other models with the highest accuracy of 0.99932 and an exceptional ROC-AUC score of 1.0, indicating its superior ability to distinguish between classes. The KNN model also shows strong performance, with an accuracy of 0.98816 and a ROC-AUC of 0.99740. In contrast, the Naive Bayes classifier demonstrates much lower performance across all metrics, with an accuracy of 0.39624 and a ROC-AUC of 0.7419, suggesting it struggles with the dataset's complexity. The Decision Tree Classifier performs nearly as well as the Random Forest with an accuracy of 0.99804 and a ROC-AUC of 0.9988, though its precision score appears to be incorrectly reported, matching that of the Naive Bayes Classifier. Overall, the Random Forest Classifier and Decision Tree Classifier exhibit the best performance, making them the preferred choices for this classification task.

Table 4.8: Comparative Analysis of the Models Used in this Research Work

Model	Accuracy	Precision	Recall	F1-Score	ROC
KNN Classifier	0.98816	0.988158	0.988170	0.988152	0.99740
Random Forest Classifier	0.99932	0.999322	0.988170	0.999319	1.0
Naive Bayes Classifier	0.39624	0.410736	0.396438	0.268998	0.7419
Decision Tree Classifier	0.99804	0.410736	0.998042	0.998041	0.9988
Logistic Regression	0.37945	0.26451	0.379460	0.259775	0.7421

Source: Researcher (2024)

4.6.2 Basis for comparing this research work with other works in literature

The works in literature compared with this research work operate under the same Android operating system (and can also use Windows operating system), used either ensemble learning approach or hybridization technique or both. The dataset used to train and analyze the models is NSL-KDD dataset. It is the desire of this researcher to compare, not only the performances of the trained models, but the contribution of the dataset used in the analysis to the performance of the models. Therefore, five other datasets are used in the comparison, to determine their importance or otherwise relative to NSL-KDD dataset. They include ECML/PKDD 2007 dataset, CSIC-HTTP 2010 dataset, CSE-CIC-IDS 2018 dataset, CIC-IDS 2017 dataset and UNSW-NB 15 dataset. These datasets are legacy datasets and more modern datasets, and they are applied by Chakir *et al.* (2023) and Saini *et al.* (2023) respectively. The base models trained include KNN, DT, NB and LR (either as single classifiers or in ensemble learning mode). They have a common intent of detecting malware or web-based attacks on host

systems and the cloud using permissions, system call logs, and API call logs. The analysis was done using cloud computing technology, ensemble learning and hybridization of ensemble learning techniques. However, while this work uses Random Forest (an ensemble bagging technique) to train the base models (DT, NB and LR), the other works hybridized RF and XGBoost predictions in addition to determine the strength of the resulting common classifier. KNN dataset does not train its models per se, rather it uses Euclidean distance measure to determine the nearest neighbor of the k set and assigns the object to the majority class in the k set of nearest neighbors. A brief introduction of the datasets follows:

CSE-CIC-IDS 2018 Dataset

This dataset is a combination of the communications security establishment (CSE) and the Canadian Institute of Cyber-security (CIC). The two bodies jointly produced the dataset. It has 16.23 million records. However, it is an unbalanced dataset with 17% classes belonging to the abnormal attack types. It also has the potential of identifying new attack types, it is modern, real-world and practical in modern day use.

CIC-IDS2017 Dataset

This dataset comprises of benign and recent attack types, and closely reflect real-world data such as PCAP files. It uses CSV files among others. Attack types in the dataset include DDOS, Botnets, Infiltration, web attacks, Heart bleed, brute force, DOS, Brute force SSH and FTP, and more.

UNSW-NB15

This dataset was created by the University of New South Wales (UNSW). Its raw network packets were generated using IXIA (a modern network testing program)

perfect storm program in UNSW. It has up to nine attack types with 49 features and a class label.

NSL-KDD Dataset

This dataset is an improvement of KDDCup'99 dataset, however, it may not be a perfect real-world representation, but an adequate benchmark dataset to help researchers compare different intrusion detection methods. It includes different attack types, which are grouped into DOS, PROBE, R2L and U2R attack types and the Normal applications. It has 42 features, including the class label.

ECML/PKDD 2007 Dataset

This dataset was generated for ECML/PKDD (European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases) discovery challenge. It contains 24504 and 10,502 valid requests for training and testing respectively, and 15110 malicious requests for testing. It includes web-based attacks such as XSS (Cross-Site Scripting), SQLI (Structured Query Language Injection), LDAP (Lightweight Directory Access Protocol) injection.

CSIC HTTP 2010 Dataset

This dataset contains requests generated to an e-commerce application. It contains two types of requests: 36,000 normal requests for training, and 36,000 normal requests for testing, and 25,000 requests classified as attacks for testing. It contains many types of web attacks like SQLI, XSS, CRLF (Carriage return, ASCII 13, \r) Line feed, ASCII 10, \n) and buffer-overflow.

The problem with the two types of datasets is that they contain only valid training requests. Therefore, they do not provide a good model that trains only with normal data

samples. As such, they will not be able to detect attacks that mimic normal activity. Hence, the authors extracted samples from attack dataset to mix with the normal dataset to use for intrusion detection.

4.6.3 Comparison of Results in this research work with That of Other Works in Literature

The comparison is done according to the datasets, each of these datasets is used to train and analyze the models.

NSL-KDD Dataset

This dataset was used to train and analyze models in Saini *et al.* (2023) and the results are depicted in Table 4.9.

Table 4.9: Performance Metrics for NSL-KDD DATASET

	Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	TNR (%)	FPR (%)	FNR (%)
1	KNN	95.03	94.14	95.19	94.67	94.89	5.11	4.81
2	DT	98.85	98.67	98.85	98.76	98.85	1.15	1.15
3	RF	99.15	99.14	99.03	99.08	99.26	0.74	0.97
4	NB	85.66	90.52	77.13	83.29	93.02	6.98	22.87
5	LR	87.39	88.92	83.14	85.93	96.06	0.94	16.86
6	HV	99.24	99.28	99.09	99.18	99.38	0.62	0.91

Source: Saini *et al.*, (2023)

Where:

KNN – K-Nearest Neighbor Classifier.
DT – Decision Tree Classifier
RF – Random Forest Classifier
NB - Naïve Bayes Classifier
LR - Logistic Regression Classifier
HV - Hard Vote Classifier

From Table 4.9, the accuracy of this research work, Table 4.8, outperformed that of Table 4.9 with RF (99.93%) as against HV (99.24%). Similarly, the ensemble classifier value of this research work, Table 4.8, SV (99.79%) outperformed the hard vote value of Table 4.9 HV (99.24%). However, the least value of Table 4.9 NB (85.66%) outperformed that of this research work, which is LR (37.94%) in Table 4.8.

Similarly, the highest F1-Score RF (99.93%) of Table 4.8 outperformed that of Table 4.9 HV (99.18%), but the lowest value of F1-Score in Table 4.9, NB (83.29%) outperformed that of Table 4.8 NB (26.89%). Again, the precision value of Table 4.8 in this research work, RF (99.93%) outperformed that is Table 4.9 HV (99.23%). However, the lowest precision value of Table 4.9 LR (88.92%) outperformed that of Table 4.8, which are two with the same value NB (41.07) and DT (41.07%).

The AUC-BOC of this research work's dataset RF (1.00) outperformed that of Table 4.9, which is HV (0.9923). This value indicates that RF is the best model in this research work and more preferable to hard vote of NSL-KDD dataset in Saini *et al.* (2023).

CSE-CIC-IDS2018 Dataset

It is pertinent to observe that this dataset is one of the modern datasets, and it is used to compare with NSL-KDD dataset used in this research work, considered a legacy dataset. Table 4.10 depicts the results of this dataset, which will be compared with the bench mark values in Table 4.8. The highest accuracy value of this research work in Table 4.8 is RF (99.93%). It outperformed that of Table 4.10, which is HV (98.92%). Similarly, the soft vote ensemble value of the research work Table 4.8 is SV (99.78%) and outperforms the ensemble value of Table 4.10, which is HV (98.92%).

	Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	TNR (%)	FPR (%)	FNR (%)
1	KNN	98.40	98.56	98.22	98.39	98.58	1.42	1.78
2	DT	98.31	98.56	98.04	98.30	98.58	1.42	1.96
3	RF	98.66	99.58	97.72	98.64	99.60	0.40	2.28
4	NB	86.77	81.45	95.08	87.74	78.53	21.47	4.92
5	LR	85.04	82.48	88.83	85.54	81.29	18.71	11.17
6	HV	98.92	99.47	98.35	98.90	99.48	0.52	1.65

Table 4.10: Performance Metrics for CSE-CIC-IDS2018 Dataset

Source: Saini *et al.* (2023)

The F1-Score value of the bench mark table (Table 4.8) is RF (99.93%), it outperformed the highest value of F1-Score in Table 4.10, which is HV (98.90%). However, the lowest F1-Score value of Table 4.10 LR (85.54%) outperformed that of Table 4.8, which NB (26.89). The precision value of this research work RF (99.93%) outperformed that of the Table 4.10, which is RF (99.54%). Indeed, the least precision value of Table 4.10 is NB (81.45%) and outperformed that of this research work, which is NB (41.07%) and DT (41.07%).

The highest value of AUC-ROC of the bench mark dataset NSL-KDD dataset is RF (1.00), and that of CSE-CIC-IDS2018 ensemble classifier is HV (98.913). Again, the research work dataset outperformed the modern dataset. It is pertinent to point out here that although NSL-KDD dataset is legacy dataset, it is a well composed dataset with normal and attack records, whereas the modern dataset is grossly imbalanced with only normal dataset. And attack features were only extracted from another evil dataset and fused into the dataset tor it to be used for this detection exercise (Saini *et al.* 2023).

CIS-IDS2017 Dataset

This modern dataset also primarily contains normal training and training datasets, but the researchers of the paper had to extract and infuse attack APT (Advanced Persistent Threats) into the dataset to detect abnormal threats in the dataset (Saini *et al.* 2023). Comparing the accuracy in Table 4.11, it is observed that the value of accuracy in Table 4.8, the bench mark values, outperformed that of Table 4.11. The obtained values are RF (99.93%) and HV (99.91%) respectively.

Table 4.11: Performance Metrics for CIC-DIS 2017 Dataset

	Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	TNR (%)	FPR (%)	FNR (%)
1	KNN	99.54	99.27	99.82	99.54	99.26	0.74	0.19
2	DT	99.79	99.77	99.81	99.79	99.77	0.23	0.19
3	RF	99.88	99.92	99.84	99.88	99.92	0.08	0.16
4	NB	85.06	79.07	95.54	86.53	74.49	25.51	4.46
5	LR	93.04	91.29	95.23	93.22	90.83	9.17	4.77
6	HV	99.91	99.88	99.95	99.91	99.88	0.12	0.05

Source: Saini *et al.* (2023)

However, the least model performance in the said dataset NB (85.06%) outperformed that of the benchmark dataset, which is LR (37.94%). Similarly, F1-score highest value of RF (99.93%) in Table 4.8 outperformed that of Table 4.11, which is HV (99.91%). However, the least performed model in terms of F1-score parameter in Table 4.11 with NB (86.53%) outperformed that of Table 4.8, which is LR (26.89%).

The precision values are very keenly contested, as that of Table 4.8 outperformed that of Table 4.11 with RF (99.93%) as against HV (99.92%), but the lowest model performance of Table 4.11, which is NB (79.07%) outperformed that if Table 4.8, which is NB (41.07%) and DT (41.07%) respectively.

UNSW-NB15 Dataset

This dataset is also regarded as a legacy dataset. The essence of comparison is because it used the same training and testing features as NSL-KDD dataset of this research work. Referring to Table 4.12, the accuracy highest value of RF (99.93%) of Table 4.8 outperforms that of the Table 4.12, which is HV (97.11%). And the lowest model performance of Table 4.12, which is NB (95.04%) indeed outperformed the lowest model value of Table 4.8 (the bench mark), which is LR (37.945%). The F1-Score value of this work, RF (99.93%) also outperformed that of the competing dataset, Table 4.12, which is LR (99.66%). However, the lowest performing model of F1-Score value in Table 4.12 is KNN (95.53%) and it outperformed that of the benchmark table (Table 4.8), which is NB (26.89%).

	Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	TNR (%)	FPR (%)	FNR (%)
1	KNN	95.50	96.30	95.56	95.53	95.43	4.57	4.44

2	DT	96.24	96.72	96.49	96.60	95.92	4.04	3.51
3	RF	96.98	96.32	98.31	97.31	95.33	4.67	1.69
4	NB	95.04	93.20	98.22	95.65	91.07	8.93	1.78
5	LR	96.18	93.84	99.66	99.66	91.85	8.15	0.34
6	HV	97.11	95.89	99.04	97.44	94.71	5.29	0.96

Table 4.12: Performance Metrics for UNSW-NB15 Dataset

Source: Saini *et al.*, (2023)

Similarly, the highest precision score of Table 4.8, which is RF (99.93%) outperformed that of Table 4.12, which is DT (96.72%); and the lowest performing model in Table 4.12, which is NB (93.20%) outperformed that of Table 4.8, which is NB (41.07%) and DT (41.07%) respectively.

ECML/PKDD2007 Dataset

This dataset was used by Chakir *et al.* (2023) with the same training algorithm and base models as the dataset used by this research work, NSL-KDD dataset. The comparison is based on the information extracted and depicted in Table 4.13. The highest accuracy value of this research work, which is depicted in Table 4.8 with RF (99.93%) outperformed the highest accuracy value in Table 4.13, which is RF (99.597%). However, the least accuracy value in this dataset (ECML/PKDD2007 Dataset), which is KNN (95.11%) outperforms that of this research work LR (37.945%). Similarly, the highest F1-Score value of this research work, which is RF (99.93%) outperformed that of the contending paper in Table 4.13, which is RF (99.129%), and the lowest model performance of Table 4.13 is KNN (88.112%) outperforms the least value in this research work, which NB (26.89%), Table 4.8 refers. Again, the highest precision value of Table 4.13, which is NB (99.952%) outperformed that of this research work, which is RF (99.93%), even its lowest model performance value that is DT (96.383%) still

outperforms that of this research work, which is NB (41.07%) and DT (41.07%) respectively. The highest value of AUC-ROC of this paper is equal to 1.0 from the HV and RF respectively. Although this research work also recorded RF (1.00) AUC, the soft vote value is indeed lower than that of Table4.13 with a value of SV (0.99960). With the high values of AUC recorded in this dataset by two models, HV and RF, which gives it an edge over this research work, the high accuracy value notwithstanding, it is the opinion of this researcher that this dataset performed creditably well with a better overall performance as against this research work. From literature, the higher the AUC value, the better the model performance. More so, ECML/PKDD2007 Dataset has a higher precision value compared to that of this research work, and, the higher the precision value, the better the model performance. These values led the researcher to conclude that this dataset has an edge over NSL-KDD dataset.

Table 4.13; Performance Evaluation for Each Classifier Based on the Dataset.

ECML/PKDD 2007 Dataset						
Classifier	A (%)	R (%)	P (%)	F1 (%)	FPR (%)	FNR (%)
NB	96.07	82.914	99.952	90.639	0.912	17.086
KNN	95.112	79.002	99.597	88.112	0.095	20.998
DT	99.140	100	96.583	98.158	1.116	0
LR	97.776	92.335	97.843	95.009	0.606	7.665
RF	99.597	100	98.274	99.129	0.523	0
HV	99.451	99.880	97.773	98.815	0.677	0.120

Source: Chakir *et al.* (2023)

Where

NB: Naïve Bayes Classifier
KNN: K-Nearest Neighbor Classifier
DT: Decision Tree Classifier
LR: Logistic Regression Classifier
RF: Random Forest Algorithm
HV: Hard Vote Classifier
A: Accuracy

R: Recall
P: Precision
F1: F1-Score
FPR: False Positive Rate
FNR: False Negative Rate

CIC- HTTP 2010 Dataset

This dataset is considered a relatively modern dataset with more of normal training and testing dataset. The dataset has imbalanced features that needed the use of SMOTE, Pearson Correlation and Information gain (IG) to balance the data labels before preprocessing. The dataset is then split to train and test the model in a ratio of 80:20, and after training, prediction of the test dataset is done and the models' performances are evaluated on the dataset. The performance parameters are contained in Table 4.14.

Table 4.14; Performance Evaluation for Each Classifier Based on Dataset

CSIC HTTP 2010 Dataset						
Classifier	A (%)	R (%)	P (%)	F1 (%)	FPR (%)	FNR (%)
NB	93.100	93.100	93.181	93.097	4.733	9.067
KNN	88.533	88.633	89.257	88.588	5.067	17.667
DT	99.140	100	96.383	98.158	1.116	0
LR	97.776	92.335	97.843	95.009	0.606	7.665
RF	99.867	99.867	99.867	99.867	0.267	0
HV	98.367	98.367	98.378	98.367	2.400	0.867

Source: Chakir *et al.* (2023)

From Table 4.14, it is observed that the highest accuracy value is RF (99.867%), it is outperformed by the highest accuracy of this research work RF (99.93%), however, the lowest accuracy value of the competing dataset, which is KNN (88.633%), outperformed that of this research work, which is LR (37.94%). Again, the F1-Score

value of this research work, RF (99.93%), outperformed that of Table 4.14, which is RF (99.867%), but its lowest model performance, which is KNN (88.588%) also outperformed that of this research work, in Table 4.8, which is NB (26.89%).

Similarly, the precision value of this research work, RF (99.93%) outperformed that of the Table 4.14, which is RF (99.867%), but its lowest model performance in terms of precision, KNN (89.257%) out smarted that of this research work, which is NB (41.07%) and DT (41.07%) respectively. It is also observed that AUC-ROC of this research work, RF (1.00) outperformed that of the work in literature, as depicted in Table 4.14, which is HV (0.96875).

Since high AUC implies that the model is better, then RF is a better model as against the HV model of the work's ensemble learning problem.

4.6.4 Lessons drawn from comparing this work with works in literature

i. The principle of producing better detection results by combining (hybridization or ensemble learning) classifiers against single classifiers, may not always be true. Depending on the problem at hand, single classifiers may do better than ensemble hard or soft vote classifiers, as evident in these works, and vice versa (Saini *et al.* 2023).

ii. The tradition of comparing the efficacy of trained models using a single dataset, may not always produce better results as against the use of multiple datasets on a single set of training models (Domingos, 2012; Saini *et al.* 2023; Chakir *et al.* 2023).

iii. The most known and widely used datasets were generated for network-based detection and may not be suitable for web-based attack detection solutions because they do not contain real-world attack samples. This has made

researchers to seek new and modern datasets for use in research works. However, this has not proved very effective as can be seen in the results of this comparison between modern and legacy attack sets, giving legacy attack sets leverage over the modern ones that mostly contain normal data features (Saini *et al.* 2023; Chakir *et al.* 2023).

iv. Traditionally, datasets are mostly imbalanced, with normal and DOS attack types having more records than some other attack types, which usually make accuracy of the model performance not always reliable, emphasis is rather shifted to precision and F1-score. Even the AUC results are best when the dataset is imbalanced (Saini *et al.* 2023). Evidence has shown that with the use of SMOTE, Pearson correlation, and information gain (IG) to balance and preprocess the dataset before normalization and reduction of features using PCA has yielded very good results, irrespective of the age of the dataset.

4.7. Publications

4.8 Ethical issues

4.8.1 Conflicts of Interest

This research work does not have any conflict of interest or personal relationship with a third party whose interest can be positively or negatively influenced by the content of this research work.

4.8.2 Citation

NSL-KDD data set, obtained from Kaggle repository, is open source, and it is used purely for research purpose and the source duly acknowledged. No alteration to the content of the dataset was made. Sources used in the body of this work are appropriately acknowledged and cited.

CHAPTER FIVE
SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Preamble

This chapter gives an overview of this research work. It summarizes the importance of electronic healthcare system (EHS) in nation building, for health is wealth; its security implications, especially to the patient, with particular emphasis on ethics in handling patients' records. EHS system is constantly targeted by malware and cyber attackers seeking to steal and extract sensitive information, and is also subjected to tactics aimed at intimidating, harassing, and disrupting the system as well as the patients it serves. The stolen information is sold for money. Efforts are made to detect the intrusions by combining algorithms to produce a more formidable model that will produce better detection through the use of voting classification. The dataset used to train the models is NSL-KDD dataset. The trained models predict the labels, which are used to compare with the expected labels using confusion matrix; and their performances are measured using cross validation parameters such as accuracy, precision, recall, F1-score and AUC. Also, contribution to knowledge and benefits of this research work to the hospital, patients and the society at large are pointed out.

5.2 Summary

Electronic healthcare system (EHS) has seriously outperformed traditional paper-based healthcare system, which unfortunately is still being used in Nigerian hospitals. With EHS, hospital personnel can assess patient's records with ease, share information between departments, units and even hospitals for inter-operability.

These benefits endeared hospitals, patients and even the staff, as they reduce the risk of patients' privacy abuse by internal and external intruders. Some malware developers'

intent is to make money from the innocuous user by distorting communication, stealing important information and even blackmail through espionage and ransomware.

This research work attempts to salvage the security implications of using mobile devices, laptops and PCs and other network-based equipment and the cloud to foster healthcare in hospitals. It combines machine learning (ML) algorithms, through ensemble learning method (bagging), using Random Forest algorithm to train base models (DT, NB and LR); and KNN. The predictions of these classifiers were aggregated to a formidable mega model (soft vote classifier) that effectively detects malware threats. The proposed system monitors applications requesting for permissions to either install, update applications, or even mischievously subvert operations, from the user, who could be the doctor, nurse, lab technician, pharmacist or even the patient. Such dangerous requests are detected and reported by intrusion detection system (IDS) to the system Administrator, the user or intrusion response system (IRS) for necessary actions.

The proposed system is developed using python programming language, in conjunction with its external libraries such as scikit-learn, pandas and NumPy, to mention but a few. It is evaluated for performance using confusion matrix standard metrics such as accuracy, precision, recall, and f-measure. This research work attained 99.93% (RF) accuracy, precision, and F1-score respectively; and an overall AUC score of 1.00 (RF). Indeed, the RF value outperformed the ensemble soft vote value of 99.79% (SV), and has been confirmed by the AUC-ROC value to be the best choice classifier of malware threats. Individual models scored and accuracy of 98.82% (KNN), 39.62% (NB), 99.80% (DT) and 37.94% (LR). Similarly, the models had F1-Score of 98.81% (KNN), 26.89% (NB), 99.80% (DT), and 25.97% (LR). The results evidenced that LR struggled with the complexity of the dataset in analyzing and detecting malware threats.

The attack labels that resulted from these analyses include Normal, DOS, PROBE, R2L and U2R. The predictions of this research work were compared with that of two other works in literature piloted by Saini *et al.* (2023) and Chakir *et al.* (2023). These papers introduced the use of multiple datasets in training and testing a specific set of models. Although the results from these works came from both legacy and modern datasets, they could not perform as good as NSL-KDD dataset used by this research work, which was properly balanced using SMOTE, Pearson correlation and information gain (IG) data extraction, with normalization and feature selection using PCA. Feature imbalance in ML datasets has been a problem in carrying out effective malware detection by trained models without bias and over fitting.

5.3 Conclusion

The results of this study indicate that ensemble learning methods do not always outperform single classifiers in network-based attack detection, as it is commonly presented in literature (Chakir *et al., 2023*). Both have their strengths and weaknesses. Ensemble learning approaches exhibit higher accuracy, precision and low false positive rate compared to single classifiers. However, in terms of recall, FNR, training and prediction time, single classifiers perform better.

The results presented in this work have important implications in efforts made by research community to combat the menace of malware threats, by emphasizing the importance of electronic healthcare system in preference to the traditional paper-based healthcare system. Though EHS is not without challenges, it is the most preferred hospital management system globally. This research work has successfully combined machine learning tools (Random Forest algorithm, NB, KNN, DT and LR models) to

improve malware threat detection and reporting using cloud computing technology (CCT) in healthcare systems, and especially hospitals in Nigeria.

Secondly, using more classifiers (ensemble learning) in an effort to curb the menace of malware has paid-off, compared to the use of single classifiers.

Thirdly, EHS has enabled the sharing of patients' records between and within healthcare professionals, for effective healthcare delivery, within ethical principles. The safety and respect of patients' privacy has been enshrined in this research work, by ensuring that only authorized persons and applications have access to patients' records.

One of the common problems in machine learning processes is the database, especially public databases that are modern, recent and contain practical normal and malicious applications for use by researchers to conduct malware detection and control. The current databases commonly used are the KDD family and UNSW-NB15, which are no longer suitable particularly in web-based attack detection. However, the most recent databases include ECS-CIC-IDS2018 and CIC-IDS2017, which have their shortcomings. For instance, they have only 2180 and 928 web based attack instances. In particular, they contain only normal applications, and not fit for training and testing malware presence in the cloud. The challenge then is to create a novel all rounded database that will contain both normal and malware applications with modern and recent real-world problems.

The dataset used in this research work is deemed legacy. But the use of SMOTE, Pearson correlation and information gain to balance the data distribution and extraction, placed it over and above the modern ones. Proper feature selection and extraction is critical in model training for better performance.

5.4 Scientific Implications of Findings

i. Data generated these days by hospital management are huge and unstructured. They are generated in the cloud, stored in the cloud (use of servers with huge disk spaces), processed in the cloud, and results transmitted to the device through its IP address as client. Thus, reducing the risk of pests, occupation of huge storage spaces, difficulty of searching for patient's information and the time, and usurping the overhead of physical security. The secured internet, policed by this product, will enable researches with good results, as malware may not even find space to avert controlled environment (sand boxes).

ii. The application of this research work will assist patients in assessing their health records, be it in the hospital or at home. Thus, making healthcare delivery patient- centered.

iii. The introduction of intelligent techniques to detect malware threats in electronic healthcare system is timely, especially these days that COVID'19 is ravaging the world with many variants, and now subvariants of omicron are circulating. The use of this product will enable the sharing of medical records of people traveling into the country, and vice versa. Thus, reducing the embarrassing situation of quarantining every traveler that enters the country from other countries, and reduce cost.

5.5 Contribution to Knowledge

i. Pham *et al.,* (2018) trained tree classifiers such as decision tree (DT) and random forest (RF) using Adaboost algorithm and NSL-KDD dataset, the result obtained was 80.59% (DT) and 80.07% (RF). This research work improved on their work by training more classifiers such as Random Forest

(RF), K-nearest neighbor (KNN), decision tree (DT), logistic regression (LR) and Naïve Bayes (NB) using NSL-KDD dataset and got the following accuracies: 99.93% (RF), 99.78% (SV), 99.80% (DT), 98.82% (KNN) and 37.94% (LR) and 39.62% (NB).

ii. Pattern matching of target object with signatures of malware families in databases had been the old process, however, this research work has trained classifiers to learn the behavior and characteristics of 67,343 normal applications and use that knowledge to detect malware, devoid of signatures of malware families, as malware developers have outsmarted the process using obfuscation techniques.

iii. Saini *et al.* (2023) introduced the use of multiple datasets in the search for effective method to detect malware threats. Their results were outperformed by this research work with just a single dataset, which has balanced features, using SMOTE.

iv. To the best of our knowledge, the traditional believe that combining ML classifiers or models in detecting malware threats would perform better than single classifiers, has been proved not to be true in all cases. Depending on the type of problem being solved, single classifiers have performed better than ensemble learning consensus classifier in some cases. For instance, RF outperformed soft vote classifier with 99.93% accuracy as against 99.78% respectively.

5.6 Suggestions for Future works

i. **Implementation** - To implement these developed and tested techniques, it is recommended that they be integrated to Application Programming Interface (API), and uploaded to the cloud. Then the front end (user interface

- UI) should be developed using JavaScript, HTML and software code control (SCC) system. The UI will interact with the model at the back end via the API when an application is input through it for analysis.

ii. **Class imbalance** – Malware datasets are often skewed, which can lead to bias and affect the performance of trained models. There is, therefore, the need to look into the imbalanced issue to ensure accurate detection across all malware categories. For instance, most malware are greatly outnumbered by normal type and DOS attack type as against PROBE, R2L and U2R attack types.

iii. **Single execution Path** – Dynamic analysis method analyses and detects malware threats from a single execution path. Multiple execution paths should be explored to enable the system to be able to identify different behaviors displayed by the suspicious executable files.

iv. Resource sharing - There is still difficulty associated with sharing resources in the cloud, especially patients' records and other security issues. This should be investigated.

REFERENCES

Abraham, S. (2017). *List of Types of Malware*, MalwareFox. https://www.malwarefox.com/malware-types/ (Retrieved on 12th June, 2021)

Abushark, Y. B., Khan, A. I., Alsohami, F., Almalawi, A., Alam, M.M., Agrawal, A., Kumar, R. and Khan, R. A. (2022). Cyber Security Analysis and Evaluation for Intrusion Detection Systems. *Computers, Materials and Continua.* DOI: 10.32604/cmc.2022.025604

Aijaz, M., Nazir, M. and Mohammad, M.N. (2023). Thread Modeling and Assessment Methods in the Health Care IT System; A Critical Review and Systematic Evaluation. *Springer Nature Computer science,* 4:714. https://doi.org/10.1007/s42979-023-02221-1

Akram, F., Liu, D., Zhao, P., Kryvinska, N., Abbas, S., and Rizwan, M. (2021).. Trusworthy Intrusion Detection in E-Healthcare Systems. *Frontiers in Public Health, 9:788347.* DOI: 10.3389/fpubh.2021.788349

Al-Ajlan, A. (2015). The Comparison Between Forward and Backward Chaining. *Internal Journal of Machine Learning and Computing*, vol. 5, No. 2. DOI: 10.7763/IJMLC.2015.v5.492.

Alder, S. (2024). Security Breaches in Health Care in 2023. *The HIPAA Journal*

Aljawarneh, S., Aldwairi, M., and Yassrin, M.B. (2017). Anomaly-Based Intrusion Detection System Through Feature Selection Analysis and Building Hybrid Efficient Model. ELSEVIER. *Journal of Computer Science*. https://dx.doi.org/10.1016/j.jocs.2017.03.006.

Amazon Web Services (AWS) (2021). *Amazon Machine Learning: Developer Guide*. Amazon Web Services Inc. http://aws.amazon.com.

Anthony, R. A. (2014). A Study on Data Mining Based Intrusion Detection System. *International Journal of Innovative Research in Advanced Engineering* (IJIRAE),3 (1),.

Anton, H. and Rorres, C. (2014). *Elementary Linear Algebra*, Applications Version, 11-th Edition. ISBN 978-43441-3. Wileyplus.com.

Arshad, S; Khan, A; Shah, M.A., and Ahmed, M. (2016). Android Malware Detection and Protection: A Survey. *International Journal of Advanced Computer Science and Applications*. (IJACSA), 7(2).

Atkinson, M. (2015). *An Analysis of Android Application Permissions*. Pew Research Center, Internet and Technology.

Attah, A. O. (2017).Implementing the Electronic Health Record in Nigeria: Prospects and Challenges. A Master's Thesis in Telemedicine and E-Health (TLM-3902).The Arctic University of Norway.

Azad, C. and Jha, U. K. (2014). Data Mining Based Hybrid Intrusion Detection Systems. *International Journal of Science and Technology*, vol 7(6): 781 - 789,

Babita, E. and Kaur, G. (2017). A Review: Network Security Based on Cryptography and Steganography Techniques. *International Journal of Advanced Research in Computer Science.* Vol. 8, No.4. E-ISSN: 0976-5697

Baby, J.J. and Jeba, J. R. (2017). Survey Paper on Various Hybrid and Anomaly Based Network Intrusion Detection Systems. *Research Journal of Applied Sciences.* 2(3-4): 304 – 310.

Berrar, D. (2018). Cross Validation. *Encyclopadia of Bioinformatics and Computational Biology,* vol. 1, ELSEVIER, pp. 542 – 545. https://dio.org/10.1016/B978-0-12-809633-820349-x

Bhandari, A. (2020). Every Thing you Should Know About Confusion Matrix for Machine Learning. *Analytics, Vadhya, 17.*

Bhuyan, M. H., Bhattacharyya, D. K., and Kalita, J. K. (2014). *Network Anomaly Detection: Methods, Systems and Tools*. IEEE Communications Surveys and Tutorials, 16(1).

Breiman, L. (2001) Random forest. *Machine learning,* 45 (1): 5-32

Brownlee, J. (2021). How to Combine Predictions for Ensemble Learning. *Machine Learning Mastery* (E-BOOK).

Bui, L.T., Vu, V.T., and Dinhm T.T. H. (2017). A Novel Evolutionary Multi-Objective Ensemble Learning Approach for Forecasting Currency Exchange Rates. *Data knowledge Engineering*. https://dx.doi.org/10.1016/j.datak.2017.07.001

Cai, L., Li, Y., Xiong, Z. (2020). JOWMDroid: Android Malware Detection Based on Feature Weighting with Joint Optimization of Weight-Mapping and classifier Parameters. *Computer Society, 102086. https://doi.org/10.1016/j.cose.2020.102086*

Chakir, O., Rehaimi, A., Sadqi, Y., Alaoui, E. A. A., Cridun, M., Gaba, G. S., and Gurtov, A. (2023). An Empirical Assessment of Ensemble Methods and Traditional Machine Learning Techniques for Web-based Attack Detection in Industry 5.0. *Journal of King Saud University - Computer and Information sciences,* 35(2023):103 – 119. https://dio.org/10.1018/j.jksuci.2023.02.009

Cheng, D., Zhang, S., Deng, Z., Zhu, Y and Zong, M. (2014). KNN Algorithm With Data-Driven K Value. *Springer International Publishing,* Switzerland. Pp. 499 – 512.

Cheng, L., Liu, F., Yao, D. (2017). *Enterprise Data Breach: Causes, Challenges, Prevention, and Future Directions.* WI8RES Data Mining Knowledge Discovery. DOI: 10.1002/widm.1211.

Chumachenko, K. (2017). *Machine Learning Methods for Malware Detection and Classification.* Bachelor's Thesis, Information Technology.

Comsast Business (2021). DDOS Threat Report: DDOS Becomes a Bigger Priority as Multi-Vector Attacks are on the Rise. (Retrieved May 20, 2024). https://corporate.comsast.com/press/releases.

Dewa, Z. and Maglaras, L. A. (2016). Data Mining and Intrusion Detection Systems. *International Journal of Advanced Computer Science and Applications.* 30 (30).

Dewa, Z. and Maglaras, L. A. (2016). Data Mining and Intrusion Detection Systems. *International Journal of Advanced Computer Science and Applications.* 30 (30).

Domingos, P. (2012). A Few Useful Things to Know About Machine Learning. *Communication of the ACM*, 55(10).

Dong, D., Ye, Z., Su, J., Xie, S., Cao, Y., Kochan, R. (2020). *A Malware Detection Method Based on Improved Fireworks Algorithms and Support Vector Machine*, 2020 IEEE 15th International Conference on Advanced Trends in Radioelectronic Telecommunications and Computer Engineering (TCSET). DOI: 10.1109/TCSET49122.2020.235556

Eysenbach, G. (2006). What is E-Health: *Journal of Medical internet Research, 3(2):e20.* DOI: 10.2196/jmir.3.2.e20

Fan, M., Ezeudoka, B.C. and Qalati, S. A. (2020). Exploring the Resistant to E-Health Services in Nigeeria: An Integrative Model Based pon the Theory of Planned Behavior and Stimulus-Organism.Response. *Humanities and Social Sciences Communication.* https://doi.org/10.1057/s41599-024-03090-6

Feizollah, A.., Anuar, N. B.., Salleh, R., Wmalina, F., Maarof, R. U. R. and Shamshirb, S. (2014). A Study of Machine Learning Classifiers for Anomaly-based Mobile Botnet Detection. *Malaysia Journal of Computer Science, 26(4):*251 – 256

Fix, E. and Hodges, J.L. (1951). Discriminatory Analysis. Non-parametric Discrimination; Consistency Properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, TX, USA.

Freund, Y. and Schapire, R.E. (1996). *Experiments with new boosting algorithm.* Proceedings of the Thirteenth International Conference on Machine Learning. Pp. 148 – 156

Garg, S and Paliyan, N. (2019). A Novel Parallel Classifier Scheme for Vulnerability Detection in Android. *Computer Electronic Engineering, 77(2019):12 – 26.* *https://doi.org/10.1016/j.ccompdeceng.2019.04.019*

GitHub Inc. (2020). *NSL-KDD Dataset*. https://www.github.com

GoodworkLabs (2018). How to Choose the Right Machine Learning Algorithm, *Machine learning (Blog).* 7100 Stevenson Blvd, Fremont CA, 94538, US

Grandini, M., Ragli, E. and Visani, O. (2020). Metrics for Multi-class Classification: An Overview, arXiv Preprint arXiv: 2008.05756

GSMA (2019). *Mobild Telecommunications Security Threat Landscape*. Floor 2, The Walbrook Building, 25 Walbrook, London EC4N 8AF United Kingdom.

Hagan.T. Demuth, H.B., Beale, M. (1996). *Neural Network design,* PWS Publishing Co. Boston MA: USA.

Harlalka, R. (2018). *Choosing the Right Machine Learning Algorithm*. @RajatHarlalka. (Retrieved on 15th March, 2021)

Hathaliya, J.J. and Tanwar, S. (2020). An Exhaustive Survey on Security and Privacy Issues in Healthcare 4.0. *Computer Communications,153(2020):311 – 335*

Health and Human Services (HHS) (2022). Electronic Medical Records in Healthcare. *Leadership for IT Security and Privacy. Across HHS.*

Heidari, M (2017). In Weld Defect Detection on Digital X-radiography Images, University of Oklahoma.

Heinold, B. (2016). *A Practical Introduction to python Programming*, http://www.brainheinold.net/python/A_Practical_Introduction_to_Python_Programming_Heinold.pdf (Retrieved 15th February, 2018).

Hira, Z.M. and Gillies, D.F. (2015). A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data. *Advances in Bioinformatics*. http://dx.doi.org/10.1155/2015/198363

Holland, S. M. (2019). *Principal Component Analysis (PCA)*. Department of Geology, University of Georgia, Atbens; GA 30602-2501.

Ibrahim, J., Danlandi, T.A., and Aderinola, M, (2017). *Comparative analysis between wired and wireless technologies in communication*: A review. Proceedings of 99th The IIER International Conference, Mecca, Saudi Arabia.

ICCC FBI (2021). Internet Crime Report. (Retrieved May 20, 2024). https://www.ic3.gov/media/PHD/AnnualReport

Idris, I., Adeleke, I., Uduimoh, A.A. and Tom, J.J. (N.D). Design Framework of Cyber Security Solutions to Threats and Attacks on Critical Infrastructure of Electricity Power Systems of Nigeria Companies. *Elizade University, Ilara, Mukin, Nigeria.*

Ikram, S.T. and Cherukuri, A.K. (2016). Improving Accuracy of Inttrusion Detection Model Using PCA and Optimized SVM. *CIT Journal of Computing and Information Technology,* 24 (2):133 – 148.

Inayat, Z; Gani, A., Anuar, N. B; Khan, M. K., and Anwar, S. (2016). Intrusion Response System: Foundations, design, and Challenges. ELSEVIER *Journal of Network and Computer Applications*, 62):53 – 74.

Ishaku, T. (2011). Water Supply Dilemma in Nigerian Rural Communities: Looking Towards the sky for an Answer. *Journal of Water Resources and Protection, 03(08):593 – 606.* DOI: 10.4236/jwarp.2011.38069

Jenyo, I., Amusan, E. A. and Emuoyibo-farhe, J. O. (2023). A Trust Management System for the Nigerian Cyber-Health Community. *International Journal of Information Technology and Computer Science, 1: 9 – 20.* DOI: 10.5815/ijitcs.2023.01.02

Jiang, X., Mao, B., Guan, J., Huang, X. (2020). Android Malware Detection Using Fine-Grained Features. Hindawi *Scientific Programming,* vol. 2020. https://doi.org/10.1155/2020/51901.38

Jolliffe, I. T. and Cadima, J. (2016). *Principal component analysis: A review and recent developments*. Philosophical Transactions. http://dx.doi.org/10.1098/rsta/2015.0202

Katiyr, N, Tripathis, S., Mumar, P., Verman, S., Sahu, A. K. and Saxena, S. (2024). AI and Cyber security: Enhancing Threat Detection and Response with Machine Learning. *Education Administration, Theory and Practice*, 30(4):6273 – 6282.

Kohavi, R (1996). *Scaling up the accuracy of naïve-bayes classifiers: A decision-tree hybrid*. KDD'96: Proceedings of the second Internation Conference on Knowledge Discovery and Data Mining, pp. 202 – 207

Kotsiantis, S. B., Zaharakis, I, and Pintelas, P. (2007). Supervised Machine Learning: A Review of Classification Techniques. *Emerging Artificial Intelligence Applications in Computer Engineering,* 160(1): 2- 24

Kruegel, C. (N.D). Full System Emulation. Achieving Successful Automated Dynamic Analysis of Evasive malware. Chris@hosline.com

Kulkarni, H and Hughes, J. T. (2022). Worker Equality in Renal Medicine. *Internal Medicine Journal,* 52(11): 1859 – 1862.

Kumar, D. A. and Das, S. K. (2023). Machine Learning Approach for malware Detection and Classirication Using Malware Analysis Framework. *International Journal of Intelligent Systems and Application in Engineering 9OKOSAE), 11(1): 330 – 335*

Kushala, M.V. AND Slaylaja, B. S. (2020). Recent Trends on Security Issues in Multi-Cliud Computing: A Survey. *In 2020 International Conference on Smart Electronics and Communications (ICOSEC), pp 777 -781, IEEE.*

Lee, T. and Mody, J. J. (2006) Behavioral Classification. *EICAR Conference, USA, 45(4): 1 – 17*

Lord, N. (2020). *Principle of Least Privilege,* Digital Guardian, Waltham, MA 02451. http://www.digitalguardian.com/author/nate-lord

Markoulidakis, I., Kopsiaftis, G., Rallis, I. and Georgoulas, I (2021). Multi-class Confusion Matrix Reduction Method and its Application on net Promoter Score Classification Problem. *In the 14th Pervasive Technologies Related to Assistive Environments Conference (pp. 412 – 419)*

Mary Stella J. and Kumar, S. (2020). Prediction and Comparison Using Adaboost and ML Algorithms with Austictic Children Dataset. *International Journal of Engineering, Research and Technology* (IJERT), 9 (7).

Mondal, A., Paul, S., Goswami, R. T. and Nath, S. (2020). Cloud Computing Security Issues and Challenges: A Review. *In IEEE, pp. 15*

Moshiri, E., Abdullah, A. B., Azlina, R., Mahmood, B. R., and Muda, Z. (2017). Malware Classification Framework for Dynamic Analysis Using Information Theory. *Indian Journal of Science and Technology,* 10(21):1 – 14 DOI: 10.17485/ijst/2017/v10i21/100023

Narudin, F. A., Feizollah, A., Anuar, N. B., and Gani, A. (2014) Evolution of machine learning. *Springer-verlag,* Berlin, Heidelberg.New York NY 10001, US: Infobase publishing, p. 27

Nasteski, V. (2017). *An Overview of Supervised Machine Learning Methods*. HORIZONS. B, 4:51–52. DOI: 10.20544/HORIZONS.B. 04.1.17. P05.

Nguyen, T., McDonald, J. T., and Glisson, W. B. (2017). *Exploitation and detection of a malicious mobile application.* Proceedings of the 50th Hawaii International Conference on System Sciences. http://hdl.handle.net/10125/41911.

Nykanen, P. (2017). Implementation and Evaluation of E-Health Ecosystems in T-sided-Markets. *E-Health Two Sided Markets. Elservia Inc.*

Okediran, O., Sijuade, A., wahab, W., and Oladimeji, A. (2022). A Framework for a Cloud-based Electronic Health Record System for Nigeria. *LAUTECH Journal of Engineering and Technology, 1692):128 – 136*

Pawar, M.V. and Anuradha, J. (2015). *Network security and types of attacks in network.* International Conference on Litelligent Computing, Communication and

Convergence (ICCC – 2015), vol.48, pp 503 – 506. DOI: 10.1016/j.procs. 2015.04.126.

Pham, N.T., Foo, E., Suriadi, S., Jeffrey, H and Lahza H.F. (2018). *Improving performance of intrusion detection systems using ensemble methods and features Selection.* In (ACSW) Australian Computer Science Week, Brisbane, OID Australia. https://dio.org/10.1145/3167918.3167951

Pooja, M. R. and Pushpalatha, M.P. (2019). A Comparative Performance Evaluation of Hybrid and Ensemble Machine Learning Models. *.Journal of Health & Medial Informatics,* vol.10, Issue 2. ISSN 2157 – 7420

Prasath, V.B.S., Alfeilat, H.A.A., Lasassmeh, O., and Hassanat, A.B.A. (2017). Distance and Similarity Measures Effect on the Performance of K-Nearest Neighboir Classifier – A Review. arXiv:1708.04321v1[cs.LG] Artificial Intelligence(cs AI). https://doi.org/10.48550/arXiv.1708.04321

Ravanshad, A. (2018). *How to Choose Machine Learning Algorithm.* https://medium.com/@aravanslad/how-to-choose-machine-learning-algorithm-9a2a448e0df (Retrieved on 15th February, 2020).

Raymond, A., Schubauer, J. and Madappa, D. (2020). Over-Privileged Permissions: Using Technology and Design to Create Legal Compliance. *Journal of Business and Technology,* Kelley School of Business Research Paper No. 2020-57, Available at SSRN:
https://ssrn.com/abstract=3546518 or http://dx.doi.org/10.2139/ssrn.3546518

Resnik, D. B. (2020). What si Ethics in Research and Why is it Important? *National Institute of Environmental Health Sciences, pp 22*

Saeed, V. A. and Asaad, R.R.(2022) Cyber Security, Threats, Vulnerability, Challenges with Proposed Solution. *Applied Computing Journal, 2(4):227 – 244..* https://doi.org/10.52098/acj.202260

Saini, N., Kasaragod, V. B., Prakasha, K., Das, A. K. (2023). A Hybrid Machine Learning Models for Detecting APT Attacks Based on Network Behavior Anomaly Detection. *Concurrency and Computation: Practice and Experience Published by John Wiley and Sons.* https://dio.org/10.1002/cpe.7865

Salvador-Meneses, J., Ruiz-Chavez, Z., and Gracia-Rodriquez, J. (2019). Compressed KNN: K-Nearest Neighbor with Data Compression. Entropy. 21(2):34. DOI: 10.3390/e21030234

Santos, I., Devesa, J., Brezo, F., Nieves, J., Bringas, P. G. (2013). OPEM: A Static-Dynamic Approach for Machine Learning-Based Malware Detection. In: Harrero, A., Conference CISIS'12-ICEUTE'12-SOCO'12 Special Sessions. *Advances in Intelligent Systems and Computing, vo.189* Springer, Berlin, Heidelberg. https://dio.org/10.1007/987-3-642-33018-6-28

Saperito, G. (2019). A Deeper Dive into the NSL-KDD Dataset. *Towards Data Science.* (Retrieved May 28, 2023). https://towardsdtascience.com/a-deeper-dive-into-the-nsl_kdd-data-set-15c753364657

Sardi, A., Rizzi, A., Sorano, E. and Guerrien, A. (2020). Cyber Risk in Health Facilities: A Systematic Literature Review. *Sustainability,* 12, 7002.

Sharma, N., Oriaku, E.A., Oriaku, N. (2020). Cost and Effects of Data Breaches, Precautions and Disclosure Laws. *Intgernational Journal of Emerging Trends in Social Sciences,* vol. 8, Issue 1., pp. 33 – 41. DOI: 10.20448/2001.81.55.41

Shatnawi, A. S., Yassen, Q. and Yateem, A. (2022). An Android Malware Detection Approach Based on Static Feature Analysis Using Machine Learning Algorithms. *Procedia Computer Science, 201(c): 653 – 658. https://doi.org/10.1016/j.procs.2022.03.086*

Shi, H., Wang, W., Wu, P., Wang, D. (2020). *Support vector machine based on localized multiple kernel learning in pre-microRNA classification.* Proceedings of the 2nd International Conference on Electrical, Communication and Computer Engineering (ICECCE), Instabul, Turkey.

Singh, A and Chatterjee, K. (2019). Security and Privacy Issues of Electronic Healthcare System: A Survey. *Journal of Information and Optimization Sciences,* 40(8):1709 – 1729. DOI: 10.1080/02522667.2019.1703265

Singh, A. (2018). *A Comprehensive Guide to Ensemble Learning (with Python Code).* Analytics, Vidhya.(Retrieved on5th January, 2020)

Singh, A. K., Wadhwa, G., Ahuja, M., Soni, K., and Sharma, K (2020). Android Malware Detection Using LSI-based Reduced Opcode Feature Vector. *Procedia Computer Science, 173(2019)(2020):291 – 298.* https://doi.org/10.1016/j.procs.2020.06.034.

Singh, B. K., Verma, K., and Thoke, A. S. (2015). Investigation on Impact of Feature Normalization Techniques on Classifier's Performance in Breast Tumor Classification. *International Journal of Computer Applications* (0975 – 8887), 116 (19).

Singhal, A. and Cowie, M. (2020). What is E-Health? *E-Journal of cardiology Practices, 18(24):1 – 10*

Smmarwar, S. K., Gupta, G. P. and Kumar, S. (2024). Android Malware Detection and Identification Framework by Leveraging the Machine and Deep Learning Techniques: A Comprehensive Review. *Telematics and Informatica Report, 14(2024):100130*

Stein, J. (2020). *Data Breach Report,* Nortyh Carolina Department of Justice. www.ncdoj.gov/complaint

Sullivan, D.T. (2015). *Survey of Malware Threats and Recommendations to Improve Cyber Security for Industrial Control Systems* version 1.0. US Army Research Laboratory (ARL)

Sun, L., Li, Z., Yan, Q., Srisa-an, W., and Pan, Y. (2016). *SigPID: Significant Permission Identification for Android Malware Detection*. University of Nebraska, Lincoln, NE 68588.

Syed, A., Purushotham, K. and Shidagani, G. (2020). Cloud Storage Security Risks, Prctice and Meassures: A Review. *IEEE International Conference for Innovation in Technology (INOCON). https://doi.org.sdl.idm.oclc.org/10.1109/INOCON50539.2020.9298281*

Tatam, M., Shanmugam, B., Azam, S., and Kannoorpatti, S. A. K. (2021). A Review of Threat Modelling Approaches for APT Style Attacks. *Heliyon 7920210e05969.* https://doi.org/10.1016/j.heliyon.2021.e05909.

Vasiliadis, G., Polychronakis, M., Joannidis, S. (2014). "GPU-Assisted Malware", *International. Journal of. Information. Security*. vol. 14. No. 3. pp. 289 – 297.

Vemprala, N. and Dietrich, G. (2019). *A social network analysis (SNA) study on data breach concerns over social media*. Proceedings of the 52nd Hawaii International Conference on System Sciences. https://hdl.hanbdle.net/10125/60155.

Verizon (2018). 2018 *Data Breach Investigations Report*. www.verizonenterprise.com/Federal (Retrieved on 4 May, 2021).

Verizon (2022). Data Breach Investigations Report. (Retrieved May 20, 2024). https;//www.verizon.com/busness/resources/report

Wallisch, P. (2014). *MATLAB for Neuroscientists*. An Introduction to Scientific Computing in MATLAB. In MATLAB for Neuroscientists (Second Edition).

Wang, L., Gao, Y., Gao, S. and Yong, X. (2021). A New Feature Selection Method Based on a Self-variant Genetic Algorithm Applied to Android Malware Detection. *Symmetry, 13(17):1 – 21* https://doi.org/10.3390/sym.13071290

Wang, P. and Wang, Y. (2015). "Malware Behavioral Detection and Vaccine Development by Using a Support Vector Machine Model Classifier", *Journal of computer and system sciences.* 81:1012 – 1026

Wang, P. and Wang, Y. (2015). "Malware Behavioral Detection and Vaccine Development by Using a Support Vector Machine Model Classifier", *Journal of computer and system sciences.* 81: 1012 – 1026.

Wesolowski, T. E., Porwik, P. and Dorozi, R (2016). Electronic Health Record Security Based on Ensemble Classification of Keystroke Dynamics. *International Journal of Applied Artificial Intelligence,* 30(6):521 – 540. DOI: 10.1080/08839514.2016.1193715

Wu, W. C. and Hung, S. H. (2014). DroidDolphin: A Dynamic Android Malware Detection Framework Using Big Data and Machine Learning. *In: Proceedings of the 2014 Conference on Research in Adaptive and Convergent Systems, pp. 247 – 252.*

Yeng, P. K., Wulthusen, S. D., and Yang, B (2020) Comparative Analysis of Threat Modeling Methods for Cloud Computing Towards Healthcare Security Practices. *International Journal of Advanced Computer Science and Applications.* 11(11):772-784

Yeng, P. K.., Nweke, L. O., Yang,B., Fauzi, M. A., and Snekkenes, E. A. (2021). Artificial Intelligence-Based Framework for Analyzing Health Care Staff Security practice: Mapping, Review and Simulation Study. *JMIR Medical Informatics*, 9(120: e19250. https://medinform.jmir.org/2021/12/e19250

Yesilyurt, M and Yalman, Y. (2016). Security Threats on Mobile Devices and Their Effects; Estimation for the Future. *International Journal of Security and its Applications*. 10(2):13 – 26. http://dx.doi.org/10.14257/ljsla.2016.10.2.02

Yin, H. and Song, D. (2019). *Whole-System Fine-grained Taint Analysis for Automatic Malware Detection and Analysis*. https://www.researchgate.net/publication/2287857-54. (Retrieved on 10 March, 2021)

Yoti, J. and Saini, H. (2017). A Study on Networks and Comparison of Wired, Wireless and Optical Networks. *International Journal of Innovative Research in Computer and Communication Engineering,* 5(3)

Zhang, J. (2016). Introduction to Machine Learning: K-Nearest Neighbors. Annals of Translational Medicine, vol. 4(11):218. DOI: 10.21037/atm.2016.03.37.

APPENDIX I

NSL-KDD TRAINING DATASET

0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,150,25,0.17,0.03,0.17,0.00,0.00,0.00,0.05,0.00,normal,20

0,udp,other,SF,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,1,0.00,0.00,0.00,0.00,0.08,0.15

,0.00,255,1,0.00,0.60,0.88,0.00,0.00,0.00,0.00,0.00,normal,15

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,123,6,1.00,1.00,0.00,0.00,0.05,0.07,

0.00,255,26,0.10,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,http,SF,232,8153,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,5,0.20,0.20,0.00,0.00,1.00,0.0

0,0.00,30,255,1.00,0.00,0.03,0.04,0.03,0.01,0.00,0.01,normal,21

0,tcp,http,SF,199,420,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,30,32,0.00,0.00,0.00,0.00,1.00,0.

00,0.09,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,121,19,0.00,0.00,1.00,1.00,0.16,0.

06,0.00,255,19,0.07,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,166,9,1.00,1.00,0.00,0.00,0.05,0.06,

0.00,255,9,0.04,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,117,16,1.00,1.00,0.00,0.00,0.14,0.0

6,0.00,255,15,0.06,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,remote_job,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,270,23,1.00,1.00,0.00,0.00,0.09

,0.05,0.00,255,23,0.09,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,133,8,1.00,1.00,0.00,0.00,0.06,0.06,

0.00,255,13,0.05,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,205,12,0.00,0.00,1.00,1.00,0.06,0.

06,0.00,255,12,0.05,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,199,3,1.00,1.00,0.00,0.00,0.02,0.06,
0.00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21
0,tcp,http,SF,287,2251,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0.00,0.00,0.00,0.00,1.00,0.0
0,0.43,8,219,1.00,0.00,0.12,0.03,0.00,0.00,0.00,0.00,normal,21
0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,2,20,1.00,0.00,1.00,0.20,0.00,0.00,0.00,0.00,warezclient,15
0,tcp,name,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,233,1,1.00,1.00,0.00,0.00,0.00,0.06,0
.00,255,1,0.00,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19
0,tcp,netbios_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,96,16,1.00,1.00,0.00,0.00,0.17,0
.05,0.00,255,2,0.01,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18
0,tcp,http,SF,300,13788,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,9,0.00,0.11,0.00,0.00,1.00,0.
00,0.22,91,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21
0,icmp,eco_i,SF,18,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,1,16,1.00,0.00,1.00,1.00,0.00,0.00,0.00,0.00,ipsweep,18
0,tcp,http,SF,233,616,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,66,255,1.00,0.00,0.02,0.03,0.00,0.00,0.02,0.00,normal,21
0,tcp,http,SF,343,1178,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,10,0.00,0.00,0.00,0.00,1.00,0.
00,0.20,157,255,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,mtp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,223,23,1.00,1.00,0.00,0.00,0.10,0.05,0.
00,255,23,0.09,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,280,17,1.00,1.00,0.00,0.00,0.06,0.0
5,0.00,238,17,0.07,0.06,0.00,0.00,0.99,1.00,0.00,0.00,neptune,18
0,tcp,http,SF,253,11905,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,10,0.00,0.00,0.00,0.00,1.00,0
.00,0.20,87,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21

5607,udp,other,SF,147,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00
,0.00,0.00,255,1,0.00,0.85,1.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,mtp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,248,2,1.00,1.00,0.00,0.00,0.01,0.06,0.0
0,255,2,0.01,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19
507,tcp,telnet,SF,437,14421,0,0,0,0,0,1,3,0,0,0,0,0,1,0,0,0,1,1,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,255,25,0.10,0.05,0.00,0.00,0.53,0.00,0.02,0.16,normal,20
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,279,7,1.00,1.00,0.00,0.00,0.03,0.06,
0.00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21
0,tcp,http,SF,227,6588,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,22,0.00,0.00,0.00,0.00,1.00,0.
00,0.18,43,255,1.00,0.00,0.02,0.14,0.00,0.00,0.56,0.57,normal,21
0,tcp,http,SF,215,10499,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,14,14,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,241,1400,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,33,33,0.00,0.00,0.03,0.03,1.00,0
.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,12,0.00,0.00,0.00,0.00,1.00,0.00,
1.00,1,53,1.00,0.00,1.00,0.51,0.00,0.00,0.00,0.00,ipsweep,17
0,tcp,finger,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,57,16,1.00,1.00,0.00,0.00,0.28,0.07,
0.00,255,59,0.23,0.04,0.01,0.00,1.00,1.00,0.00,0.00,neptune,20
0,tcp,http,SF,303,555,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,9,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,9,255,1.00,0.00,0.11,0.01,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0.00,0.00,1.00,1.00,0.50,1.00,
0.00,255,1,0.00,0.31,0.28,0.00,0.00,0.00,0.29,1.00,portsweep,20
0,udp,domain_u,SF,45,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,181,181,0.00,0.00,0.00,0.00,1
.00,0.00,0.00,255,250,0.98,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,18

1,udp,private,SF,105,147,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,41,5,0.12,0.05,0.05,0.00,0.00,0.00,0.00,0.00,normal,19
0,udp,domain_u,SF,43,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,122,202,0.00,0.00,0.00,0.00,1
.00,0.00,0.01,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal,18
0,tcp,supdup,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,205,17,1.00,1.00,0.00,0.00,0.08,0.0
6,0.00,255,17,0.07,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19
0,tcp,http,SF,324,2302,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,22,28,0.00,0.00,0.00,0.00,1.00,0
.00,0.14,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,uucp_path,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,228,13,1.00,1.00,0.00,0.00,0.06,
0.07,0.00,255,13,0.05,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,20
0,tcp,ftp_data,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,50,11,1.00,1.00,0.00,0.00,0.22,0.0
8,0.00,255,63,0.25,0.02,0.01,0.00,1.00,1.00,0.00,0.00,neptune,19
0,tcp,Z39_50,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,262,12,1.00,1.00,0.00,0.00,0.05,0.
06,0.00,255,9,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18
2,tcp,smtp,SF,1591,372,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.0
0,1.00,181,147,0.81,0.02,0.01,0.00,0.00,0.00,0.00,0.00,normal,21
9052,udp,other,SF,146,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,2,0.00,0.00,0.00,0.00,0.50
,0.50,0.00,255,2,0.01,0.66,0.99,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,290,3006,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,csnet_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,108,11,1.00,1.00,0.00,0.00,0.10,0.
08,0.00,255,11,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18
0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,80,80,0.00,0.00,0.00,0.00,1.00,0.
00,0.00,255,80,0.31,0.02,0.31,0.00,0.00,0.00,0.00,0.00,teardrop,16

0,tcp,http,SF,255,861,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,44,44,0.02,0.02,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,4,38,1.00,0.00,1.00,0.18,0.00,0.00,0.00,0.00,warezclient,12

0,tcp,ftp_data,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,258,17,1.00,1.00,0.00,0.00,0.07,0.05,0.00,255,5,0.02,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,tcp,http,SF,302,498,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,10,0.00,0.00,0.00,0.00,1.00,0.00,0.00,131,255,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,61,4,0.00,0.00,1.00,1.00,0.07,0.10,0.00,255,4,0.02,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,2,0.01,0.02,0.01,0.00,0.00,0.00,0.77,0.00,teardrop,15

0,tcp,http,SF,220,1398,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,26,42,0.00,0.00,0.00,0.00,1.00,0.00,0.05,26,255,1.00,0.00,0.04,0.03,0.00,0.00,0.00,0.00,normal,21

0,udp,domain_u,SF,43,69,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,120,120,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,245,0.96,0.01,0.01,0.00,0.00,0.00,0.00,0.00,normal,18

0,udp,domain_u,SF,44,133,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,73,75,0.00,0.00,0.00,0.00,1.00,0.00,0.03,122,212,0.88,0.02,0.88,0.01,0.00,0.00,0.08,0.00,normal,21

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,15,0.00,0.00,0.00,0.00,1.00,0.00,1.00,2,46,1.00,0.00,1.00,0.26,0.00,0.00,0.00,0.00,nmap,17

0,tcp,uucp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,135,9,1.00,1.00,0.00,0.00,0.07,0.06,0.00,255,11,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,20

0,tcp,finger,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,24,12,1.00,1.00,0.00,0.00,0.50,0.08,0.00,255,59,0.23,0.04,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,udp,domain_u,SF,43,43,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,148,228,0.00,0.00,0.00,0.00,1
.00,0.00,0.01,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal,18
0,tcp,ftp_data,SF,641,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,175,48,0.25,0.02,0.25,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,206,4,0.00,0.00,1.00,1.00,0.02,0.0
6,0.00,255,4,0.02,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,175,1,0.10,0.00,0.89,1.00,0.01,1.0
0,0.00,255,1,0.00,0.84,0.00,0.00,0.07,0.00,0.62,1.00,satan,18
0,tcp,http,SF,310,2794,0,0,0,0,1,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,63,10,0.00,0.00,1.00,1.00,0.16,0.0
8,0.00,255,10,0.04,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,smtp,SF,696,333,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00
,0.00,109,133,0.39,0.04,0.01,0.02,0.00,0.00,0.00,0.00,normal,21
0,tcp,netbios_dgm,RSTR,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.
00,0.00,0.00,236,1,0.00,0.58,0.58,0.00,0.00,0.00,0.58,1.00,portsweep,14
0,tcp,http,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,52,19,1.00,1.00,0.00,0.00,0.37,0.08,0.0
0,255,67,0.26,0.02,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19
0,tcp,netbios_dgm,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,125,7,0.00,0.00,1.00,1.00,0.
06,0.06,0.00,255,6,0.02,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,20
0,tcp,supdup,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,168,21,1.00,1.00,0.00,0.00,0.12,0.0
6,0.00,255,21,0.08,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19
0,tcp,http,SF,221,2878,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,21,58,1.00,0.00,0.05,0.03,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,145,16,1.00,1.00,0.00,0.00,0.11,0.0
6,0.00,255,31,0.12,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18
0,icmp,urp_i,SF,181,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,2,0.00,0.00,0.00,0.00,0.67,0.67
,0.00,255,93,0.36,0.01,0.46,0.00,0.00,0.00,0.00,0.00,normal,21
0,udp,domain_u,SF,42,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9,14,0.00,0.00,0.00,0.00,1.00,
0.00,0.21,84,83,0.99,0.02,0.01,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,138,6,1.00,1.00,0.00,0.00,0.04,0.06,
0.00,255,6,0.02,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21
0,tcp,ftp_data,SF,12,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,44,47,0.55,0.07,0.55,0.04,0.00,0.00,0.00,0.00,normal,20
0,tcp,auth,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,276,9,1.00,1.00,0.00,0.00,0.03,0.06,0.
00,255,9,0.04,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19
0,tcp,http,SF,210,2790,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,27,0.00,0.00,0.00,0.00,1.00,0
.00,0.11,48,255,1.00,0.00,0.02,0.09,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,321,5683,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,10,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,168,255,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,248,8,0.00,0.00,1.00,1.00,0.03,0.0
6,0.00,255,8,0.03,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,http,SF,329,3982,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,31,0.00,0.00,0.00,0.00,1.00,0.
00,0.10,24,255,1.00,0.00,0.04,0.05,0.00,0.00,0.00,0.00,normal,21
0,tcp,uucp_path,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,135,7,0.00,0.00,1.00,1.00,0.05,
0.07,0.00,255,7,0.03,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,19
0,tcp,domain,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,220,19,1.00,1.00,0.00,0.00,0.09,0.0
5,0.00,255,2,0.01,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,44,0.00,0.00,0.00,0.00,1.00,0.00,
1.00,1,95,1.00,0.00,1.00,0.51,0.00,0.00,0.00,0.00,ipsweep,15

0,tcp,http,SF,320,379,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,9,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,123,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,212,7,1.00,1.00,0.00,0.00,0.03,0.05,
0.00,72,7,0.10,0.11,0.01,0.00,0.97,1.00,0.00,0.00,neptune,18

0,tcp,http,SF,225,3762,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,10,10,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,ftp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,250,13,1.00,1.00,0.00,0.00,0.05,0.06,0.
00,255,1,0.00,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,132,15,0.00,0.00,1.00,1.00,0.11,0.
05,0.00,255,15,0.06,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

315,udp,other,SF,146,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,2,0.00,0.00,0.00,0.00,0.67,
0.67,0.00,117,2,0.02,0.54,0.78,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,247,1416,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,156,255,1.00,0.00,0.01,0.09,0.00,0.00,0.46,0.61,normal,21

0,tcp,http,SF,353,1483,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,9,0.00,0.00,0.00,0.00,1.00,0.0
0,0.33,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,udp,domain_u,SF,44,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,109,189,0.00,0.00,0.00,0.00,0
.99,0.02,0.01,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,normal,18

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,146,18,0.00,0.00,1.00,1.00,0.12,0.
05,0.00,255,10,0.04,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,csnet_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,300,6,1.00,1.00,0.00,0.00,0.02,0.0
6,0.00,255,3,0.01,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

2,tcp,smtp,SF,3065,331,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.0
0,1.00,243,119,0.48,0.02,0.00,0.02,0.01,0.02,0.00,0.00,normal,21
0,udp,other,SF,102,102,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,7,1,0.14,0.29,0.14,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,259,750,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,5,0.00,0.00,0.00,0.00,1.00,0.00,
0.40,4,255,1.00,0.00,0.25,0.03,0.00,0.00,0.00,0.00,normal,21
1082,udp,other,SF,147,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00
,0.00,0.00,255,2,0.01,0.42,0.86,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,129,16,1.00,1.00,0.00,0.00,0.12,0.0
5,0.00,255,12,0.05,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21
0,tcp,http,SF,0,354,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,32,32,0.03,0.03,0.00,0.00,1.00,0.00,
0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,20
0,tcp,http,SF,309,306,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,66,255,1.00,0.00,0.02,0.02,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,1.00,1.00,1.00,0.00,1.0
0,1,247,1.00,0.00,1.00,0.15,0.00,0.00,1.00,0.68,normal,21
0,tcp,http,SF,278,7227,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,153,255,1.00,0.00,0.01,0.05,0.00,0.00,0.00,0.00,normal,21
0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,82,82,0.00,0.00,0.00,0.00,1.00,0.
00,0.00,255,182,0.71,0.29,0.71,0.00,0.09,0.00,0.20,0.00,teardrop,16
0,tcp,http,SF,313,761,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,43,255,1.00,0.00,0.02,0.03,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,219,1212,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,15,15,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,242,255,1.00,0.00,0.00,0.01,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,245,9165,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,3,0.00,0.00,0.00,0.00,1.00,0.0

0,0.67,10,255,1.00,0.00,0.10,0.05,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,328,2243,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,21,21,0.00,0.00,0.00,0.00,1.00,0

.00,0.00,21,255,1.00,0.00,0.05,0.04,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.00,0.00,0.0

0,23,249,1.00,0.00,0.04,0.10,0.00,0.00,1.00,0.96,normal,21

0,tcp,http,SF,251,301,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0.00,0.00,0.00,0.00,1.00,0.00,

0.29,255,252,0.99,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,smtp,SF,2089,335,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,0.50,1.0

0,1.00,255,152,0.60,0.03,0.00,0.00,0.01,0.02,0.00,0.01,normal,20

0,tcp,http,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.00,0.00,0.0

0,6,255,1.00,0.00,0.17,0.21,0.00,0.00,1.00,0.88,normal,21

0,tcp,http,SF,237,511,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,

0.00,3,255,1.00,0.00,0.33,0.01,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,324,431,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,5,0.00,0.00,0.00,0.00,1.00,0.00,

0.40,107,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21

25950,tcp,private,RSTR,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,1.00,1.00,1.0

0,0.00,0.00,255,2,0.01,0.69,1.00,0.00,0.00,0.00,1.00,1.00,portsweep,15

0,tcp,Z39_50,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,110,17,0.00,0.00,1.00,1.00,0.15,0

.05,0.00,255,17,0.07,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,18

0,udp,private,SF,105,145,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,1,0.00,0.00,0.00,0.00,0.25,0

.50,0.00,255,239,0.94,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,18

0,tcp,http,SF,218,974,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.00,0.00,

0.00,6,255,1.00,0.00,0.17,0.02,0.00,0.01,0.00,0.00,normal,21

0,tcp,http,SF,325,2649,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,7,7,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,107,255,1.00,0.00,0.01,0.01,0.00,0.00,0.00,0.00,normal,21

0,tcp,uucp_path,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,246,20,0.00,0.00,1.00,1.00,0.0

8,0.07,0.00,255,20,0.08,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,19

0,tcp,bgp,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,272,19,0.00,0.00,1.00,1.00,0.07,0.07,

0.00,255,19,0.07,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,19

0,tcp,csnet_ns,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,259,8,1.00,1.00,0.00,0.00,0.03,0.0

7,0.00,255,8,0.03,0.08,0.00,0.00,1.00,1.00,0.00,0.00,neptune,20

240,tcp,http,SF,328,275,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,9,10,0.00,0.00,0.00,0.10,1.00,0

.00,0.20,255,250,0.98,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,bgp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,294,6,1.00,1.00,0.00,0.00,0.02,0.05,0.0

0,255,8,0.03,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,18

0,tcp,finger,SF,9,138,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,

0.00,195,10,0.03,0.03,0.01,0.20,0.01,0.00,0.01,0.00,normal,21

0,udp,domain_u,SF,46,46,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,96,178,0.00,0.00,0.00,0.00,1.

00,0.00,0.02,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,normal,18

1,tcp,smtp,SF,1079,334,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,138,167,0.57,0.03,0.01,0.01,0.00,0.00,0.00,0.00,normal,21

0,udp,domain_u,SF,45,139,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,79,159,0.00,0.00,0.00,0.00,1

.00,0.00,0.01,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,19

26,tcp,ftp,SF,273,903,0,0,0,5,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,

0.00,176,49,0.28,0.02,0.01,0.00,0.01,0.02,0.00,0.00,normal,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,190,20,1.00,1.00,0.00,0.00,0.11,0.0

6,0.00,255,20,0.08,0.08,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

Figure 1: NSL-KDD training dataset

Source: Github (2020)

APPENDIX II

NSL-KDD TEST DATASET

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,229,10,0.00,0.00,1.00,1.00,0.04,0.06,0.00,255,10,0.04,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,136,1,0.00,0.00,1.00,1.00,0.01,0.06,0.00,255,1,0.00,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

2,tcp,ftp_data,SF,12983,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,134,86,0.61,0.04,0.61,0.02,0.00,0.00,0.00,0.00,normal,21

0,icmp,eco_i,SF,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,65,0.00,0.00,0.00,0.00,1.00,0.00,1.00,3,57,1.00,0.00,1.00,0.28,0.00,0.00,0.00,0.00,saint,15

1,tcp,telnet,RSTO,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,8,0.00,0.12,1.00,0.50,1.00,0.00,0.75,29,86,0.31,0.17,0.03,0.02,0.00,0.00,0.83,0.71,mscan,11

0,tcp,http,SF,267,14515,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,155,255,1.00,0.00,0.01,0.03,0.01,0.00,0.00,0.00,normal,21

0,tcp,smtp,SF,1022,387,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,3,0.00,0.00,0.00,0.00,1.00,0.00,1.00,255,28,0.11,0.72,0.00,0.00,0.00,0.00,0.72,0.04,normal,21

0,tcp,telnet,SF,129,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.01,0.01,0.02,0.02,guess_passwd,15

0,tcp,http,SF,327,467,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,33,47,0.00,0.00,0.00,0.00,1.00,0.00,0.04,151,255,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,normal,21

0,tcp,ftp,SF,26,157,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,52,26,0.50,0.08,0.02,0.00,0.00,0.00,0.00,0.00,guess_passwd,7

0,tcp,telnet,SF,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,128,0.50,0.01,0.00,0.00,0.00,0.00,0.66,0.32,mscan,9

0,tcp,smtp,SF,616,330,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.00
,1.00,255,129,0.51,0.03,0.00,0.00,0.00,0.00,0.33,0.00,normal,18
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,111,2,0.00,0.00,1.00,1.00,0.02,0.0
7,0.00,255,2,0.01,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,telnet,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,120,120,1.00,1.00,0.00,0.00,1.00,0.0
0,0.00,235,171,0.73,0.07,0.00,0.00,0.69,0.95,0.02,0.00,neptune,18
37,tcp,telnet,SF,773,364200,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,38,73,0.16,0.05,0.03,0.04,0.00,0.77,0.00,0.07,normal,14
0,tcp,http,SF,350,3610,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,71,255,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,213,659,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,24,24,0.00,0.00,0.00,0.00,1.00,0.
00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,246,2090,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,16,16,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,35,255,1.00,0.00,0.03,0.05,0.00,0.00,0.00,0.00,normal,21
0,udp,private,SF,45,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,505,505,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00,normal,15
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,204,18,0.00,0.00,1.00,1.00,0.09,0.
07,0.00,255,18,0.07,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,ldap,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,118,19,0.00,0.00,1.00,1.00,0.16,0.05,
0.00,255,19,0.07,0.05,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,pop_3,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.0
0,255,87,0.34,0.01,0.01,0.00,1.00,1.00,0.00,0.00,mscan,18
0,tcp,http,SF,196,1823,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,17,17,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,277,1816,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,17,18,0.00,0.00,0.00,0.00,1.00,0
.00,0.11,36,255,1.00,0.00,0.03,0.02,0.00,0.00,0.00,0.00,normal,21

0,tcp,courier,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,116,8,0.00,0.00,1.00,1.00,0.07,0.0
7,0.00,255,8,0.03,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,18

0,tcp,discard,RSTO,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,273,13,0.00,0.00,1.00,1.00,0.05
,0.06,0.00,255,13,0.05,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,20

0,tcp,http,SF,294,6442,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,22,46,0.00,0.00,0.00,0.00,1.00,0
.00,0.11,180,255,1.00,0.00,0.01,0.01,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,300,440,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,7,7,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,icmp,ecr_i,SF,520,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,46,59,1.00,0.00,1.00,0.14,0.00,0.00,0.00,0.00,smurf,18

0,udp,private,SF,54,51,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,255,255,1.00,0.00,0.83,0.00,0.00,0.00,0.00,0.00,normal,17

805,tcp,http,RSTR,76944,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,12,12,0.00,0.00,1.00,1.00,1.
00,0.00,0.00,241,238,0.99,0.01,0.00,0.00,0.00,0.00,0.07,0.07,apache2,14

0,tcp,smtp,SF,720,281,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.00
,1.00,158,82,0.52,0.06,0.01,0.00,0.01,0.00,0.00,0.00,normal,21

0,tcp,http,SF,301,19794,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,15,15,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,20,255,1.00,0.00,0.05,0.02,0.05,0.00,0.00,0.00,normal,21

0,udp,private,SF,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,40,3,0.00,0.00,0.00,0.00,0.08,0.38,
0.00,255,3,0.01,0.58,0.99,0.00,0.00,0.00,0.01,0.00,normal,1

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,483,1,0.05,0.00,0.92,1.00,0.00,1.0
0,0.00,255,1,0.00,1.00,0.00,0.00,0.00,0.00,0.96,1.00,satan,19

0,tcp,imap4,RSTO,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,1.00,1.00,0.50,1.0
0,1.00,185,59,0.24,0.03,0.01,0.03,0.01,0.00,0.89,0.95,mscan,14
0,tcp,http,SF,209,12894,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
8,tcp,ftp,SF,220,688,0,0,0,4,0,1,0,0,0,0,4,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0
.00,53,27,0.51,0.08,0.02,0.00,0.00,0.00,0.00,0.00,buffer_overflow,1
0,udp,domain_u,SF,43,71,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,113,113,0.00,0.00,0.00,0.00,1
.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,20
0,tcp,ftp_data,SF,88382,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,15,15,0.00,0.00,0.00,0.00,1.0
0,0.00,0.00,203,114,0.38,0.01,0.38,0.02,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,144,8,1.00,1.00,0.00,0.00,0.06,0.06,
0.00,255,8,0.03,0.06,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21
0,tcp,http,SF,277,4968,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,13,13,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,13,255,1.00,0.00,0.08,0.01,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,321,2715,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,29,37,0.03,0.05,0.00,0.00,1.00,0
.00,0.08,29,255,1.00,0.00,0.03,0.04,0.03,0.00,0.00,0.00,normal,21
0,tcp,http,SF,335,3228,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,49,50,0.00,0.00,0.00,0.00,1.00,0
.00,0.04,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,mtp,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,266,8,0.00,0.00,1.00,1.00,0.03,0.06,0
.00,255,8,0.03,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,20
0,tcp,http,SF,234,3236,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,21,0.00,0.00,0.00,0.00,1.00,0.
00,0.10,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,systat,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,281,6,1.00,1.00,0.00,0.00,0.02,0.06,0
.00,255,6,0.02,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,19

0,tcp,http,SF,54540,8314,0,0,0,2,0,1,1,0,0,0,0,0,0,0,0,0,4,24,0.00,0.00,0.00,0.00,1.00,0.00,0.08,255,250,0.98,0.01,0.00,0.00,0.00,0.00,0.06,0.06,back,12

0,tcp,telnet,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,12,1.00,0.33,0.00,0.67,1.00,0.00,0.92,91,86,0.34,0.03,0.01,0.03,1.00,1.00,0.00,0.00,mscan,18

0,udp,private,SF,46,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,68,68,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.26,0.00,0.00,0.00,0.00,0.00,normal,14

0,tcp,http,SF,318,488,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,16,0.00,0.00,0.00,0.00,1.00,0.00,0.12,3,156,1.00,0.00,0.33,0.04,0.00,0.01,0.00,0.00,normal,21

0,tcp,http,SF,256,4926,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,41,0.00,0.00,0.00,0.00,1.00,0.00,0.07,255,241,0.95,0.02,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,iso_tsap,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,239,7,0.00,0.00,1.00,1.00,0.03,0.06,0.00,255,7,0.03,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,20

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,201,3,0.00,0.00,1.00,1.00,0.01,0.08,0.00,255,3,0.01,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

282,tcp,ftp,SF,160,599,0,0,0,2,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,37,0.15,0.02,0.00,0.00,0.00,0.00,0.44,0.00,warezmaster,13

0,tcp,http,SF,229,509,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,21,255,1.00,0.00,0.05,0.04,0.00,0.00,0.00,0.00,normal,21

0,tcp,imap4,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,280,8,0.00,0.00,1.00,1.00,0.03,0.06,0.00,255,8,0.03,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,18

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00,snmpgetattack,3

7428,tcp,telnet,SF,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,217,0.85,0.03,0.00,0.00,0.33,0.39,0.12,0.06,processtable,18

0,tcp,http,SF,282,396,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
4,tcp,pop_3,SF,28,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,255,249,0.98,0.02,0.00,0.00,0.00,0.00,0.00,0.00,guess_passwd,18
280,tcp,ftp_data,SF,283618,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.
00,0.00,0.00,7,10,0.86,0.29,0.86,0.20,0.00,0.00,0.00,0.00,warezmaster,9
0,tcp,http,SF,233,2239,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,255,197,0.77,0.02,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,257,8712,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,20,0.00,0.00,0.00,0.00,1.00,0.
00,0.20,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,other,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,412,1,0.15,0.00,0.83,1.00,0.00,1.00,
0.00,255,1,0.00,0.96,0.00,0.00,0.10,0.00,0.89,1.00,satan,17
0,tcp,http,SF,214,645,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,30,30,0.00,0.00,0.00,0.00,1.00,0.
00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,telnet,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,6,0.00,0.00,1.00,1.00,0.25,1.00,0.
83,106,85,0.29,0.05,0.01,0.02,0.02,0.00,0.91,0.67,mscan,15
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,109,13,0.00,0.00,1.00,1.00,0.12,0.
06,0.00,255,13,0.05,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,smtp,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,114,1,1.00,1.00,0.00,0.00,0.01,0.06,0.
00,255,1,0.00,0.05,0.00,0.00,1.00,1.00,0.00,0.00,neptune,20
0,tcp,http,SF,203,651,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,19,19,0.00,0.00,0.00,0.00,1.00,0.
00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,216,14,0.00,0.00,1.00,1.00,0.06,0.
06,0.00,255,14,0.05,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,smtp,SF,0,83,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.00,1.0
0,61,146,0.80,0.07,0.02,0.01,0.00,0.00,0.00,0.01,normal,21
0,tcp,http,SF,307,484,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,10,255,1.00,0.00,0.10,0.05,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,320,1667,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,30,0.00,0.00,0.00,0.00,1.00,0.
00,0.17,85,253,1.00,0.00,0.01,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,csnet_ns,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,251,3,0.00,0.00,1.00,1.00,0.01,0.
06,0.00,255,3,0.01,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,19
0,tcp,http,SF,215,430,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,40,255,1.00,0.00,0.03,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,358,3691,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,81,255,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,normal,21
8092,tcp,telnet,SF,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,255,67,0.26,0.03,0.00,0.00,0.00,0.01,0.11,0.19,processtable,8
0,tcp,http,SF,270,1332,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,7,255,1.00,0.00,0.14,0.04,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,337,283,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,24,29,0.00,0.00,0.00,0.00,1.00,0.
00,0.10,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,267,19,0.01,0.00,0.99,1.00,0.07,0.
06,0.00,255,19,0.07,0.06,0.00,0.00,0.01,0.00,0.99,1.00,neptune,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,231,6,0.00,0.00,1.00,1.00,0.03,0.0
7,0.00,255,6,0.02,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,icmp,ecr_i,SF,1480,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,1,25,1.00,0.00,1.00,0.52,0.00,0.00,0.00,0.00,pod,17

0,tcp,http,SF,219,1310,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,4,251,1.00,0.00,0.25,0.08,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,258,1012,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,26,26,0.00,0.00,0.00,0.00,1.00,0

.00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,324,1093,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,8,0.00,0.00,0.00,0.00,1.00,0.0

0,0.38,69,255,1.00,0.00,0.01,0.01,0.00,0.00,0.00,0.00,normal,21

0,tcp,finger,RSTO,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,58,19,0.00,0.00,1.00,1.00,0.33,0.

07,0.00,255,62,0.24,0.11,0.00,0.00,0.00,0.00,1.00,1.00,neptune,19

0,tcp,pop_3,RSTO,0,36,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,7,0.00,0.00,1.00,1.00,0.33,1.0

0,1.00,189,60,0.24,0.03,0.01,0.03,0.00,0.00,0.88,0.98,mscan,15

0,tcp,http,SF,236,10503,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,3,0.00,0.00,0.00,0.00,1.00,0.

00,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,ftp_data,SF,1699,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.

00,0.00,36,7,0.19,0.11,0.19,0.00,0.00,0.00,0.00,0.00,normal,20

0,tcp,http,SF,327,910,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,14,15,0.00,0.00,0.00,0.00,1.00,0.

00,0.13,14,255,1.00,0.00,0.07,0.03,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,303,2789,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,103,4,0.00,0.00,1.00,1.00,0.04,0.0

6,0.00,255,4,0.02,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,other,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.00,0.00,0.

00,255,13,0.05,0.02,0.00,0.00,0.00,0.00,0.06,1.00,httptunnel,17

3,tcp,pop_3,SF,28,93,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,

0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,guess_passwd,17

0,tcp,smtp,SF,0,83,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.0
0,110,25,0.23,0.13,0.01,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,http,SF,282,4532,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,96,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,normal,21
0,udp,domain_u,SF,44,113,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,5,0.00,0.00,0.00,0.00,1.00,
0.00,0.40,255,253,0.99,0.01,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,SH,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.
00,30,1,0.03,1.00,1.00,0.00,1.00,1.00,0.00,0.00,nmap,18
0,tcp,uucp,RSTO,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,119,7,0.00,0.00,1.00,1.00,0.06,0.0
7,0.00,255,7,0.03,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,20
0,tcp,other,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,497,1,0.00,1.00,0.99,0.00,0.00,1.00,0.
00,255,1,0.00,1.00,0.00,0.00,0.01,1.00,0.99,0.00,satan,20
0,tcp,ftp,SF,26,157,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.
00,155,71,0.46,0.03,0.01,0.00,0.00,0.00,0.00,0.00,guess_passwd,7
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,217,6,0.00,0.00,1.00,1.00,0.03,0.0
7,0.00,255,6,0.02,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,106,15,0.00,0.00,1.00,1.00,0.14,0.
08,0.00,255,15,0.06,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
1,tcp,smtp,SF,995,330,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00
,0.00,62,42,0.66,0.05,0.02,0.05,0.00,0.00,0.00,0.00,normal,21
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,239,12,0.00,0.00,1.00,1.00,0.05,0.
07,0.00,255,12,0.05,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
31,tcp,telnet,SF,197,1608,0,0,0,1,0,1,1,0,0,1,2,1,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0
.00,0.00,248,32,0.13,0.03,0.00,0.00,0.00,0.00,0.00,0.00,ps,0

0,tcp,http,SF,197,1830,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,26,229,1.00,0.00,0.04,0.05,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,117,10,0.00,0.00,1.00,1.00,0.09,0.

06,0.00,255,10,0.04,0.05,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

0,tcp,http,SF,230,843,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,25,25,0.00,0.00,0.00,0.00,1.00,0.

00,0.00,25,255,1.00,0.00,0.04,0.06,0.00,0.00,0.00,0.00,normal,21

0,tcp,http,SF,161,14086,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.

00,0.00,151,151,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,228,15,0.00,0.00,1.00,1.00,0.07,0.

07,0.00,255,15,0.06,0.08,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21

8080,tcp,telnet,SF,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,255,69,0.27,0.03,0.00,0.00,0.00,0.01,0.11,0.19,processtable,8

8124,tcp,telnet,SF,0,15,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.0

0,0.00,255,60,0.24,0.04,0.00,0.00,0.00,0.02,0.11,0.22,processtable,8

0,tcp,http,SF,224,278,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,5,0.00,0.00,0.00,0.00,1.00,0.00,

0.40,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,259,20,1.00,1.00,0.00,0.00,0.08,0.0

7,0.00,255,20,0.08,0.07,0.00,0.00,1.00,1.00,0.00,0.00,neptune,21

0,tcp,http,SF,321,372,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,16,17,0.00,0.00,0.00,0.00,1.00,0.

00,0.12,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21

0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,173,173,0.00,0.00,0.00,0.00,1.0

0,0.00,0.00,255,254,1.00,0.01,1.00,0.00,0.00,0.00,0.00,0.00,smurf,21

0,tcp,whois,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,282,17,0.00,0.00,1.00,1.00,0.06,0.0

6,0.00,255,17,0.07,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,20

0,udp,private,SF,48,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,5,0.00,0.00,0.00,0.00,1.00,0.00,
0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,snmpguess,15
0,tcp,http,SF,303,289,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,6,11,0.00,0.00,0.00,0.00,1.00,0.0
0,0.36,30,255,1.00,0.00,0.03,0.01,0.00,0.00,0.00,0.00,normal,21
0,icmp,ecr_i,SF,508,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.00,0.00,0.00,0.00,1.00,
0.00,0.00,255,192,0.75,0.02,0.75,0.00,0.00,0.00,0.23,0.00,smurf,18
0,tcp,private,SH,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.
00,80,1,0.01,1.00,1.00,0.00,1.00,1.00,0.00,0.00,nmap,19
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,132,12,0.00,0.00,1.00,1.00,0.09,0.
05,0.00,255,12,0.05,0.06,0.00,0.00,0.00,0.00,1.00,1.00,neptune,21
0,tcp,finger,SF,8,447,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.00,
1.00,74,10,0.04,0.31,0.01,0.20,0.01,0.00,0.00,0.00,normal,21
0,udp,domain_u,SF,43,78,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,208,207,0.00,0.00,0.00,0.00,1
.00,0.01,0.00,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,normal,18
0,tcp,telnet,SF,124,174,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.0
0,0.00,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.02,0.02,guess_passwd,16
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,226,2,0.03,0.00,0.97,1.00,0.01,0.0
7,0.00,255,2,0.01,0.08,0.00,0.00,0.02,0.00,0.98,1.00,neptune,21
0,tcp,http,SF,257,633,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3,4,0.00,0.00,0.00,0.00,1.00,0.00,
0.50,171,255,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,normal,21
0,udp,domain_u,SF,42,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7,12,0.00,0.00,0.00,0.00,1.00,
0.00,0.25,220,220,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal,21
0,udp,domain_u,SF,42,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,19,0.00,0.00,0.00,0.00,1.0
0,0.00,0.11,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal,21

Figure 2: NSL-KDD test dataset
Source: Github (2020)

APPENDIX III

FINAL RESULTS

SN	PC1	PC2	PC3	PC4	PC5	PC6	PC7
5188	-1.43341	0.596405	-0.44467	-1.14996	2.509284	1.120042	-1.56106
110012	1.208092	0.588724	0.065598	-2.68224	3.190255	1.539098	-2.00343
289664	4.76409	-4.17956	-0.4102	0.516971	0.221604	0.315511	-0.40474
11733	-3.80554	0.220176	-6.16195	0.148521	-1.61079	0.749816	-0.13229
112761	4.891938	-4.20786	-0.46565	0.473168	0.239561	0.282929	-0.35332
91777	-1.45234	-0.159	1.378551	-0.01186	0.491553	-0.4992	0.621893
241358	-2.60999	0.093091	0.456637	1.193181	0.238186	-1.14077	0.08406
18724	-1.99626	-0.54995	-6.02778	-4.54614	0.817659	0.947084	1.188159
37521	-0.59901	0.710094	0.353775	-0.04459	-0.22459	0.124985	0.222975

PC8	PC9	PC10	PC11	PC12	PC13	PC14	PC15
0.849499	0.064919	-0.36444	0.981542	-0.09704	0.566431	-1.09105	-0.30802
1.376987	0.27099	-1.39076	0.896866	-0.73486	0.056956	0.346718	0.048024
0.2378	0.55475	-0.44988	-0.40682	-0.66058	0.008154	0.413063	0.305829
-0.60043	0.506227	-1.29716	0.044736	0.301274	-0.06884	-0.01851	-0.0492
0.18615	0.370871	-0.41412	-0.39389	-0.65759	0.039806	0.403971	0.2998
-0.73715	0.197757	-1.81841	-1.07062	-2.35901	1.654002	-0.73425	-0.37695
1.828881	-0.35664	1.213269	0.734716	-0.20698	0.142829	-0.01658	0.121971
-4.17533	-3.16717	8.379447	-5.16169	-2.3807	0.301356	0.105053	0.057338
-0.80542	0.485391	-2.60376	-0.95341	-2.17076	1.730463	-0.94782	-0.43489

PC16	PC17	PC18	PC19	PC20	PC21	PC22	PC23
-1.8841	-0.10106	-0.21132	0.521873	-1.12022	0.028723	-0.14176	-0.23805
0.775836	0.451745	-0.62111	-0.14419	-0.1341	0.058582	0.01413	0.108368
0.972755	0.277142	-0.23485	-0.17009	-0.07654	0.10105	-0.00301	-0.04474
-0.14783	-0.21819	-0.02033	0.109183	0.029106	0.036225	0.056989	0.027328
0.940998	0.250505	-0.23763	-0.15985	-0.08607	0.11368	0.012623	-0.04428
-1.69693	-0.06101	-0.68544	-0.83662	4.697652	-1.31443	-0.97666	-0.29589
0.250291	-0.09655	0.033966	-0.03281	0.147765	-0.03313	0.142869	0.183885
0.442796	0.67874	0.099866	-0.19613	0.368079	-0.18241	-0.07104	-0.03548
-1.98836	-0.10621	-0.77626	-0.86778	5.51174	-1.51989	-1.06084	-0.31204

PC24	PC25	PC26	PC27	PC28	PC29	PC30	PC31
-0.04097	-0.24601	0.274055	-0.02524	-0.31038	-0.5044	0.464046	0.248291
-0.06636	0.006556	0.029092	0.01416	-0.03842	-0.03363	0.093504	-0.02029
-0.08857	-0.10364	0.024803	-0.00698	-0.03687	-0.07626	0.071422	0.110993
-0.08436	-0.03596	0.087156	0.024951	-0.01998	-0.07275	0.032027	0.064439
-0.12272	-0.13227	0.02922	-0.01486	-0.03268	-0.0861	0.095472	0.118817
0.162373	0.637566	0.206886	-0.00399	0.119879	0.321275	-0.32297	0.428508
-0.06653	0.171206	0.108496	0.065695	-0.08492	-0.01409	0.028414	-0.06793
-0.16956	-0.07961	0.190489	-0.01736	-0.32991	-0.43887	0.237063	0.279467
0.106846	0.716157	0.290157	-0.0125	0.08961	0.370457	-0.38355	0.43724

PC32	PC33	PC34	PC35	PC36	PC37	PC38	PC39
0.274289	0.168795	0.196434	0.180756	0.124714	-0.03128	0.276966	-0.21958
0.013292	-0.03607	-0.02418	-0.01627	-0.01714	0.014987	0.003005	0.017857
0.108795	-0.02033	0.052084	0.02941	-0.0094	0.009499	0.022714	-0.03871
0.056664	-0.03152	0.056972	0.015423	0.008694	0.02103	0.046384	-0.00665
0.132343	-0.02636	0.055578	0.021791	-0.00643	0.008927	0.028292	-0.04374
-0.64849	-0.09869	-0.27961	-0.02805	0.31	0.020309	-0.23659	0.029365
-0.19687	-0.09783	-0.16139	-0.04988	-0.05709	0.059904	-0.14054	0.056844
0.299154	-0.02813	0.127981	0.308976	0.054738	-0.03959	0.247038	-0.0705
-0.65779	-0.14986	-0.31848	-0.0607	0.353178	0.040466	-0.20971	0.032624

PC40	PC41	PC42	PC43	PC44	PC45	PC46	PC47
0.164281	0.271315	0.298299	-0.21565	0.161457	-0.11932	0.092712	0.019399
0.001184	0.01087	0.000783	0.083221	-0.00975	-0.01198	0.015618	0.004404
-0.01296	-0.0519	0.014324	0.176439	0.007044	-0.01006	-0.0017	-0.0012
0.026593	0.072205	0.069764	0.052549	-0.01969	0.025628	0.005628	-0.00473
-0.00978	-0.03571	0.001406	0.18288	-0.00516	-0.01508	0.000759	-0.00132
-0.26532	0.170807	-0.025	0.047128	0.05009	-0.11704	-0.08584	0.006675
-0.05366	0.010414	-0.0402	0.038305	-0.02805	-0.02097	0.036438	0.009034
0.01809	0.126591	0.353941	-0.34637	0.132045	0.196084	-0.2348	-0.13636
-0.28506	0.232716	0.003749	0.022029	0.034817	-0.11401	-0.09436	0.010222

PC48	PC49	PC50	PC51	PC52	PC53	PC54	PC55
0.01545	0.013883	-0.01929	-0.09379	0.033215	0.048525	-0.0909	-0.05882
0.00741	0.002983	-0.0018	0.006215	0.014866	-0.01186	-0.01966	-0.01573
0.006924	-0.00822	-0.00056	-0.01273	-0.00421	0.021861	-0.00565	0.002936
0.003232	0.00501	-0.00183	-0.00391	0.025313	-0.01366	-0.01784	0.004542
0.007851	-0.00818	-0.00074	-0.0133	0.005196	0.025573	-0.00801	0.004914
0.007316	-0.05436	-0.0034	-0.00966	-0.0073	-0.24796	0.175255	-0.0329
-0.00609	0.001456	0.00142	0.034947	-0.00052	-0.02172	-0.0286	-0.00334
-0.00628	0.013532	0.032552	-0.03622	-0.14356	-0.03896	-0.00622	-0.03715
0.001772	-0.04338	-0.00754	-0.00837	0.043197	-0.31472	0.191177	-0.02769

PC56	PC57	PC58	PC59	PC60	PC61	PC62	PC63
-0.10556	-0.20741	-0.00829	-0.19157	-0.03852	-0.02104	-0.05698	-0.06674
-0.01319	0.002367	-0.01418	0.000238	-0.0067	-0.00202	-0.0009	0.010973
0.046428	0.049624	-0.01655	0.013147	-0.00464	-0.01317	-0.03436	0.027717
-0.00928	-0.00951	-0.01931	-0.02858	-0.01658	0.002502	0.014562	0.004828
0.039688	0.045607	-0.02449	0.011386	-0.00897	-0.00885	-0.02353	0.03172
-0.18207	0.008792	-0.00169	0.052061	0.03718	-0.05403	0.061843	0.022992
-0.02392	0.020733	0.003774	0.004646	0.008505	0.010498	0.0128	0.020946
0.006501	-0.00617	0.034188	-0.00812	-0.00269	-0.03655	-0.13021	-0.02872
-0.20914	-0.00162	0.003402	0.026731	0.060524	-0.05699	0.105273	0.012538

PC64	PC65	PC66	PC67	PC68	PC69	PC70	PC71
0.026465	-0.03984	0.131631	-0.431	-0.09089	0.145141	-0.04534	-0.02046
0.003148	0.039914	0.005963	0.04663	0.017481	-0.04266	0.025823	-0.00114
0.037506	0.086355	-0.02088	0.115364	0.046035	-0.08443	0.030532	0.0008
0.017342	0.070286	0.009329	-0.01818	0.005931	0.017778	0.007867	-0.01791
0.038085	0.093789	-0.0201	0.118175	0.050006	-0.08391	0.034083	0.001436
-0.02295	0.134086	-0.19176	0.102926	0.074559	-0.05577	-0.00405	0.007944
-0.00428	0.022551	-0.01414	0.021962	0.00581	0.013413	0.00299	-0.01391
0.019974	-0.003	-0.0029	-0.01204	-0.0191	-0.06942	-0.00806	-0.01469
-0.0362	0.114101	-0.18693	0.076515	0.073868	-0.01479	-0.01284	0.00777

PC72	PC73	PC74	PC75	PC76	PC77	PC78	PC79
-0.05586	-0.00547	0.232443	-0.01008	0.022562	-0.32387	-0.23442	1.0158
0.009557	-0.00674	-0.05749	0.000146	0.004887	0.102742	0.022052	-0.25433
0.000616	-0.00571	-0.11614	-0.00223	0.01973	0.178398	0.029836	-0.51407
-0.00064	0.016878	0.00158	0.000748	-0.01906	-0.02431	0.004828	-0.01447
0.002349	-0.00636	-0.11437	0.000229	0.015693	0.172527	0.034322	-0.52383
0.012066	0.071227	-0.00677	-0.00635	-0.05522	-0.09755	0.019701	-0.12645
0.005417	0.013893	-0.00031	-0.00818	-0.02002	-0.03588	0.014368	0.013182
-0.00198	-0.01928	-0.05049	-0.00889	0.037035	0.107675	-0.02587	-0.10588
0.007949	0.099113	0.030275	-0.01287	-0.07317	-0.18672	0.016234	0.001142

PC80	PC81	PC82	PC83	PC84	PC85
0.187921	0.042053	-0.24301	-0.06745	0.331163	-0.51075
-0.10798	-0.05209	-0.6451	0.110284	0.252581	-0.59601
-0.15209	-0.05794	-0.81644	0.267012	0.230452	-1.01476
-0.0347	-0.0068	-0.04942	-0.01854	0.010824	-0.13111
-0.15292	-0.06084	-0.81933	0.265786	0.247251	-1.06038
0.03126	-0.01655	-0.05213	-0.04446	-0.04408	0.168358
0.000854	-0.00846	-0.05918	0.0466	0.036568	-0.05721
-0.05275	-0.03255	-0.58336	0.127527	0.312746	-0.14551
0.066488	0.004521	0.308484	-0.03548	-0.19775	0.293882

PC86	PC87	PC88	class	Predicted
0.21299	-0.04361	1.33562	normal	normal
PC86	PC87	PC88	class	Predicted
0.21299	-0.04361	1.33562	normal	normal
PC86	PC87	PC88	class	Predicted
0.21299	-0.04361	1.33562	normal	normal
PC86	PC87	PC88	class	Predicted
0.21299	-0.04361	1.33562	normal	normal
PC86	PC87	PC88	class	Predicted
0.21299	-0.04361	1.33562	normal	normal

			1.34416		0.44784	1.59974	6.40775
93402	-2.00847	-0.32392	7	-0.87271	8	9	2
		0.05118			3.63668	1.01874	
313978	-0.38632	9	-0.43239	-3.18599	1	6	-1.24182
			2.34759	1.74233	1.10589	1.54867	
103367	-1.24601	-0.03116	3	9	1	8	-1.07418
	4.91758			0.51337	0.21585	0.32708	
165945	3	-4.20257	-0.43597	3	8	4	-0.41618
		0.33763				1.12225	
1011	-3.76339	4	-6.36657	0.88681	-1.86574	5	-0.72474
		0.22910		0.94729	0.16912		0.05378
124722	-2.23267	3	0.52391	5	3	-1.04238	1
			0.39503	1.05952	0.26900		0.22119
323856	-2.49873	0.16529	6	3	6	-1.2546	4
	5.28379	3.64881					1.47950
80180	9	8	-0.69371	-0.54698	-1.47147	-1.30429	8
	4.96138			0.54299	0.20326		
213785	1	-4.19194	-0.42442	1	6	0.31619	-0.42019
		0.44740		0.24380		1.16001	
58917	-2.39428	1	-4.31487	6	-1.26502	7	-0.92368

0.15138				0.11031			
3	0.35398	-1.47465	-1.28346	1	-0.13034	-1.33129	-3.19277
0.58811	0.63857		0.33377			0.52049	0.25820
9	4	-0.3355	9	-0.69904	-0.02105	5	4
				0.49002	0.24465		0.09969
-1.5806	-0.78028	-0.34204	-1.09888	3	7	-0.01601	2
0.24255	0.40820				0.04399	0.39714	0.29800
6	3	-0.43496	-0.37666	-0.6552	1	6	4
	0.36916			0.40814			0.00139
-0.09987	3	-1.36886	0.56621	8	-0.03373	-0.01574	2
1.84116		0.95665	0.57448		0.05455		0.02316
9	-0.46042	7	9	-0.28286	1	-0.07059	5
1.78802		1.02889	0.76549		0.15863	0.01797	
7	-0.66519	7	7	-0.27148	4	6	0.06354
			1.25965	3.03521		0.34088	0.17936
-0.43894	-1.67514	0.87093	2	7	-0.34689	8	2
0.27578					0.05509	0.39358	0.28839
9	0.36633	-0.50455	-0.33665	-0.6536	8	3	2
0.21454							
1	-0.02462	-0.43128	-0.00513	-0.00431	-0.04612	-0.20231	-0.19726

0.916946	0.153092	-0.23475	-0.98576	-0.42282	0.263631	0.060319	-1.11357
1.033567	0.145004	-0.17851	-0.11793	-0.09077	0.080737	0.097974	0.082656
0.035288	0.045358	0.085788	-0.18157	-0.14968	0.037235	-0.42814	-0.60339
0.943952	0.26282	-0.25309	-0.16593	-0.07795	0.106687	0.005631	-0.04429
-0.03282	-0.21975	0.075456	0.018066	0.064638	0.028902	0.061766	-0.00929
0.119043	0.012262	-0.05259	0.000687	0.184746	-0.03645	0.141669	0.200046
0.176903	-0.04569	-0.00608	-0.02805	0.049776	0.002043	0.160566	0.211681
0.233459	0.093463	-2.2807	0.008045	1.28981	-0.99265	0.832372	-0.51714
0.927868	0.268222	-0.27246	-0.16203	-0.09826	0.113433	0.007685	-0.0408
-0.27521	0.146527	-0.47978	0.114341	-0.49198	0.111577	-0.0347	0.029222

-1.25695	-3.06764	-1.05079	-0.32325	-5.40979	0.098571	0.271482	-3.4883
-0.16237	-0.02926	0.063385	-0.00627	-0.04007	-0.05525	0.079661	0.019251
0.272412	-0.4111	-0.55079	-0.32469	0.188487	0.344298	-0.4066	0.020479
-0.11247	-0.12254	0.027896	-0.01294	-0.03326	-0.08116	0.090601	0.114726
-0.10038	-0.08251	0.023081	-0.02282	-0.01289	-0.03774	0.015703	0.044594
-0.06361	0.146737	0.097958	0.072015	-0.07619	-0.04275	0.061436	-0.06552
-0.06683	0.162442	0.104216	0.074473	-0.09043	-0.02655	0.035923	-0.06256
-2.44368	-0.55583	-3.34314	-3.53173	0.512504	-0.45322	-0.80259	0.860898
-0.11549	-0.12668	0.028635	-0.01222	-0.03314	-0.08414	0.097722	0.115323
0.091286	-0.02785	0.038211	0.072612	-0.0034	-0.09908	0.098686	0.000984

		0.13177			0.12129		0.71218
-0.99719	-1.54025	5	-1.16587	-2.30575	5	-0.86996	3
							0.01361
-0.00905	-0.08556	-0.03612	-0.02118	-0.01622	0.01284	-0.01876	2
0.33666	0.20551	0.45789	0.05614	0.18971		0.39878	
7	4	1	6	1	-0.30063	8	-0.16563
0.12521		0.05251	0.02141			0.02579	
9	-0.02374	2	4	-0.00677	0.00961	1	-0.04316
		0.06436	0.01937	0.01148			
0.14307	-0.03166	1	4	5	-0.00382	0.07905	-0.01994
					0.05738		0.06450
-0.13375	-0.06263	-0.12352	-0.03014	-0.0697	8	-0.12623	9
					0.06134		0.06623
-0.18076	-0.11119	-0.15403	-0.03813	-0.06371	7	-0.13239	9
0.83491	0.21330	1.86817		4.07285	2.16563		0.67641
7	4	9	-3.05863	2	2	-2.45326	4
0.12782		0.05312	0.01999		0.01064		
1	-0.02457	9	2	-0.00717	9	0.0265	-0.04342
0.00645		0.07862	0.01568		0.03312		
7	0.10611	8	3	-0.01605	1	0.01752	-0.01517

					0.13726		0.05408
-0.14682	-0.21911	-0.1208	-0.12267	-0.19848	1	-0.00879	7
	0.00539		0.09101			0.01327	
-0.01809	8	-0.03082	4	-0.03039	-0.00612	5	-0.01447
0.10158		0.03245		0.12356			
4	-0.13109	8	-0.00696	2	-0.02103	-0.08404	-0.01186
		0.00380	0.17691				
-0.00977	-0.03749	9	3	-0.0023	-0.01445	8.58E-05	-0.00055
0.02474	0.04753	0.04981			0.02398	0.00368	
5	1	9	0.03394	-0.02055	7	9	-0.00475
	0.00401		0.04368			0.04865	0.01565
-0.03331	8	-0.01919	5	-0.01523	-0.02473	7	4
	0.00451		0.09056			0.03518	0.00321
-0.05573	5	-0.03169	1	-0.03491	-0.01244	8	8
		2.28679					0.10681
-0.98313	-4.11546	3	-0.79452	0.46845	-0.90705	-1.06202	8
		0.00254	0.18078			0.00105	
-0.00888	-0.03348	6	2	-0.00472	-0.0156	4	-0.00058
0.08299	0.04815	0.05980		0.02588	0.00499		0.02807
3	6	5	-0.0358	4	3	0.01582	7

	0.03765		0.38166	0.15262			
-0.03533	7	-0.00362	1	1	-0.0248	-0.20366	-0.1477
0.00679			0.00289	0.01357			
4	-0.00724	-0.00034	9	2	-0.00729	-0.02671	-0.0047
				0.03089	0.00547	0.07873	0.00242
-0.00047	-0.00015	0.00199	-0.10171	7	5	6	3
0.00761				0.00391	0.02481		0.00398
3	-0.00758	-0.00072	-0.01263	8	6	-0.00709	2
	0.00892			0.02169	0.01059		0.01702
-0.00099	4	-0.00132	-0.00443	3	4	-0.01045	9
	0.00451	0.00225	0.03590				
-0.00861	9	8	4	-0.01284	-0.01069	-0.03319	-0.02008
		0.00213	0.03297				0.00098
-0.00514	-0.00048	1	1	-0.00513	-0.02122	-0.03591	3
		0.07498			0.51488	0.06260	0.18431
0.06151	-0.00977	3	-0.24582	-0.51039	5	6	7
0.00800				0.00579	0.02534		0.00430
3	-0.00769	-0.00078	-0.01277	2	6	-0.0083	2
			0.00287	0.00590	0.01595		
-0.00075	0.0057	-0.00593	7	8	7	-0.01358	-0.03957

					0.17302	0.18301	
-0.01733	-0.06009	-0.02573	-0.15096	-0.13476	6	7	0.18029
	0.02521					0.00231	0.03429
0.0429	7	-0.03159	-0.00113	-0.01325	-0.00474	3	5
0.11705				0.01546			
9	-0.07748	-0.01535	-0.00569	1	-0.0455	-0.00998	-0.03547
0.03878	0.04599						
4	2	-0.02185	0.01106	-0.0075	-0.01009	-0.02555	0.02961
					0.00429	0.01198	
-0.00907	-0.00978	-0.01027	-0.02201	-0.01259	6	8	-0.00485
	0.02233	0.01163	0.01666	0.00611	0.00950	0.00214	0.01166
-0.06247	1	7	9	8	4	5	9
	0.01721		0.00808		0.01539	0.00974	0.02790
-0.01544	7	-0.00671	3	-0.00452	2	8	2
0.92669	4.79008	1.38131	2.17472	5.86201		2.71549	4.72027
7	4	7	3	9	-1.84729	7	5
0.03692	0.04354		0.01033				0.03045
3	2	-0.02406	1	-0.00919	-0.00868	-0.02345	8
		0.01377					
-0.0483	-0.03103	9	-0.01511	-0.0112	0.0024	-0.01357	-0.02411

-0.02283	-0.04117	0.32673	0.185074	0.078244	-0.13912	0.129997	
0.021302	0.089144	-0.00487	0.086259	0.030044	-0.03967	0.024926	
0.01522	0.024208	0.017749	-0.02315	0.033763	-0.04966	0.015726	
0.037248	0.088557	-0.01921	0.113988	0.047821	-0.08247	0.032677	
0.007953	0.00921	0.014574	-0.01343	-0.00321	0.013434	0.000731	
-0.013	0.000387	-0.01399	0.022538	0.001467	-0.00207	0.006419	
0.001388	0.042752	-0.0187	0.050749	0.012703	-0.00411	0.009233	
-3.13074	-2.81353	-2.9511	-0.48083	-0.15682	-0.50776	2.736617	
0.037634	0.091431	-0.01901	0.114039	0.048765	-0.08268	0.033812	
-0.00469	0.017142	0.027395	-0.11264	-0.02454	0.036674	0.004331	

0.013557	0.065301	-0.01261	0.038651	-0.28163	-0.01561	0.01736	0.388389
-0.01046	0.008455	0.0076	-0.07861	-0.00363	-0.00235	0.093733	0.029018
0.035364	-0.01856	-0.01438	0.011729	0.017141	0.055743	0.116237	-0.03273
0.001496	0.001561	-0.00668	-0.11259	-7.22E-05	0.016875	0.171387	0.031702
-0.00789	0.000474	0.003255	0.001342	0.002986	-0.00921	-0.01358	0.008073
-0.00379	0.00828	-0.00076	-0.00218	-0.00675	-0.00367	0.000275	0.014535
-0.01652	0.012102	0.013505	-0.02175	-0.00887	-0.02367	-0.00996	0.037569
0.304367	2.813478	-0.81805	-4.28553	0.676899	-2.15346	-3.02031	-0.29966
0.001348	0.002249	-0.00681	-0.1121	0.000444	0.015691	0.170377	0.033434
-0.01253	-0.00844	-0.00941	0.037057	0.004995	-0.00193	-0.00671	-0.0523

0.310078	-0.02983	-0.03452	-0.16805	-2.01771	0.07269
-0.35292	-0.11713	-0.03231	-0.58415	0.235833	0.311408
-0.20904	-0.00894	0.072983	0.497691	-0.0434	-0.31022
-0.51221	-0.15051	-0.0592	-0.81376	0.262462	0.239644
-0.02173	-0.01084	0.004634	0.040834	-0.01029	-0.05564
0.021011	-0.00393	-0.02859	-0.14757	-0.00588	0.056306
-0.07926	-0.0244	-0.02309	-0.14755	0.03957	0.054862
2.457507	0.497404	0.028088	1.125286	-0.06559	0.115585
-0.51425	-0.15186	-0.06111	-0.82355	0.258799	0.24499
0.216915	-0.02044	-0.03331	-0.27421	-0.14254	0.038104

-1.15699	-1.31997	-0.3013	0.648507	DOS	DOS
-0.78918	0.318157	0.247363	1.591094	U2R	U2R
0.35957	-0.15293	-0.18622	0.305336	DOS	DOS
-1.04296	0.264488	0.574554	-0.16736	PROBE	PROBE
-0.06109	0.003275	0.002845	-0.08558	DOS	DOS
0.113992	-0.0383	0.031037	-0.11301	normal	normal
-0.15854	0.002336	0.038658	-0.36615	U2R	U2R
1.159934	-0.1756	-0.32748	1.88739	normal	normal
-1.05705	0.266951	0.578012	-0.19794	R2L	R2L
0.17542	0.019401	-0.09956	0.567177	normal	normal

APPENDIX IV

SYSTEM DEVELOPMENT (PYTHON) CODE


```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "3ad25937-bea9-457e-ba87-4b00dd451d35",
   "metadata": {
    "tags": []
   },
   "outputs": [],
   "source": [
    "import numpy as np\n",
    "import pandas as pd\n",
    "import seaborn as sns\n",
    "from scipy.io import arff\n",
    "import matplotlib.pyplot as plt\n",
    "from sklearn.ensemble import VotingClassifier\n",
    "pd.set_option('display.max_columns', 200)"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "728bad7d-fa2e-4859-a5ca-2a47add33261",
```

```
"metadata": {
 "tags": []
},
"source": [
 "# Loading the Dataset"
]
},
{
"cell_type": "code",
"execution_count": 2,
"id": "f5090ad1-739f-4330-945a-6ecc6ebadd05",
"metadata": {
 "tags": []
},
"outputs": [],
"source": [
 "#Load the training dataset\n",
 "data = pd.read_csv(\"KDDTrain+.txt\")\n",
 "data_2, meta = arff.loadarff('KDDTrain+.arff')\n",
 "data_2 = pd.DataFrame(data_2)\n",
 "\n",
 "\n",
 "# Load the Test Dataset\n",
 "df = pd.read_csv(\"KDDTest+.txt\")\n",
 "df_2, meta = arff.loadarff('KDDTest+.arff')\n",
```

```
 "df_2 = pd.DataFrame(df_2)"
]
},
{
"cell_type": "code",
"execution_count": 3,
"id": "7ed94bb1-44fd-47e1-aa33-a6b53ebc46c3",
"metadata": {
 "tags": []
},
"outputs": [
 {
  "data": {
   "text/html": [
    "<div>\n",
    "<style scoped>\n",
    "    .dataframe tbody tr th:only-of-type {\n",
    "        vertical-align: middle;\n",
    "    }\n",
    "\n",
    "    .dataframe tbody tr th {\n",
    "        vertical-align: top;\n",
    "    }\n",
    "\n",
    "    .dataframe thead th {\n",
```

```
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>0</th>\n",
"      <th>tcp</th>\n",
"      <th>ftp_data</th>\n",
"      <th>SF</th>\n",
"      <th>491</th>\n",
"      <th>0.1</th>\n",
"      <th>0.2</th>\n",
"      <th>0.3</th>\n",
"      <th>0.4</th>\n",
"      <th>0.5</th>\n",
"      <th>0.6</th>\n",
"      <th>0.7</th>\n",
"      <th>0.8</th>\n",
"      <th>0.9</th>\n",
"      <th>0.10</th>\n",
"      <th>0.11</th>\n",
"      <th>0.12</th>\n",
"      <th>0.13</th>\n",
```

```
"      <th>0.14</th>\n",
"      <th>0.15</th>\n",
"      <th>0.16</th>\n",
"      <th>0.18</th>\n",
"      <th>2</th>\n",
"      <th>2.1</th>\n",
"      <th>0.00</th>\n",
"      <th>0.00.1</th>\n",
"      <th>0.00.2</th>\n",
"      <th>0.00.3</th>\n",
"      <th>1.00</th>\n",
"      <th>0.00.4</th>\n",
"      <th>0.00.5</th>\n",
"      <th>150</th>\n",
"      <th>25</th>\n",
"      <th>0.17</th>\n",
"      <th>0.03</th>\n",
"      <th>0.17.1</th>\n",
"      <th>0.00.6</th>\n",
"      <th>0.00.7</th>\n",
"      <th>0.00.8</th>\n",
"      <th>0.05</th>\n",
"      <th>0.00.9</th>\n",
"      <th>normal</th>\n",
"      <th>20</th>\n",
```

```
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>0</td>\n",
"      <td>udp</td>\n",
"      <td>other</td>\n",
"      <td>SF</td>\n",
"      <td>146</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
```

```
"     <td>0</td>\n",
"     <td>0</td>\n",
"     <td>13</td>\n",
"     <td>1</td>\n",
"     <td>0.0</td>\n",
"     <td>0.0</td>\n",
"     <td>0.0</td>\n",
"     <td>0.0</td>\n",
"     <td>0.08</td>\n",
"     <td>0.15</td>\n",
"     <td>0.00</td>\n",
"     <td>255</td>\n",
"     <td>1</td>\n",
"     <td>0.00</td>\n",
"     <td>0.60</td>\n",
"     <td>0.88</td>\n",
"     <td>0.00</td>\n",
"     <td>0.00</td>\n",
"     <td>0.00</td>\n",
"     <td>0.00</td>\n",
"     <td>0.00</td>\n",
"     <td>normal</td>\n",
"     <td>15</td>\n",
"   </tr>\n",
"   <tr>\n",
```

```
    "      <th>1</th>\n",
    "      <td>0</td>\n",
    "      <td>tcp</td>\n",
    "      <td>private</td>\n",
    "      <td>S0</td>\n",
    "      <td>0</td>\n",
    "      <td>0</td>\n",
    "      <td>0</td>\n",
"      <td>77</td>\n",
    "      <td>0.30</td>\n",
    "      <td>0.03</td>\n",
    "      <td>0.30</td>\n",
    "      <td>0.00</td>\n",
    "      <td>0.00</td>\n",
    "      <td>0.00</td>\n",
    "      <td>0.00</td>\n",
    "      <td>0.00</td>\n",
    "      <td>normal</td>\n",
    "      <td>21</td>\n",
    "    </tr>\n",
    "  </tbody>\n",
    "</table>\n",
    "<p>125972 rows × 43 columns</p>\n",
    "</div>"
   ],
```

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

```
"125971   0    0    0    0    0    0    0    0    0    1    1   0.0  \n",
"\n",
"       0.00.1  0.00.2  0.00.3  1.00  0.00.4  0.00.5  150   25  0.17  0.03  \\\n",
"0        0.0    0.0    0.0  0.08    0.15    0.00  255    1  0.00  0.60   \n",
"1        1.0    0.0    0.0  0.05    0.07    0.00  255   26  0.10  0.05   \n",
"2        0.2    0.0    0.0  1.00    0.00    0.00   30  255  1.00  0.00   \n",
"3        0.0    0.0    0.0  1.00    0.00    0.09  255  255  1.00  0.00   \n",
"4        0.0    1.0    1.0  0.16    0.06    0.00  255   19  0.07  0.07   \n",
"...      ...    ...    ...   ...     ...     ...  ...  ...   ...   ...  \n",
"125967    1.0    0.0    0.0  0.14    0.06    0.00  255   25  0.10  0.06   \n",
"125968    0.0    0.0    0.0  1.00    0.00    0.00  255  244  0.96  0.01   \n",
"125969    0.0    0.0    0.0  1.00    0.00    0.00  255   30  0.12  0.06   \n",
"125970    1.0    0.0    0.0  0.06    0.05    0.00  255    8  0.03  0.05   \n",
"125971    0.0    0.0    0.0  1.00    0.00    0.00  255   77  0.30  0.03   \n",
"\n",
"       0.17.1  0.00.6  0.00.7  0.00.8  0.05  0.00.9   normal  20  \n",
"0        0.88    0.00    0.00    0.00  0.00    0.00   normal  15  \n",
"1        0.00    0.00    1.00    1.00  0.00    0.00  neptune  19  \n",
"2        0.03    0.04    0.03    0.01  0.00    0.01   normal  21  \n",
"3        0.00    0.00    0.00    0.00  0.00    0.00   normal  21  \n",
"4        0.00    0.00    0.00    0.00  1.00    1.00  neptune  21  \n",
"...      ...     ...     ...     ...   ...     ...      ... ..  \n",
"125967   0.00    0.00    1.00    1.00  0.00    0.00  neptune  20  \n",
"125968   0.01    0.00    0.00    0.00  0.00    0.00   normal  21  \n",
"125969   0.00    0.00    0.72    0.00  0.01    0.00   normal  18  \n",
```

```
    "125970    0.00    0.00    1.00    1.00  0.00    0.00  neptune  20  \n",
    "125971    0.30    0.00    0.00    0.00  0.00    0.00   normal  21  \n",
    "\n",
    "[125972 rows x 43 columns]"
   ]
  },
  "execution_count": 3,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "data"
]
},
{
"cell_type": "code",
"execution_count": 4,
"id": "8942e3f7-cee7-457e-8e55-42cee306b721",
"metadata": {
 "tags": []
},
"outputs": [
 {
  "data": {
```

```
"text/html": [
 "<div>\n",
 "<style scoped>\n",
 "    .dataframe tbody tr th:only-of-type {\n",
 "        vertical-align: middle;\n",
 "    }\n",
 "\n",
 "    .dataframe tbody tr th {\n",
 "        vertical-align: top;\n",
 "    }\n",
 "\n",
 "    .dataframe thead th {\n",
 "        text-align: right;\n",
 "    }\n",
 "</style>\n",
 "<table border=\"1\" class=\"dataframe\">\n",
 "  <thead>\n",
 "    <tr style=\"text-align: right;\">\n",
 "      <th></th>\n",
 "      <th>0</th>\n",
 "      <th>tcp</th>\n",
 "      <th>private</th>\n",
 "      <th>REJ</th>\n",
 "      <th>0.1</th>\n",
 "      <th>0.2</th>\n",
```

```
"      <th>0.3</th>\n",
"      <th>0.4</th>\n",
"      <th>0.5</th>\n",
"      <th>0.6</th>\n",
"      <th>0.7</th>\n",
"      <th>0.8</th>\n",
"      <th>0.9</th>\n",
"      <th>0.10</th>\n",
"      <th>0.11</th>\n",
"      <th>0.12</th>\n",
"      <th>0.13</th>\n",
"      <th>0.14</th>\n",
"    <td>0.00</td>\n",
"      <td>0.00</td>\n",
"      <td>0.00</td>\n",
"      <td>0.0</td>\n",
"      <td>0.44</td>\n",
"      <td>1.00</td>\n",
"      <td>mscan</td>\n",
"      <td>14</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>22543 rows × 43 columns</p>\n",
"</div>"
```

```
],
"text/plain": [
"      0  tcp  private  REJ   0.1   0.2  0.3  0.4  0.5  0.6  0.7  0.8  \\\n",
"0     0  tcp  private  REJ    0     0   0   0   0   0   0   0  \n",
"1     2  tcp  ftp_data  SF  12983    0   0   0   0   0   0   0  \n",
"2     0  icmp  eco_i  SF   20     0   0   0   0   0   0   0  \n",
"3     1  tcp  telnet  RSTO   0    15   0   0   0   0   0   0  \n",
"4     0  tcp   http  SF   267  14515   0   0   0   0   0   1  \n",
"...  ..  ...     ...  ...  ...  ... ... ... ... ... ... ...  \n",
"22538  0  tcp   smtp  SF   794   333   0   0   0   0   0   1  \n",
"22539  0  tcp   http  SF   317   938   0   0   0   0   0   1  \n",
"22540  0  tcp   http  SF  54540  8314   0   0   0   2   0   1  \n",
"22541  0  udp  domain_u  SF   42    42   0   0   0   0   0   0  \n",
"22542  0  tcp  sunrpc  REJ    0     0   0   0   0   0   0   0  \n",
"\n",
"      0.9  0.10  0.11  0.12  0.13  0.14  0.15  0.16  0.17  0.18  229  10  \\\n",
"0       0    0    0    0    0    0    0    0    0    0  136   1  \n",
"1       0    0    0    0    0    0    0    0    0    0   1   1  \n",
"2       0    0    0    0    0    0    0    0    0    0   1  65  \n",
"3       0    0    0    0    0    0    0    0    0    0   1   8  \n",
"4       0    0    0    0    0    0    0    0    0    0   4   4  \n",
"...    ...   ...  ...  ...  ...  ...  ...  ...  ...  ...  ... ..  \n",
"22538   0    0    0    0    0    0    0    0    0    0   1   1  \n",
"22539   0    0    0    0    0    0    0    0    0    0   2  11  \n",
"22540   1    0    0    0    0    0    0    0    0    0   5  10  \n",
```

```
"22541   0    0    0    0    0    0    0    0    0    0   4   6  \n",
"22542   0    0    0    0    0    0    0    0    0    0   4  10  \n",
"\n",
"      0.00  0.00.1  1.00  1.00.1  0.04  0.06  0.00.2  255  10.1  0.04.1  \\\n",
"0      0.0   0.00   1.0    1.0  0.01  0.06   0.00  255    1   0.00  \n",
"1      0.0   0.00   0.0    0.0  1.00  0.00   0.00  134   86   0.61  \n",
"2      0.0   0.00   0.0    0.0  1.00  0.00   1.00   3   57   1.00  \n",
"3      0.0   0.12   1.0    0.5  1.00  0.00   0.75   29   86   0.31  \n",
"4      0.0   0.00   0.0    0.0  1.00  0.00   0.00  155  255   1.00  \n",
"...    ...   ...   ...   ...   ...   ...   ...   ...  ...   ...   ...  \n",
"22538  0.0   0.00   0.0    0.0  1.00  0.00   0.00  100  141   0.72  \n",
"22539  0.0   0.00   0.0    0.0  1.00  0.00   0.18  197  255   1.00  \n",
"22540  0.0   0.00   0.0    0.0  1.00  0.00   0.20  255  255   1.00  \n",
"22541  0.0   0.00   0.0    0.0  1.00  0.00   0.33  255  252   0.99  \n",
"22542  0.0   0.00   1.0    1.0  0.25  1.00   1.00  255   21   0.08  \n",
"\n",
"      0.06.1  0.00.3  0.00.4  0.00.5  0.00.6  1.00.2  1.00.3  neptune  21  \n",
"0       0.06    0.00    0.00    0.00     0.0    1.00    1.00  neptune  21  \n",
"1       0.04    0.61    0.02    0.00     0.0    0.00    0.00   normal  21  \n",
"2       0.00    1.00    0.28    0.00     0.0    0.00    0.00    saint  15  \n",
"3       0.17    0.03    0.02    0.00     0.0    0.83    0.71    mscan  11  \n",
"4       0.00    0.01    0.03    0.01     0.0    0.00    0.00   normal  21  \n",
"...      ...     ...     ...     ...     ...     ...     ...     ... ..  \n",
"22538   0.06    0.01    0.01    0.01     0.0    0.00    0.00   normal  21  \n",
"22539   0.00    0.01    0.01    0.01     0.0    0.00    0.00   normal  21  \n",
```

```
    "22540    0.00    0.00    0.00    0.00    0.0    0.07    0.07    back  15  \n",
    "22541    0.01    0.00    0.00    0.00    0.0    0.00    0.00   normal  21  \n",
    "22542    0.03    0.00    0.00    0.00    0.0    0.44    1.00    mscan  14  \n",
    "\n",
    "[22543 rows x 43 columns]"
   ]
  },
  "execution_count": 4,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "df"
]
},
{
"cell_type": "code",
"execution_count": 5,
"id": "c2df8e8f-2d2e-4420-9958-e98e1b8d7965",
"metadata": {
 "tags": []
},
"outputs": [
 {
```

```
"data": {
 "text/html": [
  "<div>\n",
  "<style scoped>\n",
  "    .dataframe tbody tr th:only-of-type {\n",
  "        vertical-align: middle;\n",
  "    }\n",
  "\n",
  "    .dataframe tbody tr th {\n",
  "        vertical-align: top;\n",
  "    }\n",
  "\n",
  "    .dataframe thead th {\n",
  "        text-align: right;\n",
  "    }\n",
  "</style>\n",
  "<table border=\"1\" class=\"dataframe\">\n",
  "  <thead>\n",
  "    <tr style=\"text-align: right;\">\n",
  "      <th></th>\n",
  "      <th>duration</th>\n",
  "      <th>protocol_type</th>\n",
  "      <th>service</th>\n",
  "      <th>flag</th>\n",
  "      <th>src_bytes</th>\n",
```

```
"      <th>dst_bytes</th>\n",
"      <th>land</th>\n",
"      <th>wrong_fragment</th>\n",
"      <th>urgent</th>\n",
"      <th>hot</th>\n",
"      <th>num_failed_logins</th>\n",
"    <td>0.30</td>\n",
"      <td>0.03</td>\n",
"      <td>0.30</td>\n",
"      <td>0.00</td>\n",
"      <td>0.00</td>\n",
"      <td>0.00</td>\n",
"      <td>0.00</td>\n",
"      <td>0.00</td>\n",
"      <td>b'normal'</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>125973 rows × 42 columns</p>\n",
"</div>"
],
"text/plain": [
"       duration protocol_type      service   flag  src_bytes  dst_bytes  \\\n",
"0           0.0        b'tcp'  b'ftp_data'  b'SF'      491.0        0.0   \n",
"1           0.0        b'udp'     b'other'  b'SF'      146.0        0.0   \n",
```

```
"2          0.0       b'tcp'   b'private'  b'S0'       0.0       0.0   \n",
"3          0.0       b'tcp'     b'http'  b'SF'     232.0     8153.0   \n",
"4          0.0       b'tcp'     b'http'  b'SF'     199.0      420.0   \n",
"...        ...         ...         ...    ...       ...        ...   \n",
"125968     0.0       b'tcp'   b'private'  b'S0'       0.0       0.0   \n",
"125969     8.0       b'udp'   b'private'  b'SF'     105.0     145.0   \n",
"125970     0.0       b'tcp'     b'smtp'  b'SF'    2231.0     384.0   \n",
"125971     0.0       b'tcp'    b'klogin'  b'S0'       0.0       0.0   \n",
"125972     0.0       b'tcp'  b'ftp_data'  b'SF'     151.0       0.0   \n",
"\n",
"       land  wrong_fragment  urgent  hot  num_failed_logins logged_in  \\\n",
"0      b'0'           0.0     0.0  0.0              0.0      b'0'   \n",
"1      b'0'           0.0     0.0  0.0              0.0      b'0'   \n",
"2      b'0'           0.0     0.0  0.0              0.0      b'0'   \n",
"3      b'0'           0.0     0.0  0.0              0.0      b'1'   \n",
"4      b'0'           0.0     0.0  0.0              0.0      b'1'   \n",
"...     ...           ...     ...  ...              ...       ...   \n",
"125968  b'0'          0.0     0.0  0.0              0.0      b'0'   \n",
"125969  b'0'          0.0     0.0  0.0              0.0      b'0'   \n",
"125970  b'0'          0.0     0.0  0.0              0.0      b'1'   \n",
"125971  b'0'          0.0     0.0  0.0              0.0      b'0'   \n",
"125972  b'0'          0.0     0.0  0.0              0.0      b'1'   \n",
"\n",
"       num_compromised  root_shell  su_attempted  num_root  \\\n",
"0              0.0         0.0          0.0       0.0   \n",
```

```
    "1              0.0       0.0       0.0      0.0  \n",
    "2              0.0       0.0       0.0      0.0  \n",
    "3              0.0       0.0       0.0      0.0  \n",
    "4              0.0       0.0       0.0      0.0  \n",
    "...            ...       ...       ...      ...  \n",
    "125968         0.0       0.0       0.0      0.0  \n",
    "125969         0.0       0.0       0.0      0.0  \n",
    "125970         0.0       0.0       0.0      0.0  \n",
    "125971         0.0       0.0       0.0      0.0  \n",
    "125972         0.0       0.0       0.0      0.0  \n",
    "\n",
    "        num_file_creations  num_shells  num_access_files  num_outbound_cmds
\\\n",
    "0               0.0       0.0           0.0              0.0  \n",
    "1               0.0       0.0           0.0              0.0  \n",
    "2               0.0       0.0           0.0              0.0  \n",
    "3               0.0       0.0           0.0              0.0  \n",
    "4               0.0       0.0           0.0              0.0  \n",
    "...             ...       ...           ...              ...  \n",
    "125968          0.0       0.0           0.0              0.0  \n",
    "125969          0.0       0.0           0.0              0.0  \n",
    "125970          0.0       0.0           0.0              0.0  \n",
    "125971          0.0       0.0           0.0              0.0  \n",
    "125972          0.0       0.0           0.0              0.0  \n",
    "\n",
```

```
"      is_host_login is_guest_login  count  srv_count  serror_rate  \\\n",
"0            b'0'         b'0'    2.0       2.0         0.0   \n",
"1            b'0'         b'0'   13.0       1.0         0.0   \n",
"2            b'0'         b'0'  123.0       6.0         1.0   \n",
"3            b'0'         b'0'    5.0       5.0         0.2   \n",
"4            b'0'         b'0'   30.0      32.0         0.0   \n",
"...           ...          ...    ...       ...         ...   \n",
"125968       b'0'         b'0'  184.0      25.0         1.0   \n",
"125969       b'0'         b'0'    2.0       2.0         0.0   \n",
"125970       b'0'         b'0'    1.0       1.0         0.0   \n",
"125971       b'0'         b'0'  144.0       8.0         1.0   \n",
"125972       b'0'         b'0'    1.0       1.0         0.0   \n",
"\n",
"      srv_serror_rate  rerror_rate  srv_rerror_rate  same_srv_rate  \\\n",
"0                 0.0          0.0              0.0           1.00   \n",
"1                 0.0          0.0              0.0           0.08   \n",
"2                 1.0          0.0              0.0           0.05   \n",
"3                 0.2          0.0              0.0           1.00   \n",
"4                 0.0          0.0              0.0           1.00   \n",
"...               ...          ...              ...           ...   \n",
"125968            1.0          0.0              0.0           0.14   \n",
"125969            0.0          0.0              0.0           1.00   \n",
"125970            0.0          0.0              0.0           1.00   \n",
"125971            1.0          0.0              0.0           0.06   \n",
"125972            0.0          0.0              0.0           1.00   \n",
```

```
"\n",
"       diff_srv_rate  srv_diff_host_rate  dst_host_count  dst_host_srv_count  \\\n",
"0              0.00               0.00          150.0              25.0   \n",
"1              0.15               0.00          255.0               1.0   \n",
"2              0.07               0.00          255.0              26.0   \n",
"3              0.00               0.00           30.0             255.0   \n",
"4              0.00               0.09          255.0             255.0   \n",
"...             ...                ...            ...               ...   \n",
"125968         0.06               0.00          255.0              25.0   \n",
"125969         0.00               0.00          255.0             244.0   \n",
"125970         0.00               0.00          255.0              30.0   \n",
"125971         0.05               0.00          255.0               8.0   \n",
"125972         0.00               0.00          255.0              77.0   \n",
"\n",
"       dst_host_same_srv_rate  dst_host_diff_srv_rate  \\\n",
"0                        0.17                    0.03   \n",
"1                        0.00                    0.60   \n",
"2                        0.10                    0.05   \n",
"3                        1.00                    0.00   \n",
"4                        1.00                    0.00   \n",
"...                       ...                     ...   \n",
"125968                   0.10                    0.06   \n",
"125969                   0.96                    0.01   \n",
"125970                   0.12                    0.06   \n",
"125971                   0.03                    0.05   \n",
```

```
"125972                0.30                0.03   \n",
"\n",
"       dst_host_same_src_port_rate  dst_host_srv_diff_host_rate  \\\n",
"0                         0.17                    0.00   \n",
"1                         0.88                    0.00   \n",
"2                         0.00                    0.00   \n",
"3                         0.03                    0.04   \n",
"4                         0.00                    0.00   \n",
"...                       ...                     ...   \n",
"125968                    0.00                    0.00   \n",
"125969                    0.01                    0.00   \n",
"125970                    0.00                    0.00   \n",
"125971                    0.00                    0.00   \n",
"125972                    0.30                    0.00   \n",
"\n",
"       dst_host_serror_rate  dst_host_srv_serror_rate  dst_host_rerror_rate  \\\n",
"0                      0.00                      0.00                  0.05   \n",
"1                      0.00                      0.00                  0.00   \n",
"2                      1.00                      1.00                  0.00   \n",
"3                      0.03                      0.01                  0.00   \n",
"4                      0.00                      0.00                  0.00   \n",
"...                    ...                       ...                   ...   \n",
"125968                 1.00                      1.00                  0.00   \n",
"125969                 0.00                      0.00                  0.00   \n",
"125970                 0.72                      0.00                  0.01   \n",
```

```
   "125971              1.00                 1.00              0.00   \n",
   "125972              0.00                 0.00              0.00   \n",
   "\n",
   "       dst_host_srv_rerror_rate       class  \n",
   "0                      0.00   b'normal'  \n",
   "1                      0.00   b'normal'  \n",
   "2                      0.00  b'anomaly'  \n",
   "3                      0.01   b'normal'  \n",
   "4                      0.00   b'normal'  \n",
   "...                     ...         ...  \n",
   "125968                    0.00  b'anomaly'  \n",
   "125969                    0.00   b'normal'  \n",
   "125970                    0.00   b'normal'  \n",
   "125971                    0.00  b'anomaly'  \n",
   "125972                    0.00   b'normal'  \n",
   "\n",
   "[125973 rows x 42 columns]"
  ]
 },
 "execution_count": 5,
 "metadata": {},
 "output_type": "execute_result"
}
],
"source": [
```

```
 "data_2"
]
},
{
"cell_type": "code",
"execution_count": 6,
"id": "ae5ed70f-f39c-40e6-9cd1-df95a7675995",
"metadata": {
 "tags": []
},
"outputs": [
 {
  "data": {
   "text/html": [
    "<div>\n",
    "<style scoped>\n",
    "    .dataframe tbody tr th:only-of-type {\n",
    "        vertical-align: middle;\n",
    "    }\n",
    "\n",
    "    .dataframe tbody tr th {\n",
    "        vertical-align: top;\n",
    "    }\n",
    "\n",
    "    .dataframe thead th {\n",
```

```
"       text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>duration</th>\n",
"      <th>protocol_type</th>\n",
"      <th>service</th>\n",
"      <th>flag</th>\n",
"      <th>src_bytes</th>\n",
"      <th>dst_bytes</th>\n",
"      <th>land</th>\n",
"      <th>wrong_fragment</th>\n",
"      <th>urgent</th>\n",
"      <th>hot</th>\n",
"      <th>num_failed_logins</th>\n",
"      <th>logged_in</th>\n",
"      <th>num_compromised</th>\n",
"      <th>root_shell</th>\n",
"      <th>su_attempted</th>\n",
"      <th>num_root</th>\n",
"      <th>num_file_creations</th>\n",
"      <th>num_shells</th>\n",
```

```
"    <th>num_access_files</th>\n",
"    <th>num_outbound_cmds</th>\n",
"    <th>is_host_login</th>\n",
"    <th>is_guest_login</th>\n",
"    <th>count</th>\n",
"    <th>srv_count</th>\n",
"    <th>serror_rate</th>\n",
"    <th>srv_serror_rate</th>\n",
"    <th>rerror_rate</th>\n",
"    <th>srv_rerror_rate</th>\n",
"    <th>same_srv_rate</th>\n",
"    <th>diff_srv_rate</th>\n",
"    <th>srv_diff_host_rate</th>\n",
"    <th>dst_host_count</th>\n",
"    <th>dst_host_srv_count</th>\n",
"    <th>dst_host_same_srv_rate</th>\n",
"    <th>dst_host_diff_srv_rate</th>\n",
"    <th>dst_host_same_src_port_rate</th>\n",
"    <th>dst_host_srv_diff_host_rate</th>\n",
"    <th>dst_host_serror_rate</th>\n",
"    <th>dst_host_srv_serror_rate</th>\n",
"    <th>dst_host_rerror_rate</th>\n",
"    <th>dst_host_srv_rerror_rate</th>\n",
"    <th>class</th>\n",
"  </tr>\n",
```

```
" </thead>\n",
" <tbody>\n",
"  <tr>\n",
"   <th>0</th>\n",
"   <td>0.0</td>\n",
"   <td>b'tcp'</td>\n",
"   <td>b'private'</td>\n",
"   <td>b'REJ'</td>\n",
"   <td>0.0</td>\n",
"   <td>0.0</td>\n",
"   <td>b'0'</td>\n",
"   <td>0.00</td>\n",
"   <td>0.00</td>\n",
"   <td>0.0</td>\n",
"   <td>0.44</td>\n",
"   <td>1.00</td>\n",
"   <td>b'anomaly'</td>\n",
"  </tr>\n",
" </tbody>\n",
"</table>\n",
"<p>22544 rows × 42 columns</p>\n",
"</div>"
],
"text/plain": [
"      duration protocol_type      service     flag  src_bytes  dst_bytes  \\\n",
```

```
"0         0.0       b'tcp'   b'private'   b'REJ'       0.0       0.0   \n",
"1         0.0       b'tcp'   b'private'   b'REJ'       0.0       0.0   \n",
"2         2.0       b'tcp'  b'ftp_data'    b'SF'    12983.0       0.0   \n",
"3         0.0      b'icmp'    b'eco_i'    b'SF'       20.0       0.0   \n",
"4         1.0       b'tcp'    b'telnet'  b'RSTO'       0.0      15.0   \n",
"...       ...         ...         ...      ...        ...       ...   \n",
"22539     0.0       b'tcp'     b'smtp'    b'SF'     794.0     333.0   \n",
"22540     0.0       b'tcp'     b'http'    b'SF'     317.0     938.0   \n",
"22541     0.0       b'tcp'     b'http'    b'SF'   54540.0    8314.0   \n",
"22542     0.0       b'udp'  b'domain_u'   b'SF'      42.0      42.0   \n",
"22543     0.0       b'tcp'    b'sunrpc'   b'REJ'       0.0       0.0   \n",
"\n",
"      land  wrong_fragment  urgent  hot  num_failed_logins logged_in  \\\n",
"0     b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"1     b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"2     b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"3     b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"4     b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"...    ...           ...     ...  ...             ...       ...   \n",
"22539 b'0'           0.0     0.0  0.0             0.0      b'1'   \n",
"22540 b'0'           0.0     0.0  0.0             0.0      b'1'   \n",
"22541 b'0'           0.0     0.0  2.0             0.0      b'1'   \n",
"22542 b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"22543 b'0'           0.0     0.0  0.0             0.0      b'0'   \n",
"\n",
```

```
"      num_compromised  root_shell  su_attempted  num_root \\\n",
"0              0.0       0.0         0.0       0.0   \n",
"1              0.0       0.0         0.0       0.0   \n",
"2              0.0       0.0         0.0       0.0   \n",
"3              0.0       0.0         0.0       0.0   \n",
"4              0.0       0.0         0.0       0.0   \n",
"...            ...       ...         ...       ...   \n",
"22539          0.0       0.0         0.0       0.0   \n",
"22540          0.0       0.0         0.0       0.0   \n",
"22541          1.0       0.0         0.0       0.0   \n",
"22542          0.0       0.0         0.0       0.0   \n",
"22543          0.0       0.0         0.0       0.0   \n",
"\n",
"       num_file_creations  num_shells  num_access_files  num_outbound_cmds
\\\n",
"0                0.0       0.0            0.0               0.0   \n",
"1                0.0       0.0            0.0               0.0   \n",
"2                0.0       0.0            0.0               0.0   \n",
"3                0.0       0.0            0.0               0.0   \n",
"4                0.0       0.0            0.0               0.0   \n",
"...              ...       ...            ...               ...   \n",
"22539            0.0       0.0            0.0               0.0   \n",
"22540            0.0       0.0            0.0               0.0   \n",
"22541            0.0       0.0            0.0               0.0   \n",
"22542            0.0       0.0            0.0               0.0   \n",
```

```
"22543              0.0        0.0             0.0             0.0   \n",
"\n",
"      is_host_login is_guest_login  count  srv_count  serror_rate  \\\n",
"0            b'0'          b'0'  229.0      10.0         0.0   \n",
"1            b'0'          b'0'  136.0       1.0         0.0   \n",
"2            b'0'          b'0'    1.0       1.0         0.0   \n",
"3            b'0'          b'0'    1.0      65.0         0.0   \n",
"4            b'0'          b'0'    1.0       8.0         0.0   \n",
"...           ...           ...    ...       ...         ...   \n",
"22539        b'0'          b'0'    1.0       1.0         0.0   \n",
"22540        b'0'          b'0'    2.0      11.0         0.0   \n",
"22541        b'0'          b'0'    5.0      10.0         0.0   \n",
"22542        b'0'          b'0'    4.0       6.0         0.0   \n",
"22543        b'0'          b'0'    4.0      10.0         0.0   \n",
"\n",
"      srv_serror_rate  rerror_rate  srv_rerror_rate  same_srv_rate  \\\n",
"0              0.00         1.0             1.0           0.04   \n",
"1              0.00         1.0             1.0           0.01   \n",
"2              0.00         0.0             0.0           1.00   \n",
"3              0.00         0.0             0.0           1.00   \n",
"4              0.12         1.0             0.5           1.00   \n",
"...             ...         ...             ...           ...   \n",
"22539          0.00         0.0             0.0           1.00   \n",
"22540          0.00         0.0             0.0           1.00   \n",
"22541          0.00         0.0             0.0           1.00   \n",
```

```
"22542          0.00        0.0          0.0         1.00   \n",
"22543          0.00        1.0          1.0         0.25   \n",
"\n",
"      diff_srv_rate  srv_diff_host_rate  dst_host_count  dst_host_srv_count  \\\n",
"0            0.06            0.00         255.0             10.0   \n",
"1            0.06            0.00         255.0              1.0   \n",
"2            0.00            0.00         134.0             86.0   \n",
"3            0.00            1.00           3.0             57.0   \n",
"4            0.00            0.75          29.0             86.0   \n",
"...           ...             ...           ...              ...   \n",
"22539        0.00            0.00         100.0            141.0   \n",
"22540        0.00            0.18         197.0            255.0   \n",
"22541        0.00            0.20         255.0            255.0   \n",
"22542        0.00            0.33         255.0            252.0   \n",
"22543        1.00            1.00         255.0             21.0   \n",
"\n",
"      dst_host_same_srv_rate  dst_host_diff_srv_rate  \\\n",
"0                       0.04                    0.06   \n",
"1                       0.00                    0.06   \n",
"2                       0.61                    0.04   \n",
"3                       1.00                    0.00   \n",
"4                       0.31                    0.17   \n",
"...                      ...                     ...   \n",
"22539                   0.72                    0.06   \n",
"22540                   1.00                    0.00   \n",
```

"22541 1.00 0.00 \n",

"22542 0.99 0.01 \n",

"22543 0.08 0.03 \n",

"\n",

" dst_host_same_src_port_rate dst_host_srv_diff_host_rate \\\n",

"0 0.00 0.00 \n",

"1 0.00 0.00 \n",

"2 0.61 0.02 \n",

"3 1.00 0.28 \n",

"4 0.03 0.02 \n",

"... ... ... \n",

"22539 0.01 0.01 \n",

"22540 0.01 0.01 \n",

"22541 0.00 0.00 \n",

"22542 0.00 0.00 \n",

"22543 0.00 0.00 \n",

"\n",

" dst_host_serror_rate dst_host_srv_serror_rate dst_host_rerror_rate \\\n",

"0 0.00 0.0 1.00 \n",

"1 0.00 0.0 1.00 \n",

"2 0.00 0.0 0.00 \n",

"3 0.00 0.0 0.00 \n",

"4 0.00 0.0 0.83 \n",

"... ... ... \n",

"22539 0.01 0.0 0.00 \n",

```
 "22540              0.01                0.0             0.00   \n",
 "22541              0.00                0.0             0.07   \n",
 "22542              0.00                0.0             0.00   \n",
 "22543              0.00                0.0             0.44   \n",
 "\n",
 "      dst_host_srv_rerror_rate       class  \n",
 "0                     1.00  b'anomaly'  \n",
 "1                     1.00  b'anomaly'  \n",
 "2                     0.00   b'normal'  \n",
 "3                     0.00  b'anomaly'  \n",
 "4                     0.71  b'anomaly'  \n",
 "...                    ...         ...  \n",
 "22539                   0.00   b'normal'  \n",
 "22540                   0.00   b'normal'  \n",
 "22541                   0.07  b'anomaly'  \n",
 "22542                   0.00   b'normal'  \n",
 "22543                   1.00  b'anomaly'  \n",
 "\n",
 "[22544 rows x 42 columns]"
 ]
},
"execution_count": 6,
"metadata": {},
"output_type": "execute_result"
}
```

],
"source": [
"df_2"
]
},
{
"cell_type": "code",
"execution_count": 7,
"id": "316c5da3-adf4-4043-af99-303880d54541",
"metadata": {
"tags": []
},
"outputs": [
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAjoAAAIjCAYAAAAKkbGTAAAAOXRF
WHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjcuMiwgaHR0cHM6Ly9tY
XRwbG90bGliLm9yZy8pXeV/AAAACXBIWXMAAA9hAAAPYQGoP6dpAACQf
ElEQVR4nOzdeVxU1f8/8NeAgIAwsiOJghuCYCmWIimYiuZGWmmhKC6ouaKi6a
fcyi33UnNPNBcq99IQVxR3UTTcd3DBFcEFAeH8/vDH/ToMInMZVK6v5+Mxj+L
OmfecGWd5z7nnvI9KCCFAREREpEAGb7oDREREREpEh0rSDQyQEhMHXiN2TWx
wmOkRERKRYTHSIiIhIsZjoEBERkWIx0SEiIiLFYqJDRERErEisVEh4iIiBSLiQ69ky
IiIqBSqaRL6dKl4ejoiEaNGmHixIm4ffu2u3GjBkDlUqlcSwzMxO9e/dGuXXLlYGh
oiA8++AAAcP/+fXz11Vewt7eHSqXCZ5999hoelTybbN2/GmDFj9B533759GDNmD

B48eKB13a+//oqIiAi932dh5f77Hzly5LXeb0hICFxcXDSOqVQqnZ9/uf9mee+rOJ6H
GzduYMyYMYiPj9e6Lr/3EFFxY6JD77QlS5Zg//792Lp1K+bMmYMPPvgAP/30E9z
d3bFt2zaNtj169MD+/fs1js2dOxfz58/Hd999h9jYWPz+++8AgB9//BHr1q3DjBkzsH//
fkyePPm1PSZdbd68GWPHjtV73H379mHs2LFvZaLzNtm/fz969Oih023k/pvJuS9d3b
hxA2PHjs030cnvPURU3Eq96Q4QvUmenp6oU6jHq6eO9ePO9Pfnn3+OQYMG4ePPP0a7du1
w/vx5ODg4AADKly+P8xKfz+P8uXLa9w+ISEBpqam6NevVr7ty5o9EHFxcXXtGrVCmvXrkWtWrVQunRp6Rd8cnIyevXqhfLLy
8PExAR2dnbo2rUr7ty5o9EHFxcXtGrVCmvXrkWtWrVQunRp6Rd8cnIyevXqhfLLy
8PY2Biurq4YO3Ysnj17Jt3+ypUrUKlUmDp1KqZPnw5XV1eUKVMGPj4+OHDgg
NQuJCQEc+bMkR5P7uXKlSsvfX62bt2KwMBAlC9fHqVL0aVKlXQq1cv3L17V+
O5Gjp0KADA1dVV43lwcXHByZMnERMTIx3PPZ3z9lTBkyBB988AHUajWsr
a3h4+ODDRs2aPVDpVKhX79+mDdvvHtzd3WFiYoKlS5cCAM6cOYOvv/4aDg4O
MDExQYUKFdC5c2dkZGS89HHdvHkTkT3t7eqFq1Ks6fPw8AuHTpEr766is4OTnBx
MQEDg4OaNy4cb6jFXFRETAzc0NJiYmcHd3x7x7Jly/Jtl/d00pMnTxAeHg5XV1eUL
l0a1tbWqFOnDlatWgXg1f9mBT0vLzZo1oLl26pNHGxc
UFISEhWrf19/eHv78/AGDXrl348MMPAQBdu3aV+pZ7n7dTrvVdgZrUUtFRaF27dowN
TVF9erV8dtvv2m0+0e9XzQvQyHNEhykeLFi1gaGiI3bt3v7TN/v/v8/J7jfmIJXoHu
MHgOdf6vv370efPn2QmpqKFStWAAA8PDyQk5ODwmBA7NmzB8OGDUP9+vV
x9epVjB49Gv7+/jhy5IjGaMRo0dx+vRpfP/993B993B1dYW5uTmSk5Px0UcfwcDAAK
NGjUKLypWXf/9+jBs3DleuXMGfBACMHDkSLVq0wOX
Ll6FWqzFy5Ey5Eg8fvwYXq3Gl1jev1jilUK5cuZc+7osXL8Lhw8Hwe+zqXL8LHxwc9evSAWq3GlStXMH36dH
z88cf477//7//YGRkhB49euDQDUrK0p9MH5fMc1OvKRdxwt6EBPGwHnCLAfKqeobPoUM7n
FAJiYmAAAMjlycP/+fYSHh+O9995DZmYtm3bhnmHJkiXo3LmzRl/Wr1+P8Px0Xv2+PP
Xv2YNSoUXB0dIS9vT2Ohz+Ojz/+GLa2tvjhhx9QtWpVV3Lx5Exs3bkRmZqZ0Xy9

KSEhAixYtUL58eezfvx+2trbS6yA7OxuTJ09GhQoVcPfuXezbty/f03EvioiIQNeuXR
EYGIhp06YhNTUVY8aMQUZGBgwMCv59OXjwYPz+++8YN24catWqhcePHyM
hIQH37t2T/g1f9W+W3/NSkO7du6Np06ZYuXIlkpKS8P3338Pf3x8nTpxA2bJlC7zti
2rXro0lS5aga9eu+P7779GyZUsAeOUozvHjxzFkyBAMHz4cDg4OWLRoEbp3744q
VaqgYcOGhXpeiF5KEL2DlixZIgCIw4cPv6Pv7SNg4cPv7SNg4ODcHd3l/4ePXq0yPuW6dKlizA3
N9e6rZ+fn6hRo4bGsVsHWsVWrVVgkAYs2aNRrHDx8+LACIX3/9VTpWsWJFYWWhoKM6
ePavRtlevXqJMmTLi6tWrGsenTp0qAIiTJ08KIYS4fPmyACC8vLzEs2fPpHaHDh0S
AMSqVaukY3379tV6XIWVk8j36UKlUiI2NRUREBH7++WdzMfW9wW+usm
XL4ubNm1qnyY4dOyZ7YUrZsmW1vkuKWvMqL87RKUCZMmUQFxcHNzc3vcY
tSUuh09LS0KJFC5w8w8eRIPHz6Ek5MTkpOT4ePjg82bN8tacXDjxg00atQIhoaGOH/+
POrUqSNVGd29e7esmkJ5V1wIIXDz5k2MGTMGZZxcNMCnq/m69K1C/7++2/pyzgrKwuBgYGIiIiAWq2
W3edcFy9exLhx44pU7Z4+by6Y8eOwc/CgpCnTp1MHjwYIwfPx4/////wzAgMDsXXrVtSuXb
vQk5ELW48HgOyiiXknsGZlZeHYsWMYOXIko8fr1WYND9jx47F0KFDYWZm
Vmzz1HKtW7cO06ZN06h7NHToUnsfW9wW+uYcOGYf/+/fjrr79QrVo1HD16F
Ldu3ULnzp3RuXnNnWc9Deno6cnJypO+SK1euYP369X3B3d0ezZs1k9TMvJjoFaNSo
Eb777js0adLkTXfllSpVqoTDw9d6bg6dO3bg6NGjUuLXvyvVi9+5Ww74AtasqVOcrKwsBAQEY
pWLVqlUbMolQZZW8yshACzs7OiIyMlFX2/XV48uuRJvitYatasqVOcrKwsBAQEY
P78+ahWrZo+uwi1Wo2jR4+icuXKGonmcxmbmhqdPnxYp/oULF3Dq1CkAz0dJ
8q5E08X9+/elSci7ldu3CyZMnYtb27Z1GnNS0fRBo27evrLiBgYF48OABVq1S
N169fR8eOHWFlZaXZmjExcXmppi1qxZ6NatvyZ+a+5bm7tSSn1R0+4uZ1//59PH3
6FE5OTsjJycHUhH/kyJGwsrKSF9d12r17NWoS4uLLhC3yY7OxuxsbGoW
bNmiXiMwPPCfocOHcIdIqX3Hu3Dl89NFHsLKyKT+Dp1r31VdfYcGYkoVnz5
5JFajlIUICAhAhAu3bt0Lt3bzx58gQbNmzAmTJkzB69Gh06tRJo9tPnpTTzG4n
5JFajllIUICAhAu3bt0Lt3bzx48ADVq1eHkZER7t69S+nTpxdpB4JcTHQKcPHiRfTu

3RudOnWCp6en1hCwrl9GuYpjKbSBgQGSk5O1RkNu3bqFChUqyF5W+6KnT5/Cx
MREb6fw9Cnvh7uBgQHs7OxQpUoVrWqmRfHo0SOtuiRyTgXcuXMHXbt2xb///pv
v9XJ+ddvZ2WHfvn163+/LwcEBUVFRqFWrlkaiEx0dje7du0srhuRYvHgxZsyYIa20
q1q1KsLCwtCjRw9Z8QwNDWFra4sGDRpIp6v0Uao+KSkJgYGYGBSEhIgLOzs3QK1
8vLCxs2bNCpkuvVq1chhEClSpVw6NAh2NnZSdcZGxvD3t5edh2hkqZbt274+eefY
WFhoXH88ePH6N+/P3777Te93t/p06fx4Ycf6lzstXTp0jh9+nShVzbJoc+6Nkp
ZHWqb/w8HCkp6djzpw5RerrpUuXcOOTIEahUKtSqVatIP05sbW0RExODGjVqYNG
iRZg1axxOHTuGNWvWYNSoUYVedVYQztEpwJ07d3Dx4kWNMuUqlarIQ8AuLi
56Wwr94oZvHEvbt24tUZTcnJwfjx4/HvHnzcOvWLanmzciREiRI+Hi4iJ7
Gezvv/+O+fPn49KlS9JmhjNmzEClSpVkDdfXXr2idI3yKM6EJDk5GXPnztWYU9a3b
19pYmZJI2dPouzsbKxfv16jEGFgYKDsCu7ymuE3bt2Cs7MzMjMjMzZ
cfu1q0bfH19WRiwGPHU1RvQo0cPrFy5EiNHjnzTXXmly5cv5cv6z3m5MmTMXToU
MydOxeenp56j/+uKlOmDOLj4zF//nHaNeuHfr27YusrCxZMbOyshAQE
ID58+crqhy9rr/vLly4gBYtWuD69etwc3OEELar23Tpk2yksiSt2rjxo3S/Zsk
WjAnJ2dja2b99e6B+DLzN79mx8+eWX2LNnj15ui8FBfBg8ejB9//h+k3QdfNF1u0aAEhBFa
sWAFra2sAz5/XTp06wAz5/XTp06wcDAAJs2bdK06R3R3R0VNBnVV6F
+Yw9fvy4Vj2mFxkZGpk1Dq1atCn3feSeS1atAi9e/eGqakpk1DSyZsOsyjnNR
to4ovMGnDhxAh988988AGA53u9vOhtmJh87NgxaQSgoB2v5fb1bSuIqBv+83y6NEjl
C5dWnbczpZ07Y/HixXrfYYiwtLYSHbmBY3JiwDB6QkB3h+CnrSpEnw9fWVFFPy5cuwbX
VOPb555/Dzc0NcXFxRervuyjjq64jvZ1VcXByys7OlCtnnzp2DoaFhfc9q127tpSIrq64v
Dhw1r/Xvrw/fff44cffDw4cNhYGCg9/jERON2Llz55vuQoFe7F9x9JWF0PQr95esS
qXCqFGjNFZeWdn4+DBg1JiLUdx7TBdkpiYmORbZffRo0cwNjaWFbNixYoAtCt
Z16hRg6d0ZXjxs2r69OmwLDA0qVLpaJaxJaJ7KSkp6Nq1Kxo0aFAFCoeGXLsXly5dhb2+
PxMREnU93J3Z6ZiY6dOjAJKcY8dQVvRF5J3Z6eHigTZs2sid2vstyl3jHxjHxMTAx8d

H44vX2NgYLi4uCA8Pl33qqbi2QnmTdD111blzZxw9ehSLFy+Wqi4fPHgQoaGh8P
b2RkREhM59KElba5Q07733HqKjo1GjRg2N4wkJCQgICChUodOePXti6dKlcHJyQ
mJiIsqXL//Sz6einF4aNGgQ7Ozs8L///U92DCoYR3TotSuOiZ3vstxfsl27dsXPP/+s92
XQb/sIpBwNGjTQmhtWkF9++QVdunSBj4+PNKfu2bNnaNOmDX7++WdZfRg0aB
BKlSqFxMREuLu7S8c7dOiAQYMGMdEpgrS0NNy6dUsr0bl9+3a+I3P5WbBgAdq
1a4cLFy5gwIABCA0NLZbaX9nZ2Zg8eTK2bNnyVs7ZVAKO6NBrVxwTO4mA5+
X3jYyMpM0XN2zYgCVLLsDDwwNjxoyRfZop14ULF3D69GkIIeDh4YEqVarIjlW
SttYoaTp37oyYmBhMmzZNKmNx4MA06FA0bNsTSpUt1ite1a1f88ssvxZLoK
HHE9G3DRIdeO3Nzcxw4cEBrJ+Djx4/D19eXH/BvocOHD+Ovv/5CYmKiVs2QtW
vXvqFeafvfw4www8xfPhwfP7557h06okOjq9Ub1q/rfTw9lIcHvMIBuO6tDWYhLsv+6BcMmLmo8YkxmiFWRvzMpFZ8vV8nNDfTPe3f5gdjsarXQ/+dK86w/2c/1L//M3UT+/mQdl3Q2n2VslDWzIB1g==
"text/plain": [
"<Figure size 640x480 with 1 Axes>"
]
},

```
   "metadata": {},
    "output_type": "display_data"
   }
  ],
  "source": [
   "data['normal'].value_counts().plot(kind = 'bar', title = 'Different attarcks distributins',
xlabel= 'atarcks');"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 8,
  "id": "2ae4e738-c461-4599-a654-d09f710c49af",
  "metadata": {
   "tags": []
  },
  "outputs": [
   {
    "data": {
     "text/plain": [
      "normal\n",
      "normal             67342\n",
      "neptune            41214\n",
      "satan               3633\n",
      "ipsweep             3599\n",
```

```
 "portsweep          2931\n",
 "smurf              2646\n",
 "nmap               1493\n",
 "back                956\n",
 "teardrop            892\n",
 "warezclient         890\n",
 "pod                 201\n",
 "guess_passwd         53\n",
 "buffer_overflow      30\n",
 "warezmaster          20\n",
 "land                 18\n",
 "imap                 11\n",
 "rootkit              10\n",
 "loadmodule            9\n",
 "ftp_write             8\n",
 "multihop              7\n",
 "phf                   4\n",
 "perl                  3\n",
 "spy                   2\n",
 "Name: count, dtype: int64"
]
},
"execution_count": 8,
"metadata": {},
"output_type": "execute_result"
```

```
   }
  ],
  "source": [
   "data['normal'].value_counts()"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "943882e1-03eb-452a-827e-8ef7672942e2",
  "metadata": {
   "tags": []
  },
  "source": [
   "The attack types are grouped based on their characteristics and objectives. Here's why each attack belongs to its chosen group:\n",
   "\n",
   "#### DOS (Denial of Service)\n",
   "These attacks aim to disrupt the availability of a service by overwhelming the target with excessive requests or exploiting vulnerabilities\n”.
```