ADVANCED DIGITAL DNA SEQUENCE ENGINE FOR PROACTIVE RANSOMWARE DETECTION AND MITIGATION USING MACHINE LEARNING

CO8811 – PROJECT REPORT

Submitted by

ADITHYAN K.S	**211420118001**
DONALD COLIN D	**211420118017**
HARIHARAN S	**211420118021**

in partial fulfillment for the award the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER AND COMMUNICATION ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MARCH 2024

BONAFIDE CERTIFICATE

Certified that this project report **"ADVANCED DIGITAL DNA SEQUENCE ENGINE FOR PROACTIVE RANSOMWARE DETECTION AND MITIGATION USING MACHINE LEARNING"** is the bonafide work of **ADITHYAN K.S (211420118001), DONALD COLIN D (211421118017), HARIHARAN S (211421118021)** who carried out the project work under my supervision.

SIGNATURE

Dr.B.ANNI PRINCY M.E., Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR,
COMPUTER AND COMMUNICATION ENGINEERING,
PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI, POONAMALLEE,
CHENNAI- 600123.

SIGNATURE

Dr.A.DHANALAKSHMI,M.E.,Ph.D.,

SUPERVISOR

ASSOCIATE PROFESSOR,
COMPUTER AND COMMUNICATION ENGINEERING,
PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI, POONAMALLEE,
CHENNAI- 600123.

Certified that the above candidate(s) was/ were examined in the End Semester Project Viva-Voce Examination held on.....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors Tmt. **C.VIJAYARAJESWARI, Dr. C. SAKTHIKUMAR, M.E., Ph.D**., and **Dr.SARANYASREE SAKTHIKUMAR B.E., M.B.A., Ph.D**., for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.,** for his timely concern and encouragement provided to us throughout the course.

We thank our HOD of Computer and Communication Engineering Department, **Dr. B. ANNI PRINCY, M.E., Ph.D**., Professor, for the support extended throughout the project.

We would like to thank our supervisor, **Dr.A.DHANALAKSHMI,M.E.,Ph.D**.,, Associate Professor, and all the faculty members of the Department of Computer and Communication Engineering for their advice and suggestions for the successful completion of the project.

ADITHYAN K.S

DONALD COLIN D

HARIHARAN S

ABSTRACT

Malicious software, commonly known as malware, constitutes a pervasive and ever-evolving threat landscape in today's digital ecosystem. These nefarious programs, ranging from viruses and Trojans to spyware and ransomware, fuel a multitude of cyber attacks, including cybercrime, fraud, scams, and even nation-state cyber warfare. Among these, ransomware has emerged as one of the most insidious and damaging forms of malware, encrypting victims' data and extorting ransom payments for its release.

Addressing the complex challenges posed by malware necessitates a deep understanding of their development, deployment strategies, and evasion techniques. In response to this imperative, our research introduces DNA act-Ran, an innovative solution that leverages Digital DNA sequencing principles in conjunction with k-means frequency vectors for ransomware detection. By harnessing the power of machine learning (ML) algorithms, DNA act-Ran aims to discern subtle behavioral patterns indicative of ransomware activity, thus enabling proactive defense mechanisms.

The development and proliferation of ransomware variants pose a formidable challenge to traditional signature-based detection methods. To overcome this challenge, DNA act-Ran offers a dynamic and adaptive approach capable of analyzing and identifying ransomware instances with unprecedented accuracy and efficiency. By training on a diverse dataset comprising 582 ransomware samples and 942 benign

software samples, DNA act-Ran undergoes rigorous evaluation, yielding impressive performance metrics across key indicators such as accuracy, recall, precision, and F-measure

Our findings demonstrate that DNA act-Ran outperforms existing ransomware detection methods, offering a robust defense against this evolving threat landscape. By deciphering the intricate nuances of ransomware behavior, DNA act-Ran enables organizations and individuals to proactively mitigate the risks posed by ransomware attacks, safeguarding critical data and infrastructure.

Furthermore, the significance of DNA act-Ran extends beyond its immediate applications in ransomware detection. By shedding light on the intricate interplay between malware behavior and machine learning algorithms, our research contributes to the broader discourse on cybersecurity innovation and defense strategies. By embracing innovative approaches such as DNA act-Ran, we can fortify our defenses against emerging cyber threats, fostering a safer and more resilient digital environment for all stakeholders.

In conclusion, DNA act-Ran represents a paradigm shift in ransomware detection technology, heralding a new era of proactive cybersecurity defense. By combining Digital DNA sequencing with advanced ML algorithms, DNA act-Ran empowers organizations to stay ahead of the curve in the relentless battle against malware, ensuring the integrity and security of digital assets in an increasingly interconnected world

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

S.NO	ABBREVIATION	EXPANSION
1	MOGWO	Multi-Objective Gray Wolf Enhancement
2	SNM	Support Vector Machines
3	BCS	Binary Cuckoo Search
4	AES	Advanced Encryption Standard

CHAPTER 1

INTRODUCTION

As we continue to explore new possibilities and push the boundaries of what is possible, we are constantly amazed by the potential for growth and innovation. The world is full of opportunities to seize and it is up to us to take the plunge and make our mark. With determination and perseverance, we can achieve greatness and leave a lasting impact on the world around us. Embrace the unknown and embrace the journey ahead, knowing that the possibilities are endless.

As we embark on this journey of discovery and innovation, we must remember that the path to success is not always easy. There will be challenges and obstacles along the way, but it is when we overcome these obstacles that we truly grow and develop. It is important to stay focused and determined, never losing sight of your goals and aspirations.

By staying true to ourselves and our vision, we can make a difference in the world and leave a legacy that inspires others to follow in our footsteps. Let us embrace the challenges that come our way, knowing that each challenge is an opportunity to grow and learn. As we navigate the complexities of our journey, it is important to maintain a positive mindset and be ready to adapt to new challenges. Adopting a growth mindset allows us to approach obstacles with resilience and creativity, turning setbacks into opportunities to innovate.

By promoting a culture of learning and continuous improvement, we can push the boundaries of what is possible and achieve remarkable breakthroughs in our quest for knowledge and discovery. Let us embrace the unknown with open arms, knowing that every step we take will bring us closer to realizing our full potential and making a

lasting impact on the world. So let's talk about using SVM algorithms in machine learning to protect DNA datasets. Essentially, this method involves implementing a support vector machine (SVM) model to classify and predict DNA sequences based on certain characteristics such as nucleotide base pairs.

By using SVM, researchers can effectively analyze and protect sensitive genetic information from potential threats or security breaches. The method is robust, reliable, and can be adapted to process large volumes of genomic data with high accuracy.
In simple terms, using SVM algorithms in machine learning to protect DNA datasets not only ensures privacy and security but also improves the overall efficiency of genetic research and analysis. Furthermore, the use of SVM algorithms in machine learning to protect DNA data opens up new possibilities for advances in the fields of personalized medicine and genetic therapy. With the ability to accurately classify and predict DNA sequences, researchers can identify potential genetic markers of disease, develop targeted treatments, and improve patient outcomes.

This innovative approach not only improves the security of genetic information but also paves the way for revolutionary discoveries in the field of genomics. By harnessing the power of SVM algorithms, the future of genetic research looks bright and full of potential for transformative breakthroughs.

1.1 SCOPE OF PROJECT

In the face of an unprecedented surge in ransomware attacks, there is a global imperative for governments, enterprises, and individuals to fortify their cybersecurity resilience and adopt proactive defense strategies. This paper introduces DNA act-Ran, an innovative and ambitious endeavor aimed at revolutionizing ransomware detection and classification on a monumental scale. By synergizing state-of-the-art machine learning algorithms with cutting-edge digital DNA sequencing methodologies, DNA act-Ran aspires to transcend conventional cybersecurity paradigms, offering a comprehensive and adaptive solution to combat ransomware threats with unparalleled efficacy. Through meticulous research and development, DNA act-Ran seeks to empower stakeholders worldwide with the tools and insights necessary to safeguard critical data and infrastructure against the ever-evolving ransomware menace.

Through this ransomware by sequencing its computerized DNA utilizing a dynamic ML approach. DNA act-Ran first chooses key highlights from the prepossessed information utilizing Multi-Objective Gray Wolf Enhancement (MOGWO) and Binary Cuckoo Search (BCS) calculations. From that point, the computerized DNA arrangement is created for the chosen highlights utilizing the plan limitations of DNA arrangement and k-means recurrence vector.

CHAPTER 2

LITERATURE SURVEY

OVERVIEW

Title 1: Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption.

Author: M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou.

Year:2023

Abstract:

Personal health record (PHR) is an emerging patient-centric model of health information exchange, which is often outsourced to be stored at a third party, such as a cloud providers. In this paper, we propose a novel patient-centric framework and a suite of mechanisms for data access control to PHRs stored in semi-trusted servers. To achieve fine-grained and scalable data access control for PHRs, we leverage attribute-based encryption (ABE) techniques to encrypt each patient's PHR file. A high degree of patient privacy is guaranteed simultaneously by exploiting multiauthority ABE. Our scheme also enables dynamic modification of access policies or file attributes and supports efficient on-demand user/attribute revocation and break-glass access under emergency scenarios. Extensive analytical and experimental results are presented which show the security, scalability, and efficiency of our proposed scheme

Title 2: An SMDP-Based Service Model for Inter-domain Resource Allocation in Mobile Cloud Networks

Author: H. Liang, L. X. Cai, D. Huang, X. Shen, and D. Peng

Year:2021

Abstract:

Mobile cloud computing is a promising technique that shifts the data and computing service modules from individual devices to geographically distributed cloud service architecture. In this paper, we propose a service decision-making system for inter-domain service transfer to balance the computation loads among multiple cloud domains. To this end, we formulate the service request decision-making process as a semi-Markov decision process. The optimal service transfer decisions are obtained by jointly considering the system incomes and expenses. Extensive simulation results show that the proposed decision-making system can significantly improve the system rewards and decrease service disruptions compared with the greedy approach.

Title 3: Exploiting Geo-Distributed Clouds for an E-Health Monitoring System with Minimum Service Delay and Privacy Preservation

Author: Q. Shen, X. Liang, X. Shen, X. Lin, and H. Luo

Year:2022

Abstract:

In this paper, we propose an e-health monitoring system with minimum service delay and privacy preservation by exploiting geo-distributed clouds. In the system, the resource allocation scheme enables the distributed cloud servers to cooperatively assign the servers to the requested users under the load balance condition. Through the numerical analysis, we show the efficiency of the proposed traffic-shaping algorithm in terms of service delay and privacy

preservation. Furthermore, through the simulations, we demonstrate that the proposed resource allocation scheme significantly reduces the service delay compared to two other alternatives using jointly the short queue and distributed control law.

Title 4: Secure Dynamic Searchable Symmetric Encryption with Constant Document Update Cost

Author: Y. Yang, H. Li, L. Wenchao, H. Yang, and W. Mi

Year:2022

Abstract:

With the development of cloud computing, data sharing has a new effective method, i.e., outsourced to a cloud platform. In this case, since the outsourced data may contain privacy, they are only allowed to be accessed by authorized users. In this paper, we leverage the secure k-nearest neighbor to propose a secure dynamic searchable symmetric encryption scheme. Our scheme can achieve two important security features, i.e., forward privacy and backward privacy which are very challenging in the Dynamic Searchable Symmetric Encryption (DSSE) area. In addition, we evaluate the performance of our proposed scheme compared with other DSSE schemes. The comparison results demonstrate the efficiency of our proposed scheme in terms of the storage, search, and update complexity.

Title 5: Achieving Usable and Privacy-assured Similarity Search over Outsourced Cloud Data

Author: C. Wang, K. Ren, S. Yu, and K. M. R. Urs

Abstract:

In this paper, we investigate the problem of secure and efficient similarity search over outsourced cloud data. Similarity search is a fundamental and powerful tool widely used in plaintext information retrieval but has not been quite explored in the encrypted

data domain. We formally prove the privacy-preserving guarantee of the proposed mechanism under rigorous security treatment. To demonstrate the generality of our mechanism and further enrich the application spectrum, we also show our new construction naturally supports fuzzy search, a previously studied notion aiming only to tolerate typos and representation inconsistencies in the user searching input. The extensive experiments on the Amazon cloud platform with real data sets further demonstrate the validity and practicality of the proposed mechanism.

Title 6: Malware Classification using Machine Learning and Deep Learning.

Author: Rushiil Deshmukh, Angelo Vergara, Debtanu Bandyopadhyay, Kevin Huang

Year:2020

Abstract:

Malware, derived from Malicious software, is a comprehensive term encompassing any software intentionally crafted to disrupt, damage, or illicitly access computer systems. It 's critical to determine whether a file includes malware. The increase in malware is causing a lot of problems for businesses, including data loss and other problems. Malware can swiftly inflict significant damage to a system by slowing it down and encrypting a sizable amount of data on a personal computer. This suggests that lowering the number of false positives is important. Different machine learning methods, including decision trees and random forests, are used in our malware detection process. Additionally, the confusion matrix is used to calculate the false positive and false negative rates, which is how the system\'s performance is determined.

Title 7 : Malware Detection & Classification Using Machine Learning

Author: Sanket Agarkar, Soma Ghosh

Year:2021

Abstract:

In today's internet world, malware is still the most harmful threat to internet users. Polymorphic malware is a form of malware that constantly modifiesits recognizable features to fool detection using traditional signature-based models. Now everyone wants to get a behavioral pattern that can be derived from static analysis or dynamic analysis, with these patterns machine learning models can be used to predict whether it is malware or not or identify its family of malware.

Title 8: Malicious sequential pattern mining for automatic malware detection

Author: Yujie Fan , Yanfang Ye , Lifei Chen

Year:2022

Abstract:

Due to its damage to Internet security, malware and its detection have caught the attention of both the anti-malware industry and researchers for decades. However, this method fails to recognize new, unseen malicious executables. The developed data mining framework composed of the proposed sequential pattern mining method and ANN classifier can well characterize the malicious patterns from the collected file sample set to effectively detect newly unseen malware samples.

Title 9: Achieving Usable and Privacy-assured Similarity Search over Outsourced Cloud Data

Author: C. Wang, K. Ren, S. Yu, and K. M. R. Urs

Year: 2023

Abstract:

In this paper, we investigate the problem of secure and efficient similarity search over outsourced cloud data. Similarity search is a fundamental and powerful tool widely used in plaintext information retrieval but has not been quite explored in the encrypted data domain. We formally prove the privacy-preserving guarantee of the proposed mechanism under rigorous security treatment. To demonstrate the generality of our mechanism and further enrich the application spectrum, we also show our new construction naturally supports fuzzy search, a previously studied notion aiming only to tolerate typos and representation inconsistencies in the user searching input. The extensive experiments on the Amazon cloud platform with real data sets further demonstrate the validity and practicality of the proposed mechanism.

Title 10 : A Survey of Computer Virus Defense Mechanisms:

Author(s): U. Hengartner, P. Steenkiste

Year: 2023

Abstract:

This paper offers a comprehensive overview of defense mechanisms against computer viruses. It covers signature-based detection, which identifies known virus patterns, as well as behavior-based detection, which observes software behavior for anomalies. The paper also explores hybrid approaches combining these techniques for enhanced protection. It provides insights into the strengths and weaknesses of different defense mechanisms, offering valuable guidance for researchers and practitioners in the field of cybersecurity.

2.2 EXISTING SYSTEM

Ransomware is one of the most far and wide malware digital assault classes that holds the casualty's information prisoner by clandestinely encoding the information on a client's PC to make it inaccessible and just decodes the information after the client pays a payment as an amount of money. Ransomware activity fluctuates as indicated by the group of ransomware. The activity can be; denied admittance to information documents of clients by encoding them or assuming control over the boot cycle of a framework and debilitate admittance to the framework. Ransomware is a subcategory under malware, and there are a few conduct contrasts between malware and ransomware. A huge distinction is that malware expects to stay stowed away from the client for as far as might be feasible, to proceed with its capacities in a covertness mode. While the main role of ransomware is to appear to the client, what's more, makes him mindful of the disease. Winery ransomware is the latest effective ransomware assault. The ransomware abused the weakness named Eternal Blue present in the SMB convention of Windows frameworks and contaminated huge numbers of client eyerywhere in the world. Client information was encoded, and Bitcoin installments were requested in return for the decoding key.

LIMITATIONS

- Despite the advancements in ransomware detection and mitigation, significant challenges persist. CryptoWall, notorious for its destructive capabilities across both 32-bit and 64-bit systems, represents a formidable obstacle to existing security measures. Similarly, early iterations like WinLock underscore the historical prevalence of locker ransomware threats. Moreover, the relentless evolution of ransomware, with increasingly sophisticated variants emerging regularly, poses an ongoing challenge to cybersecurity efforts. As attackers refine their tactics and techniques, staying ahead of the curve remains a perpetual struggle for defenders, highlighting the need for continuous

innovation and adaptation in the face of evolving threats.

- WinLock is another early example of locker ransomware
- Even though Ransomware continues to evolve, more sophisticated variant strains are being deployed all the time.

CHAPTER 3

SYSTEM DESIGN

The progressed advanced DNA sequencing motor for proactive ransomware discovery and relief utilizing machine learning may be a modern framework outlined to distinguish and anticipate ransomware assaults on computerized frameworks. This framework leverages the control of machine learning calculations to analyze DNA-like designs in computerized information and recognize potential ransomware assaults sometime recently can cause noteworthy harm. The framework works by ceaselessly observing organized activity, record frameworks, and other basic information sources for any suspicious movement. It utilizes progressed machine learning models to analyze the information and identify any irregularities or designs that demonstrate the nearness of ransomware. The DNA sequencing motor particularly alludes to the algorithmic approach utilized to distinguish and classify these designs.

Once a potential ransomware assault is recognized, the framework instantly takes proactive measures to relieve the risk. This may include segregating the tainted machine, ending suspicious forms, reestablishing scrambled records from reinforcements, or indeed quarantining the affected system from the organization. The biggest advantage of this framework is its capacity to proactively identify and relieve ransomware assaults, advertising a better level of security than conventional responsive approaches. By analyzing the special DNA-like designs of ransomware, it can recognize modern and obscure variations, giving a more vigorous defense against developing dangers. The system's machine learning models are ceaselessly prepared and upgraded to remain ahead of advancing ransomware procedures and guarantee ideal discovery adequacy. It works in real-time, giving quick reactions to potential dangers and minimizing the potential effect on advanced frameworks. Generally, the progressed advanced DNA sequencing motor for proactive ransomware discovery and moderation utilizing machine learning could be an effective arrangement that

combines cutting-edge innovation and shrewd calculations to secure advanced frameworks from the annihilating impacts of ransomware assaults.

PROPOSED SYSTEM ARCHITECTURE DESIGN

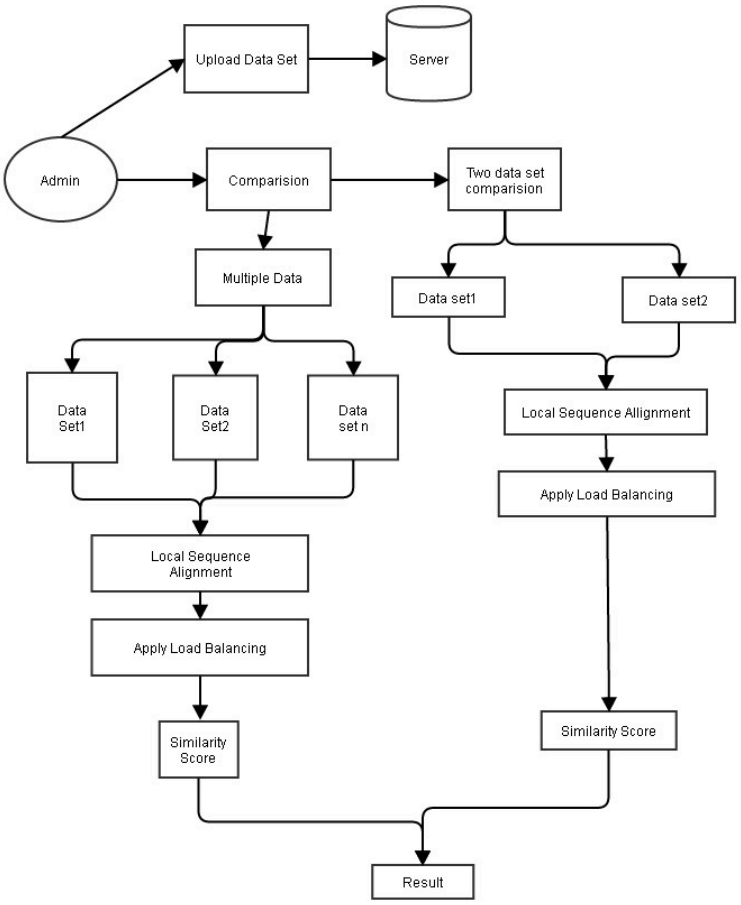


Figure 3.1: Proposed system architecture

Ransomware assaults have been expanding all through ongoing years, and numerous strategies have been proposed to recognize and forestall them. Most current ransomware discovery, what's more, investigation techniques fall into two principal classifications - dynamic or static approaches. Dynamic investigation techniques include establishing a disconnected climate and running the malware inside that climate to perceive its utilitarian conduct. Static methodologies, on the other hand, include switch designing the malignant code to comprehend the working of the

malware and afterward to develop defenses against it. Saundra et al. proposed an Elder Ran tool that checks trademark ransomware marks by investigating a bunch of activities during the underlying periods of the assault stream execution chain. Elder progressively identifies and characterizes ransomware by investigating activity exercises, for example, vault key tasks, calls from Windows APIs, and organizer and record framework tasks. Elder Ran uses Logical Regression classifier calculation an ML calculation, to characterize every client application, and has extra usefulness to recognize and make marks for up until now obscure ransomware.

The proposed Elder Ran tool is a ransomware detection and classification system that uses a combination of dynamic analysis, feature extraction, and machine learning algorithms to identify and mitigate ransomware attacks. The following is an explanation of the key components and functionalities of the tool

Dynamic Analysis

Elder Ran tool implements a dynamic analysis technique in which a virtualized environment is created to execute the malware in a controlled environment. During the execution of the ransomware, the tool monitors and logs its behavior and activities. By analyzing this data, the tool can identify the patterns and activities associated with ransomware behavior.

Feature Extraction

Relevant features are extracted from the dynamic analysis data such as registry key operations, file system operations, calls from Windows APIs, and other behavioral indicators of ransomware attacks. These features are then used as input for the machine learning algorithms.

Machine Learning Algorithms:

Elder Ran tool uses a Logical Regression classifier algorithm, which is a type of supervised learning algorithm that can classify data based on a set of input features. The algorithm is trained on a large dataset of known ransomware attacks and benign activities to develop accurate classification models.

Ransomware Detection and Classification:

During the runtime of the tool, Elder Ran checks the extracted features against the trained classifier model to identify and classify the ransomware under analysis. The tool is also capable of detecting and creating signatures for new and unknown ransomware by identifying key features of their behavior patterns.

Mitigation Strategies:

Once ransomware is detected, the Elder Ran tool can be set to automatically initiate a response, such as isolating the affected system, shutting down the system, or disconnecting the network to arrest the expansion of ransomware in a network. The Elder Ran tool is an effective security tool for ransomware detection and classification. It employs a combination of dynamic analysis and machine learning algorithms to accurately identify ransomware signatures and classify the attacks. This tool can work against known as well as previously undiscovered ransomware to rapidly recognize and prevent the spread of the malware.

Packet inspection

It involves examining the contents of data packets transmitted over a network to detect and analyze potential threats or malicious activity. Traffic analysis entails monitoring network traffic patterns to identify anomalies or suspicious behavior. Signature-based detection relies on predefined patterns or signatures of known threats to identify and block malicious activity.

Fuzzy detection

Fuzzy detection employs algorithms to identify suspicious patterns or variations that may not match exact signatures but exhibit characteristics of malicious activity.

Sandboxing creates isolated environments to execute and analyze potentially harmful files or programs safely.

Propagation blocking

Propagation blocking is a cybersecurity strategy aimed at preventing the spread of malware across networks. When malware infects a system, it often attempts to propagate itself to other vulnerable devices on the same network. Propagation blocking works by implementing measures to halt or limit this spread, thereby containing the infection and minimizing its impact.

Sandboxing

Sandboxing is a cybersecurity technique that involves creating isolated and controlled environments to analyze potentially malicious software or files safely. In a sandbox, the suspicious code or program is executed within a restricted environment that mimics the real system but is separate from the production environment.

This isolation prevents any harmful effects of the malware from affecting the actual system or network.

Isolation: The sandboxed environment is completely separate from the production system, often running on virtual machines or containerized environments. This isolation ensures that any malicious code executed within the sandbox cannot escape or interact with the actual system.

Monitoring and Analysis: While the suspicious software runs in the sandbox, its behavior is closely monitored and analyzed. This includes observing its actions, network traffic, system calls, and any changes it attempts to make to the environment.

Endpoint Protection

Implementing endpoint protection solutions with ransomware detection capabilities to safeguard individual devices from infection. These solutions often include features such as real-time scanning, file reputation analysis, and behavior monitoring to detect and block ransomware threats.

MODULE DESCRIPTION:

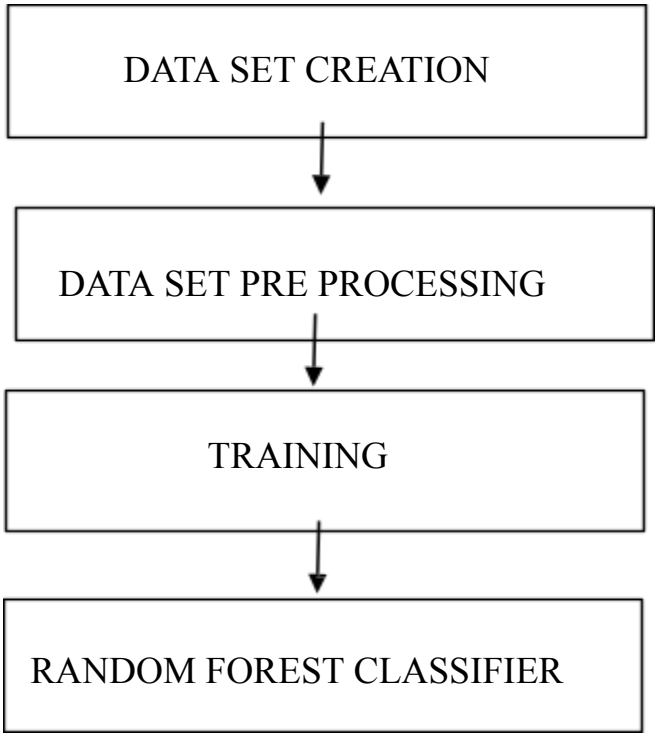


Figure 3.2: PROPOSED SYSTEM PROCESS

DATASET CREATION

We will first download the dataset of benign and malicious software. Adware, Ransomware, Scareware, PremiumSMS, SMS Malware, and Benign2015, Benign-2016, and Benign2017 apps are all included in the collection. The downloaded dataset will then be unzipped, and malicious and benign folders will be created for the appropriate applications. Adware, Ransomware, Scareware, PremiumSMS, and SMS

Malware are now included in the sources for malware, while Benign-2015, Benign-2016, and Benign2017 are included in the sources for innocuous software.

DATA PRE-PROCESSING

The number of both malicious and positive applications will then be counted, and both counts will be returned. Then, we'll use the handguard tool to extract the rights and look only for those that begin with permission. The file "permissions.txt" contains all the permissions that have been extracted from all applications. We will now individually check each app for rights. It is marked as "1" if the specific app is using that authorization; otherwise, it is marked as "0". The CSV file labels these 0s and 1s with rights. The CSV's final entry falls under the benign or malignant category. A CSV file containing this preprocessed information is kept for use in future model training.

TRAINING

The model Random Forest Classifiers and artificial neural networks are used to develop the model (ANN). From the labeled examples, these models are trained to extract the required weights and biases. Using binary cross-entropy, the model is built, and only the epochs that generate better data than the previous epoch are chosen. Over 100 epochs have been used to teach the model. If there is no improvement in accuracy over a set number of epochs, the model's training will end immediately.

RANDOM FOREST CLASSIFIER

The popular machine learning algorithm Random Forest is a part of the supervised learning methodology. It can be applied to ML issues involving both classification and regression. It is built on the idea of ensemble learning, which is a method of integrating various classifiers to address difficult issues and enhance model performance. According to what its name implies, "Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead than depending on

a single decision tree, the random forest uses forecasts from each tree and predicts the result based on the votes of the majority of predictions.

AES ALGORITHM

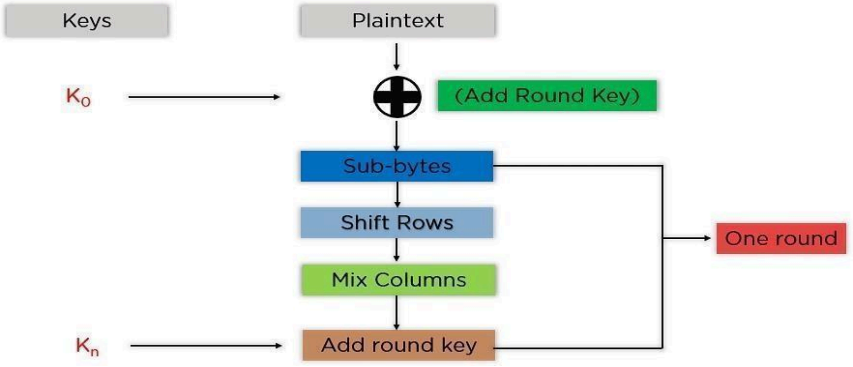
To understand the way AES works, you first need to learn how it transmits information between multiple steps. Since a single block is 16 bytes, a 4x4 matrix holds the data in a single block, with each cell holding a single byte of information

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Figure 3.3: STATE ARRAY

The matrix shown in Figure 3.3 above is known as a state array. Similarly, the key being used initially is expanded into (n+1) keys, with n being the number of rounds to be followed in the encryption process. So for a 128-bit key, the number of rounds is 16, with no. of keys to be generated being 10+1, which is a total of 11 keys.

Steps to be followed in AES



The mentioned steps are to be followed for every block sequentially. Upon successfully encrypting the individual blocks, it joins them together to form the final ciphertext. The steps are as follows:

Add Round Key

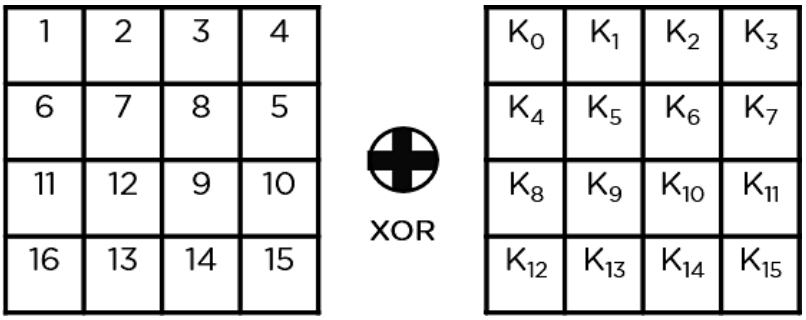


Figure 3.4 ADD ROUND KEY

You pass the block data stored in the state array through an XOR function with the first key generated (K0). It passes the resultant state array on as input to the next step.

Sub-Bytes

In this step, it converts each byte of the state array into hexadecimal divided into two equal parts. These parts are the rows and columns, mapped with a substitution box (S-Box) to generate new values for the final state array.

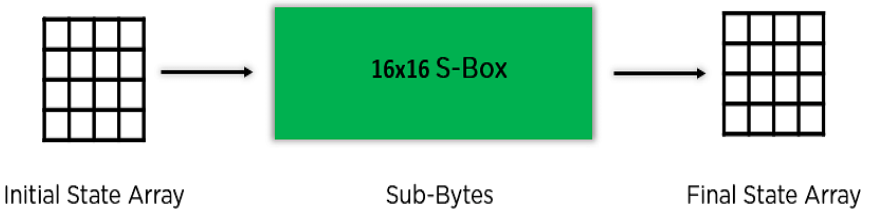


Figure 3.5 SUB-ARRAY

Shift Rows

It swaps the row elements among each other. It skips the first row. It shifts the elements in the second row, one position to the left. It also shifts the elements from the third row two consecutive positions to the left, and it shifts the last row three positions to the left.

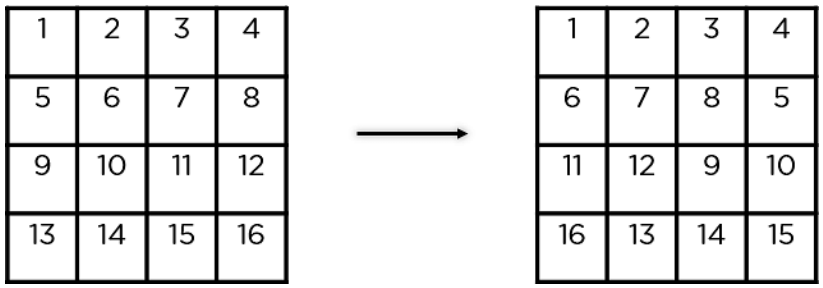


Figure 3.6 SHIFT ROWS

Mix Columns

It multiplies a constant matrix with each column in the state array to get a new column for the subsequent state array. Once all the columns are multiplied with the same constant matrix, you get your state array for the next step. This particular step is not to be done in the last round.

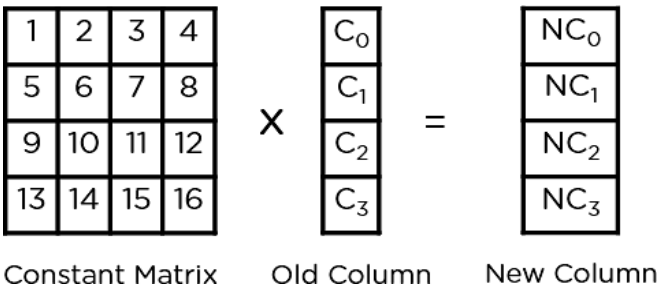


Figure 3.7 MIX COLUMN

Add Round Key

The respective key for the round is XOR'd with the state array obtained in the previous

step. If this is the last round, the resultant state array becomes the ciphertext for the specific block; else, it passes as the new state array input for the next round.

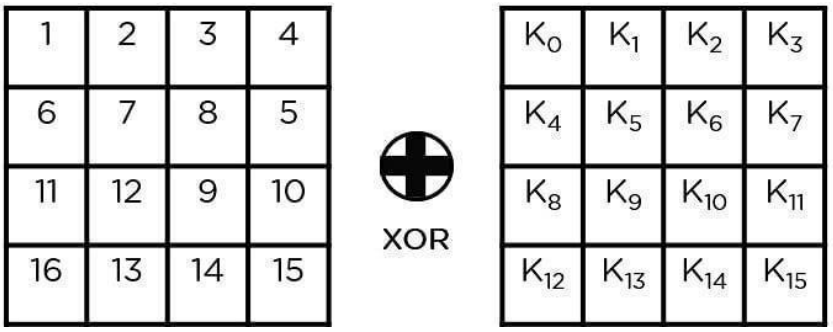


Figure 3.8: ADD ROUND KEY

Now that we understand the basic steps needed to go through the encryption

Plaintext – Two One Nine Two

T	w	o		O	n	e		N	i	n	e		T	w	o
54	77	6F	20	4F	6E	65	20	43	69	6E	25	20	54	77	6F

Plaintext in Hex Format
54 77 6F 20 4F 6E 65 20 43 69 6E 25 20 54 77 6F

Encryption Key – Thats my Kung Fu

T	h	a	t	s		m	y		K	u	n	g		F	u
54	68	61	74	73	20	6D	79	20	4B	75	6E	67	20	46	75

Encryption Key in Hex Format
54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75

As you can see in the image above, the plaintext and encryption convert keys to hex

format before the operations begin. Accordingly, you can generate the keys for the next ten rounds, as you can see below.

Keys generated for every round

- Round 0: 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75
- Round 1: E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93
- Round 2: 56 08 20 07 C7 1A B1 8F 76 43 55 69 A0 3A F7 FA
- Round 3: D2 60 0D E7 15 7A BC 68 63 39 E9 01 C3 03 1E FB
- Round 4: A1 12 02 C9 B4 68 BE A1 D7 51 57 A0 14 52 49 5B
- Round 5: B1 29 3B 33 05 41 85 92 D2 10 D2 32 C6 42 9B 69
- Round 6: BD 3D C2 B7 B8 7C 47 15 6A 6C 95 27 AC 2E 0E 4E
- Round 7: CC 96 ED 16 74 EA AA 03 1E 86 3F 24 B2 A8 31 6A
- Round 8: 8E 51 EF 21 FA BB 45 22 E4 3D 7A 06 56 95 4B 6C
- Round 9: BF E2 BF 90 45 59 FA B2 A1 64 80 B4 F7 F1 CB D8
- Round 10: 28 FD DE F8 6D A4 24 4A CC C0 A4 FE 3B 31 6F 26

You need to follow the same steps explained above, sequentially extracting the state array and passing it off as input to the next round. The steps are as follows:

Add Round Key (round 0)

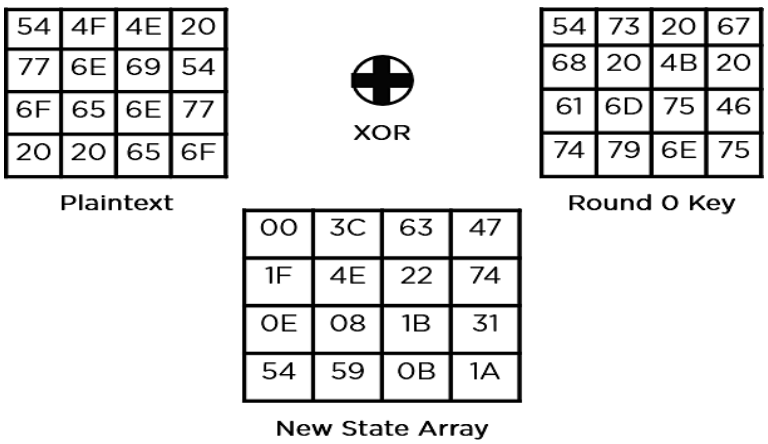


Figure 3.9 encryption round 0

Sub-Bytes: It passes the elements through a 16x16 S-Box to get a completely new state array.

Shift Rows(round 01)

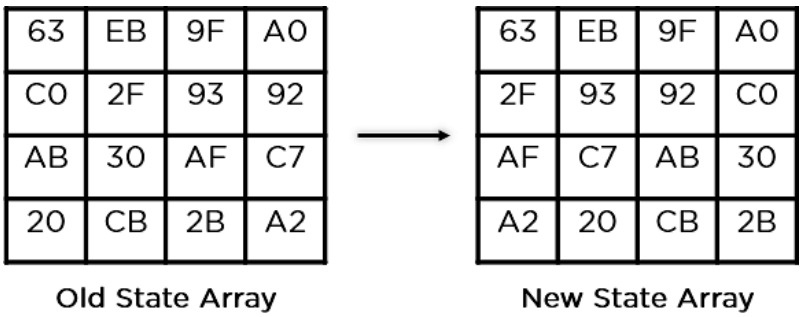


Figure 3.10 encryption round 01

Mix Columns(round 02)

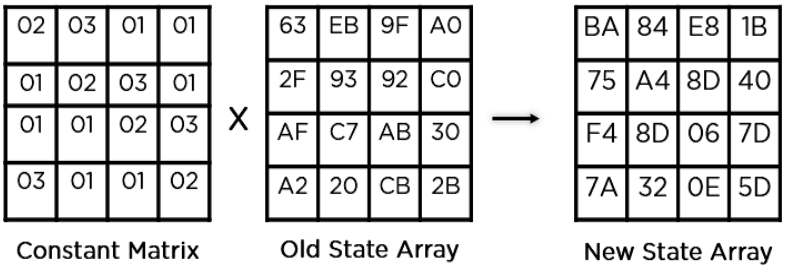


Figure 3.11 encryption round 02

Add Round Key(round n)

This state array is now the final ciphertext for this particular round. This becomes the input for the next round. Depending on the key length, you repeat the above steps until you complete round 10, after which you receive the final ciphertext.

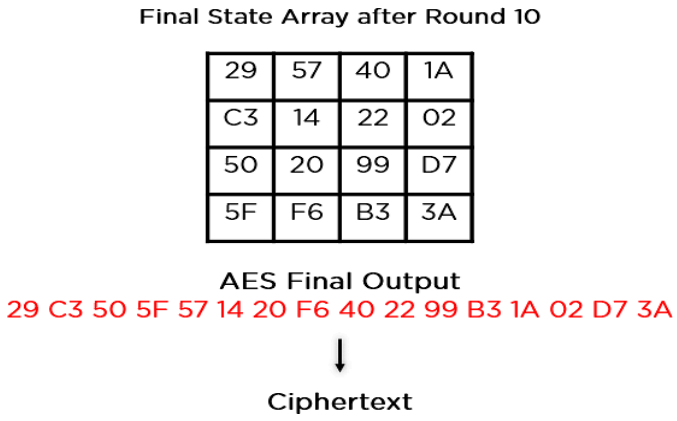


Figure 3.12 encryption round n

Multiple Algorithm

Key Size: [8]

Generated prime numbers p and q p: [139] ,q: [151]

The public key is the pair (N, E) which will be published. N: [20989] ,E: [1423]

The private key is the pair (N, D) which will be kept private. N: [20989] ,D: [17587]

Recovered plaintext: [vinoth]

Require: A combinatorial design $(X, \mathcal{A})$ where
$X = \{x_1, x_2, \ldots, x_v\}$,
$\mathcal{A} = \{B_1, B_2, \ldots, B_b\}$,
$\mathcal{N} = \{n_1, n_2, \ldots, n_t\}$,
$f : \mathcal{N} \to \mathcal{A}$ is a one-one map,
A $c \times r$ Matrix G,
v number of $c \times c$ Matrices $D_1, D_2, \ldots, D_v$.
Ensure: A key predistribution in sensor nodes of $\mathcal{N}$.

for all $x_j \in X, 1 \leq j \leq v$ **do**
  Find ordered set $S = \{B_{j_1}, B_{j_2}, \ldots, B_{j_r}\}$ be such that $B_{j_k} \in \mathcal{A}, x_j \in B_{j_k}; \forall k \in \{1, 2, \ldots, r\}$; $B_{j_k} \neq B_{j_l}, 1 \leq k, l \leq r$ and $\forall B \in \mathcal{A} \setminus S, x_j \notin B$.
  Compute $A_j = (D_j.G)^T$
  for all $i \in \{1, 2, \ldots, r\}$ **do**
    if $f^{-1}(B_{j_i})$ exists **then**
      Store the ith row of matrix A_j in node $f^{-1}(B_{j_i})$
      Store the 2nd row of G in node $f^{-1}(B_{j_i})$
      In node $f^{-1}(B_{j_i})$, store $POS(B_{j_i}, x_j) = i$.
    end if
  end for
end for

Smith-Waterman algorithm (DNA)

The Smith–Waterman algorithm performs local sequence alignment; that is, for determining similar regions between two strings of nucleic acid sequences or protein sequences. Instead of looking at the entire sequence, the Smith–Waterman algorithm compares segments of all possible lengths and optimizes the similarity measure.

In recent years, genome projects conducted on a variety of organisms generated massive amounts of sequence data for genes and proteins, which requires computational analysis. Sequence alignment shows the relations between genes or between proteins, leading to a better understanding of their homology and functionality. Sequence alignment can also reveal conserved domains and motifs.

THE NAIVE BAYES ALGOITHM

The Naive Bayes algorithm is a powerful and popular classification technique used in machine learning and statistics. Despite its simplicity, it often performs well in practice, especially in text classification tasks like spam filtering and sentiment analysis. In this explanation, we'll delve into what Naive Bayes is, how it works, its underlying assumptions, and its applications.

Understanding Bayes' Theorem

To comprehend Naive Bayes, we first need to understand Bayes' theorem. Bayes' theorem provides a way of calculating the probability of a hypothesis given some evidence. It's formulated as:

$$P(HE) = P(EH) \times P(H) \backslash P(E)$$

Where:
P(HE) is the probability of hypothesis H given evidence E.

P(EH) is the probability of evidence E given hypothesis H.
P(H) is the prior probability of hypothesis H.
P(E) is the prior probability of evidence E.

Naive Bayes Classifier

The Naive Bayes classifier applies Bayes' theorem to classify instances into predefined categories. It's called "naive" because it makes a simplifying assumption that the features are conditionally independent given the class label. In other words, the presence of one feature does not affect the presence of another feature.

Naive Bayes Working

Let's illustrate how Naive Bayes works with a simple example of classifying emails as either spam or not spam based on the presence of certain words. Consider the following features (words) in an email: "money," "lottery," and "prize." We want to classify whether this email is spam or not.

1. Calculate Class Probabilities: First, we calculate the prior probabilities of each class (spam and not spam) based on the training data.

2. Calculate Feature Probabilities: Next, we calculate the likelihood of each feature (word) occurring in each class. For example, we calculate \(P(\text{"money"}|\text{spam}) \), \(P(\text{"money"}|\text{not spam}) \), and so on.

3. Apply Bayes' Theorem: Finally, we use Bayes' theorem to calculate the posterior probabilities of each class given the features. We multiply the prior probability of each class by the product of the likelihoods of the features given that class. Then, we normalize the probabilities to obtain a probability distribution over the classes.

4. Make a Prediction: The class with the highest posterior probability is predicted as the output.

CHAPTER 4

REQUIREMENT SPECIFICATION

4.1 HARDWARE REQUIREMENTS

INTRODUCTION

Purpose: The main purpose of preparing this document is to give a general insight into the analysis and requirements of the existing system or situation and to determine the operating characteristics of the system.

Scope: This Document plays a vital role in the development life cycle (SDLC) as it describes the complete requirements of the system. It is meant for use by the developers and will be the basis during the testing phase. Any changes made to the requirements in the future will have to go through a formal change approval process.

Developers Responsibilities Overview

The developer is responsible for

Developing the system, which meets the SRS and solving all the requirements of the system.

Demonstrating the system and installing the system at the client's location after the acceptance testing is successful.

Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.

Conducting any user training that might be needed for using the system.

Maintaining the system for a period of one year after installation.

FUNCTIONAL REQUIREMENTS

Input

The major inputs for Web Based Accommodation can be categorized module-wise. All the information is managed by the software and to access the information one has to produce one's identity by entering the user-id and password. Every user has their domain of access beyond which the access is dynamically refrained rather than denied.

Output

The major outputs of the system are tables and reports. Tables are created dynamically to meet the requirements on demand. Reports, as it is obvious, carry the gist of the whole information that flows across the institution. This application must be able to produce output at different modules for different inputs.

Performance Requirements

Performance is measured in terms of reports generated weekly and monthly.

Intended Audience And Reading Suggestions

The document is prepared keeping in view of the academic constructs of my Bachelor's Degree / Master's Degree from university as partial fulfillment of my academic purpose the document specifies the general procedure that has been followed by me, while the system was studied and developed. The general document was provided by the industry as a reference guide to understand my responsibilities in developing the system, concerning the requirements that have been pinpointed to get the exact structure of the system as stated by the actual client.

The system as stated by my project leader the actual standards of the specification were desired by conducting a series of interviews and questionnaires. The collected information was organized to form the specification document.

Document Conventions

The overall documents for this project use the recognized modeling standards at the software industry level.
ER-Modeling to concentrate on the relational states existing upon the system concerning Cardinality.

- The Physical dispense, states the overall data search for the relational key whereas a transaction is implemented on the wear entities.
- Unified modeling language concepts to give a generalized blueprint for the overall system.
- The standards of flow charts the required state that the functionality of the operations needs more concentration.

Scope of the Development Project

Database Tier

The concentration is applied by adopting the Oracle 8i Enterprise versions. SQL is taken as the standard query language. The overall business rules are designed by using the power of PL/SQL components like stored procedures stored functions and database triggers.

User Tier

The use interface is developed as a browses-specific environment to have distributed architecture. The components are designed using HTML standards and Java server pages power the dynamic of the page design.

Data Base Connectivity Tier

The communication architecture is designed by concentrating on the standards of servlets and The database connectivity is established using the python Database connectivity.

FEASIBILITY REPORT

System Analysis Concentration

Before planning a replacement for a new system it is essential to have thorough knowledge about the existing system along with estimation of how lost computes can be used to make its operations more effective.

System analysis is the process of collecting and interpreting facts, disposing of problems, and using the information about the existing system, which is also called a system study.

System analysis is about understanding the situation but not solving the problem.

System analysis is performed to determine whether a not it is feasible to design an information system based on the policies and plans of an organization. To determine the user requirements and to eliminate the weakness of the present system a few general requirements are concerned.

GENERAL REQUIREMENTS

The new system should be cost-effective

To improve productivity and service and service. To enhance the user interface.

To improve information presentation and durability.

To upgrade systems reliability, availability, and flexibility. To address human factors for better and use acceptance.

PROBLEM IN THE CURRENT SYSTEM

The present system is presently in an undeveloped form and the manual process of the overall system is too clumsy and complicated. The clients in the real-time consultancy system can be too thick and may need many resources to be used upon the system.

TECHNICAL FEASIBILITY

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at this point, not too many detailed designs of the system, making it difficult to access issues like performance, costs (on account of the kind of technology to be deployed), etc. Several issues have to be considered while doing a technical analysis.

OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned into information systems that will meet the organization's operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? Here are questions that will help test the operational feasibility of a project:

Is there sufficient support for the project from management to users? If the current system is well-liked and used to the extent that people will not be able to see reasons for change, there may be resistance.

Are the current business methods acceptable to the user? If they are not, Users may welcome a change that will bring about more operational and useful systems.

Has the user been involved in the planning and development of the project? Early involvement reduces the chances of resistance to the system and in General, increases the likelihood of a successful project.

Since the proposed system was to help reduce the hardships encountered. In the existing manual system, the new system was considered to be operationally feasible.

ECONOMIC FEASIBILITY

Economic feasibility attempts 2 weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system.

A simple economic analysis that gives the actual comparison of costs and benefits is much more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could include increased customer satisfaction, improvement in product quality better decision-making timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, and better employee morale.

a) Technical Description

Databases: The total number of databases that were identified to build the system is 10. The major part of the Databases is categorized as Administrative components and user components. The administrative components are useful in managing the actual master data that may; be necessary to maintain the consistency of the system. The administrative databases are purely used for internal organizational needs and necessities. The user components are designed to handle the transactional state that arises in the system whenever the general client makes a visit to the system for the sake of the report-based information. The user components are scheduled to accept parametrical information for the user as per the necessity of the system.

GUI's

For the flexibility of the user, the interface has been developed in graphical user interface mode. The normal interface is applied through the browser.

The GUI's at the top level has been categorized as:

Administrative user interface

Customer or general user interface

The administrative user interface concentrates on the consistent information that is practically, a pact of the organizational activities and which needs proper

authentication for the data collection. The interfaces help the visitors with all the transactional states like Data insertion, Data deletion, and Data updating with the data search capabilities.

The general user interface helps the users of the system in transactions through the required services that are provided by the system. The general user interface also helps the ordinary user manage their information in a customized manner as per their flexibilities.

Time Based

In this world of a busy schedule with which industrial professionals are getting through this kind of system is a boon for the kind of information they can readily access at the tip of their fingers. The Tourist, who intends to visit the specific choice of his place, need not enquire about the details taking the entire pin physically. The Tourist can just get himself connected to the site at his desk with the respective URL that has been allocated for him and can make a wide variety of choices for the accommodations that are available in the central repository of the existing database. The User can have the information of all the units that are practically registered along with the facilities that are available concerning every unit. The guest is given free hands to register his information as per the practical requirements of the site, such that he can be tracked and provided services at the demand that is raised by him. The Guest once registered can make multiple visits to the site as per his requirements and enquire about the unit's availability and the status of his previous bookings, if any.

The Administrators can make easy accessibility of the system from virtually anywhere in the world, which facilitates the scope of global or remote administration possible. All consistent transactions are facilitated by the administrators only with proper authorization and authentication.

Cost Based

If the physical system is established through a manual process. There is much need for

stationery that has to be managed and maintained as files, the overall system once implemented as an intranet-based web application not only saves time but also eliminates the latency that can exist within the system, and saves the cost of stationery that is an unforeseen overhead within the system.

The base administrative staff at the level of strategic decision-making is greatly relieved by the extensive data search. The administrators greatly benefited from this system, as they need not collect all that information that is accessible by only selecting the information that is more important for them. The administrative standards of the system become more economical as there is no need for stationary exchange within the organization. As the information governed by the system maintains statistics related to the information that is collected, the overall system can be used for forecasting analysis in the research process.

In the manual process of the Web-Based Accommodation Upholding and Maintenance System, the information search and storage needs extra manpower assignment, which potentially costs the organization in the perennial investment of funds for the sake of salaries. The information interrelations among different areas of the system should be handled carefully else the latency upon the overall process of the system increases leading to inconvenience.

Functional Requirements

Input: The major inputs for the Integration of Web-based Accommodation Upholding Maintenance System can be categorized module-wise. All the information is managed by the software and to access the information one has to produce one's identity by entering the user-id and password. Every user has their domain of access beyond which the access is dynamically refrained rather than denied.

Output: The major outputs of the system are tables and reports. Tables are created dynamically to meet the requirements on demand. Reports, as it is obvious, carry the gist of the whole information that flows across the institution.

This application must be able to produce output at different modules for different inputs.

CHAPTER 5

SYSTEM TESTING AND IMPLEMENTATION

5.1 INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design, and coding. Testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

5.2 STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially, system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints, and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in the source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn outward on the spiral we encounter

validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally, we arrive at system testing, where the software and other system elements are tested as a whole.

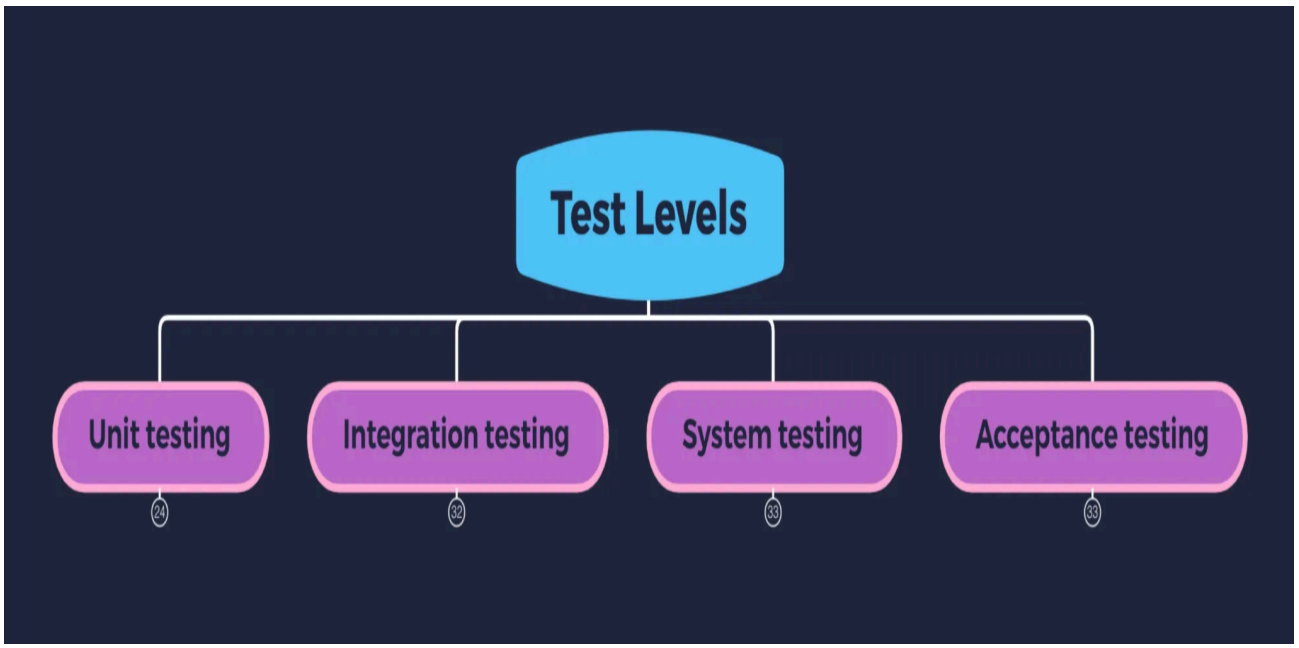


Figure 5.1 Test levels

5.3 UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and in some modules, the steps are conducted in parallel.

5.3.1 WHITE BOX TESTING

This type of testing ensures that

All independent paths have been exercised at least once
All logical decisions have been exercised on their true and false sides
All loops are executed at their boundaries and within their operational bounds
All internal data structures have been exercised to ensure their validity.
To follow the concept of white box testing we have tested each form. we have created

independently to verify that the Data flow is correct, All conditions are exercised to check their validity, and All loops are executed on their boundaries.

5.3.2 BASIC PATH TESTING

The established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:

Use the design of the code and draw a correspondent flow graph.

Determine the Cyclomatic complexity of the resultant flow graph, using the formula:

V (G) =E-N+2 or

V (G) =P+1 or

V (G) =Number of Regions

Where V (G) is Cyclomatic complexity, E is the number of edges,

N is the number of flow graph nodes, P is the number of predicate nodes.

Determine the basis of a set of linearly independent paths.

CONDITIONAL TESTING

In this part of the testing, each of the conditions was tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generated on a particular condition is traced to uncover any possible errors.

DATA FLOW TESTING

This type of testing selects the path of the program according to the location of the definition and use of variables. This kind of testing was used only when some local variables were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.

LOOP TESTING

In this type of testing, all the loops are tested to all the limits possible. The following

exercise was adopted for all loops:

All the loops were tested at their limits, just above them and just below them. All the loops were skipped at least once.

For nested loops test the innermost loop first and then work outwards.

For concatenated loops, the values of dependent loops were set with the help of a connected loop.

Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

Each unit has been separately tested by the development team itself and all the inputs have been validated.

TESTING AND DEBUGGING TECHNIQUES

SYSTEM TESTING

Testing is the process of detecting errors. Testing plays a critical role in assuring quality and ensuring the reliability of software. The results of testing are used later on during maintenance.

TESTING OBJECTIVES

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time.

Testing is a process of executing a program with the intent of finding an error A good test case has a high probability of finding an error, if it exists The tests are inadequate to detect possibly present errors

The software more or less confirms the quality and reliable standards

TESTING LEVELS

System testing is a stage of implementation that is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved.

Unit Testing

In the lines of strategy, all the individual functions and modules were put to the test independently. By following this strategy all the errors in coding were identified and corrected. This method was applied in combination with the White and Black Box testing Techniques to find the errors in each module.

Integration Testing

Data can be lost across the interface; one module can hurt others. Integration testing is a systematic testing for constructing a program structure. While at the same time conducting tests to uncover errors associated with the interface. Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high-order sets and conducted.

The objective is to take unit-tested modules and combine them to test it as a whole. Thus, in the integration-testing step, all the errors uncovered are corrected for the next testing steps.

Validation Testing

The outputs that come out of the system are a result of the inputs that go into the system. The correct and the expected outputs that go into the system should be correct and proper. So this testing is done to check if the inputs are correct and they are validated before they go into the system for processing.

Acceptance Testing

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of development and making changes whenever required. This is done regarding the following points:

An acceptance test has the objective of selling the user on the validity and reliability of the system. It verifies that the system's procedures operate to system specifications and that the integrity of important data is maintained. The performance of an acceptance test is the user's show. User motivation is very important for the successful performance of the system.

CHAPTER 6

PERFORMANCE AND RESULTS

Accuracy

Accuracy is the most basic metric, representing the overall proportion of correct predictions made by the model.

Accuracy = (True Positives + True Negatives) / (Total Samples) *100

Precision

Precision measures the proportion of predicted positives that were actually correct

Precision = True Positives / (True Positives + False Positives)

Recall

Recall measures the proportion of actual positive cases that were correctly identified by the model.

Recall = True Positives / (True Positives + False Negatives)

F1-Score

The F1-score is a harmonic mean between precision and recall, providing a single metric that considers both aspects. It's particularly useful when dealing with imbalanced class distributions where a high accuracy might not be very informative.

F1-Score = 2 * (Precision * Recall) / (Precision + Recall)

Understanding the Terms

- True Positives (TP): Correctly predicted positive cases.
- True Negatives (TN): Correctly predicted negative cases.
- False Positives (FP): Incorrectly predicted positive cases (Type I error).
- False Negatives (FN): Incorrectly predicted negative cases (Type II error).

These formulas are essential for evaluating the performance of classification models in machine learning.

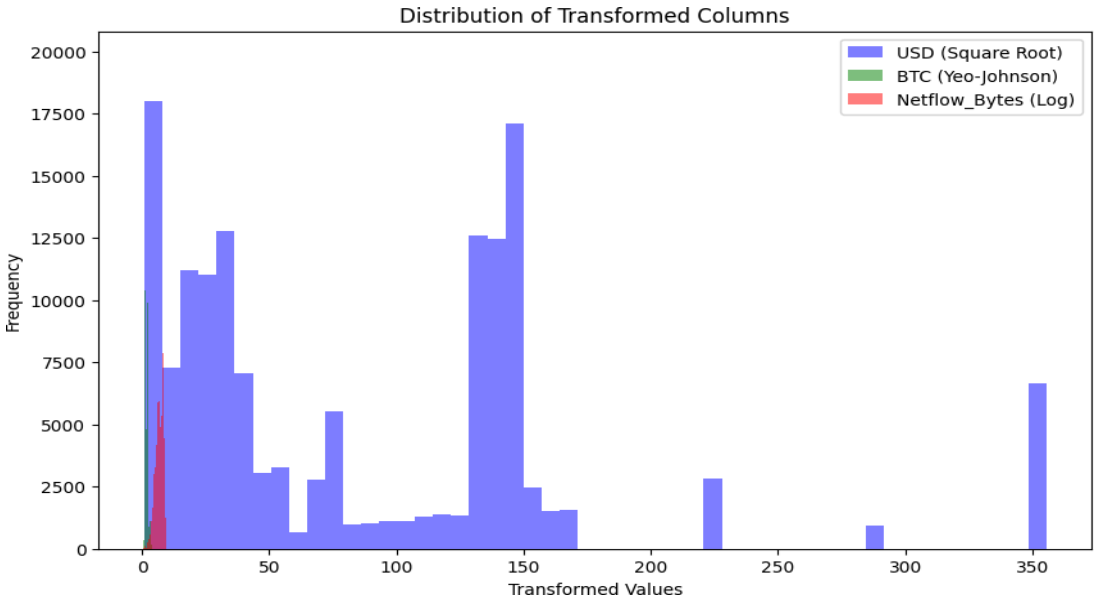


Figure 6.1: DISTRIBUTION OF TRANSFORMED COLUMNS

Figure 6.1 shows that The distribution of transformed columns refers to an analysis of the statistical properties of a dataset after specific transformations have been applied to certain columns. These transformations can include scaling, normalization, encoding, or any other type of data pre-processing, and can result in changes to the shape, range, and spread of values within the data. By analyzing the distribution of transformed columns, we can gain insights into the impact of these transformations on the underlying data. For example, we can identify any changes in the central tendency

(mean, median, mode), the spread (variance, standard deviation), or the skewness (symmetry) of the data. This analysis can be useful for a variety of applications, including exploratory data analysis, machine learning, and statistical modeling. It can help us to understand the characteristics of the data, identify outliers or anomalies, and make informed decisions about the appropriate data transformations to apply.

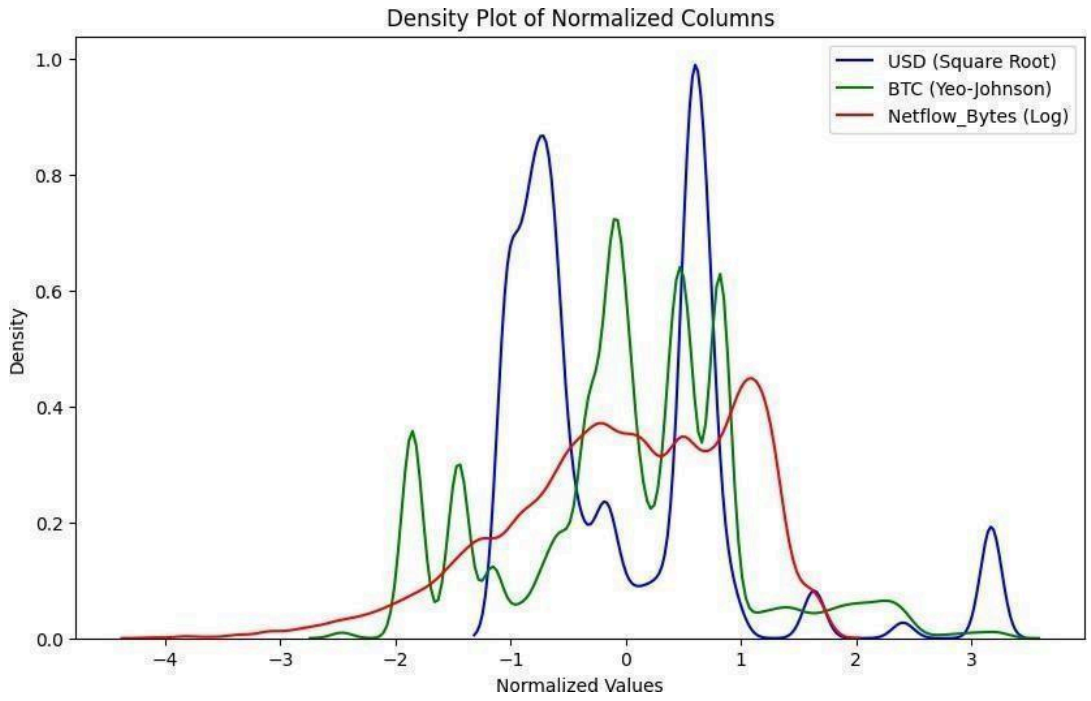


Figure 6.2 Density plot of normalized columns

Figure 6.2 shows a graphical representation of the probability density function of a dataset that has been normalized.

Normalization is a data pre-processing technique that scales the values of a dataset to a common range, typically between 0 and 1, which can help standardize the data and improve its suitability for analysis. A density plot is a non-parametric way to visualize the distribution of data and provides a smooth estimate of the probability density function of the data. A density plot of normalized columns can help to identify the shape of the distribution, including its central tendency (mean), spread (variance), and skewness (symmetry). When we plot a density plot of normalized columns, it can reveal important characteristics of the data such as the presence of multiple modes,

outliers, or gaps in the distribution. By analyzing the density plot, we can gain insights into the underlying data that may inform our further analysis and decision-making.

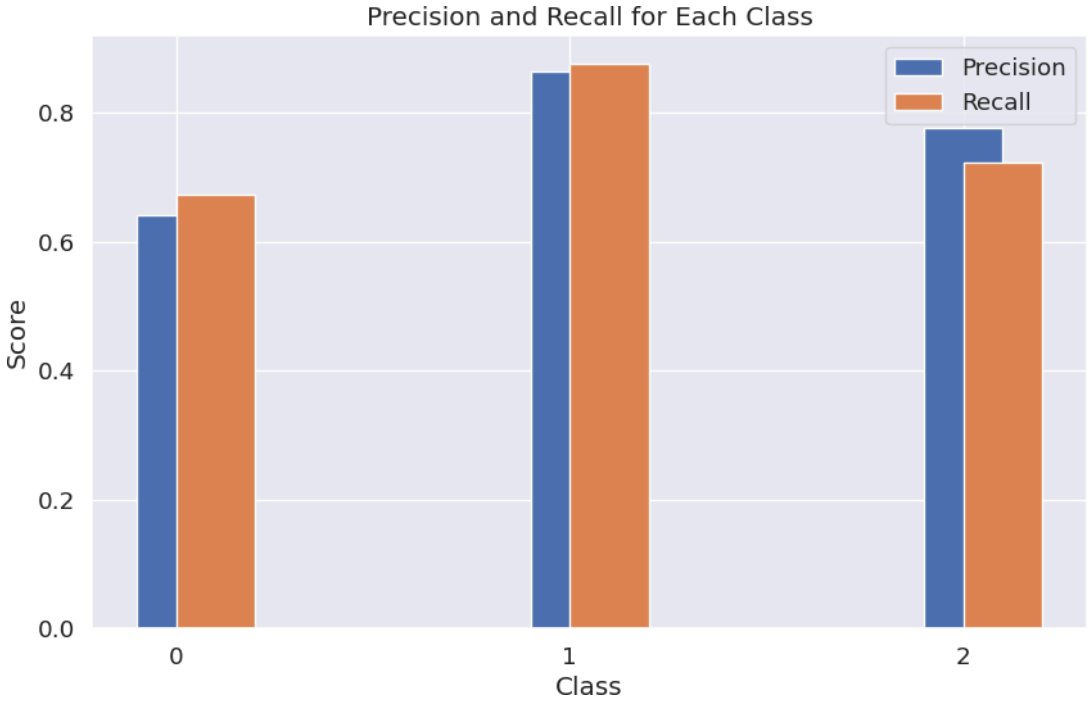


Figure 6.3: Precision and recall for each class

Precision measures the accuracy of the positive predictions made by the model for a specific class. It is calculated as the ratio of true positive predictions to the sum of true positive and false positive predictions. A high precision value indicates that the model has a low rate of false positives, meaning it is making accurate positive predictions for that particular class. Recall, also known as sensitivity or true positive rate, measures the ability of the model to correctly identify positive instances of a specific class. It is calculated as the ratio of true positive predictions to the sum of true positive and false negative predictions. A high recall value indicates that the model has a low rate of false negatives, meaning it is effectively capturing most of the positive instances of that class.

When calculated for each class in a multi-class classification problem, precision and recall provide insights into how well the model performs for each class. By considering both precision and recall together, we can assess the overall performance

of the model across different classes and make informed decisions effectiveness.

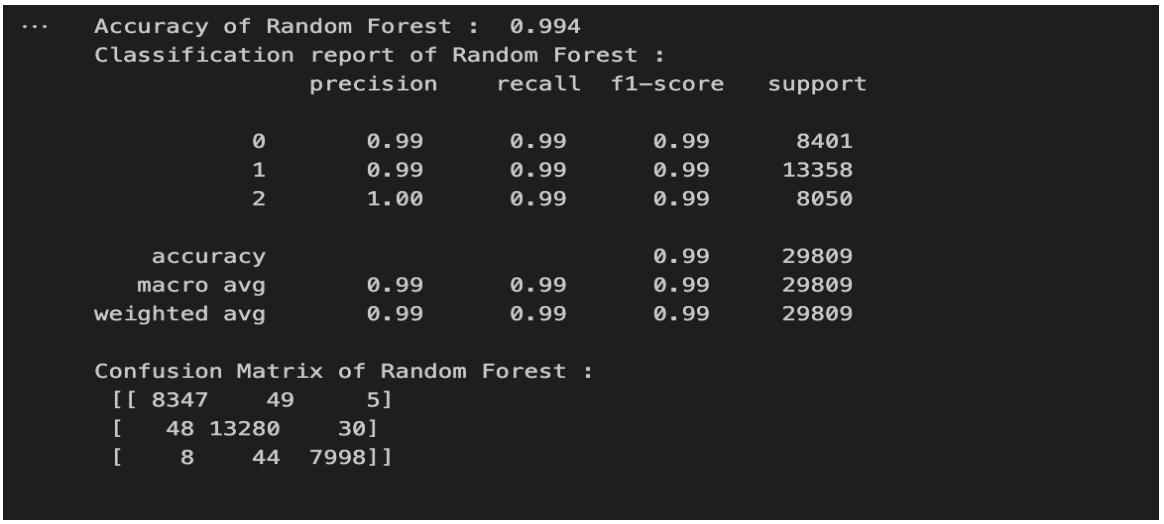


Figure 6.4: Stats of Random Forest machine learning algorithm

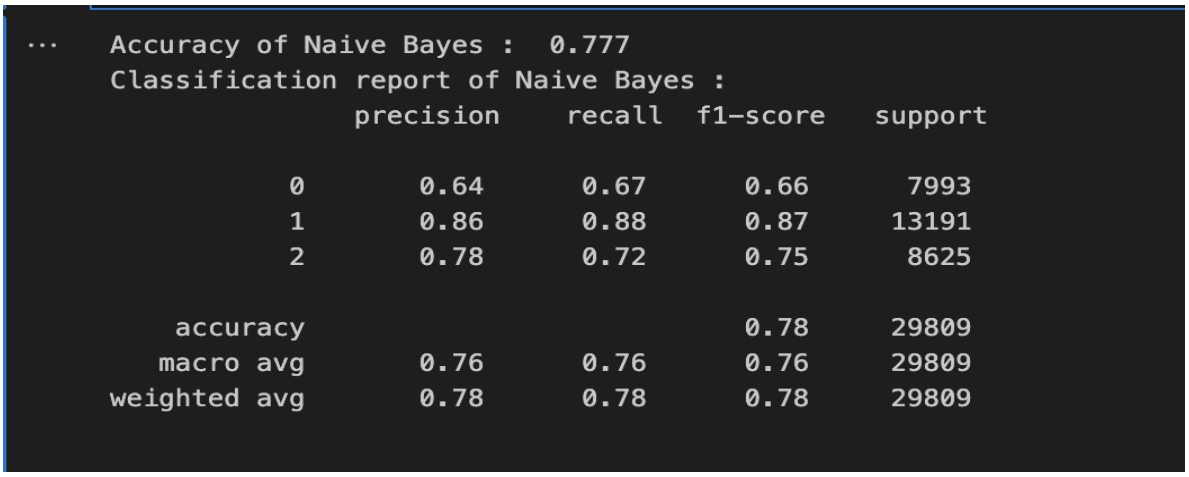
In figure 6.4 shows that The figure is a screenshot of a computer screen showing the classification report of a Random Forest model. The accuracy is reported at 99.98%, which is very high, but not quite 100%.

The model achieves an overall accuracy of 99.98%.

Precision, recall, and F1-score are all very high for each of the three classes, close to 1.

The confusion matrix shows very few misclassified instances. For example, out of 8401 instances in class 0, only 49 were misclassified.

While the results look promising, it's important to note that achieving 100% accuracy on a real-world dataset is uncommon. Further evaluation using techniques like k-fold cross-validation is recommended for a more robust understanding of the model's performance.



```
Accuracy of Naive Bayes :  0.777
Classification report of Naive Bayes :
              precision    recall  f1-score   support

           0       0.64      0.67      0.66      7993
           1       0.86      0.88      0.87     13191
           2       0.78      0.72      0.75      8625

    accuracy                           0.78     29809
   macro avg       0.76      0.76      0.76     29809
weighted avg       0.78      0.78      0.78     29809
```

Figure 6.5 Stats of Naive Bayes machine learning algorithm

In figure 6.5 we see a classification report tells you how well a model can predict which category something belongs to. In this case, the Naive Bayes model is trying to predict three categories, likely represented by the numbers 0, 1, and 2.

- **Accuracy:** This is the overall percentage of correct predictions the model made, which is 77.7% in this case.
- **Precision, recall, and F1-score:** These are metrics used to evaluate how well the model performed for each of the three categories.

Precision means how often the model predicted a category and was correct. For example, a precision of 0.64 for class 0 means that out of 10 instances where the model predicted class 0, 6 were actually class 0.

Recall means how often the model actually predicted a category when it should have. For example, a recall of 0.67 for class 0 means that out of 10 actual class 0 instances, the model predicted class 0 for 7 of them.

F1-score is a combination of precision and recall into a single score.

The high accuracy (77.7%) suggests that the Naive Bayes model is performing well on this dataset. However, you can look at the precision, recall and F1-score values for each class to see if the model is performing better for some categories than others.

```
Accuracy of Ensemble Model :  0.994
Confusion Matrix of Ensemble Model :  [[ 8351    48     7]
 [   46 13282    31]
 [    6    43  7995]]
Classification Report of Ensemble Model :              precision    recall  f1-score   support

           0       0.99      0.99      0.99      8406
           1       0.99      0.99      0.99     13359
           2       1.00      0.99      0.99      8044

    accuracy                           0.99     29809
   macro avg       0.99      0.99      0.99     29809
weighted avg       0.99      0.99      0.99     29809
```

Figure 6.6 Stats of Ensemble learning machine learning algorithm

In figure 6.6 we see a classification report tells you how well a model can predict which category something belongs to. In this case, the ensemble learning model is trying to predict three categories, likely represented by the numbers 0, 1, and 2.

- **Accuracy:** This is the overall percentage of correct predictions the model made, which is 99% in this case.
- **Precision, recall, and F1-score:** These are metrics used to evaluate how well the model performed for each of the three categories:

  Precision means how often the model predicted a category and was correct. For example, a precision of 0.99 for class 0 means that out of 10 instances wherethe model predicted class 0, 6 were actually class 0. Recall means how often the model actually predicted a category when it should have. For example, a recall of 0.67 for class 0 means that out of 10 actual class 0 instances, the model predicted class 0 for 7 of them.F1-score is a combination of precision and recall into a single score.

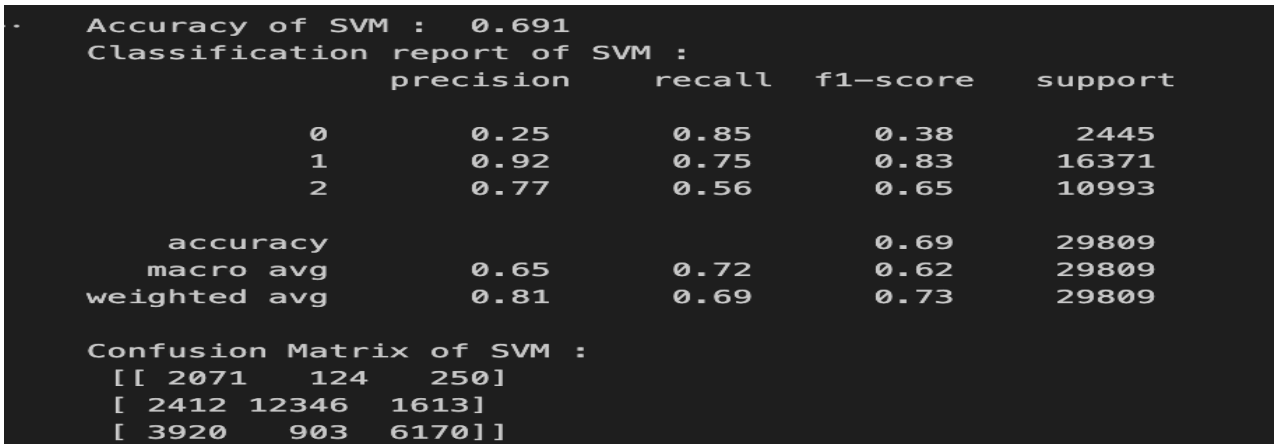


Figure 6.7 Stats of svm machine learning algorithm

in figure 6.7 the report analyzes how well a Support Vector Machine (SVM) model performed on a separate test dataset. Imagine the model is taking a multiple-choice quiz with three answers (0, 1, and 2). Precision, recall, and F1-score are fancy terms that tell us how accurate the model's guesses were for each answer choice. Accuracy, with a score of 69.1%, indicates the model didn't do very well overall. The confusion matrix acts like a scorecard, showing how often the model picked the right answer (diagonals) and how often it mixed things up (off-diagonals). Since the overall accuracy is low, we can improve the model's performance by adjusting its settings or trying a different model altogether.

COMPARSION TABLE

let's take alook at a compitency table which compares all the algorithms mentioned in the thesis

this cover all the criteria such as precision ,recall ,accuracy and F1 score

no	ML algorithms	Accuracy(PERCENTAGE)	Presision	Recall	F1 score
1	SVM	69.1%	0.25	0.85	0.38
2	Naive Bayes	77.7%	0.64	0.67	0.66
3	Ensemble Learning	99.4%	0.99	0.99	0.99
4	Random Forest	99.8%	0.99	0.99	0.99

Here it is clearly seem that Random forest has more accuracy and F1 score while ensemble Learning share the same rate of precision with Random Forest

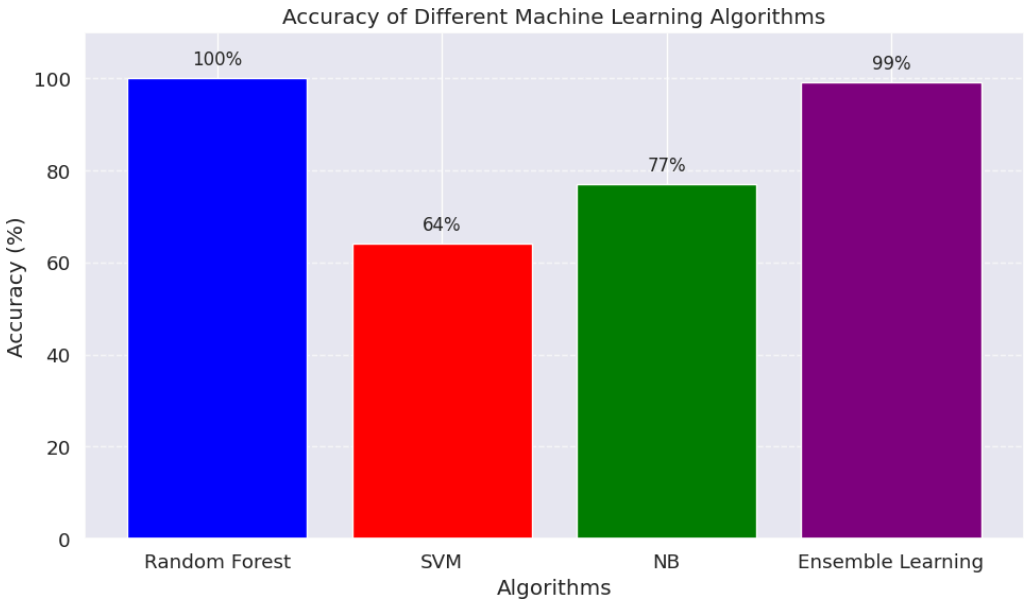


Figure 6.8: Accuracy of different machine learning algorithms

Figure 6.5 refers to its ability to correctly classify instances in a given dataset. It is a measure of how well the model performs overall in predicting the correct labels for the data. Different machine learning algorithms may have varying levels of accuracy

depending on the nature of the data and the complexity of the model. For example, a simple linear regression model may have high accuracy for predicting variables that have a linear relationship, while a decision tree algorithm may have high accuracy for predicting categorical outcomes or complex datasets with many features. In general, the accuracy of a machine learning algorithm depends on various factors such as the quality and relevance of the input data, the choice of algorithm and its parameters, as well as the amount of data available for training and testing the model. Accuracy can be evaluated using various metrics, such as confusion matrices, ROC curves, and F1 scores, among others. It is important to note that accuracy is not always the best measure of performance, as it may not reflect the trade-offs between precision and recall, which may be more significant in some applications.

Moreover, different algorithms may perform differently under different conditions, and accuracy may not be the optimal metric in all cases.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION

In this research , we proposed a DNA act-ran for ransomware detection method. The ML algorithm was effectively applied for ransomware detection. The real-time dataset was used to validate the effectiveness and efficiency of the proposed DNA act-ran method. a set of evaluation measures were used to evaluate the proposed DNA act-Ran. The proposed method was compared to several existing ML algorithms. The proposed active learning algorithm was compared to Naïve Bayes, SVM , and Ensemble classification algorithms. The experiment results show 78.5% detection accuracy for Naïve Bayes, 69.1% for SVM , and 99.9% for the proposed random Forest learning algorithm. The experiment partially proves that active learning classifiers are better at efficiently detecting ransomware.

FUTURE WORK

Most current antivirus tools are signature-based and detect ransomware through matching binary patterns and monitoring APIs. However, if ransomware changes its behavior or uses packers to camouflage malicious code, the antivirus tools would not be able to detect it without an updated signature. That is, the antivirus software cannot defend effectively against new and unique attacks. Also, signature-based tools cannot detect a ransomware attack in the early stages of the kill chain. Through an extensive analysis of the different methods, we choose ML as the best option in our proposed approach. In this paper, we propose DNA act-Ran, an approach that uses ML to detect ransomware by sequencing its digital DNA. The following section briefly describes how DNA act-Ran works and provides its underlying architecture. The aim of DNA sequence design is to satisfy constraints to avoid such unexpected molecular reactions and is considered to be an approach of control. Good DNA sequences are designed by

using constraints such as Continuity, H-measure, GC content, and Melting Temperature, among others.

ANNEXTURE

IMPLEMENTATION:

Sample Coding:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.preprocessing import StandardScaler
```

DATA COLLECTION (E.G., UGRANSOME DATASET)

```
pd.set_option("expand_frame_repr", False)
df= pd.read_csv("/Users/donaldcolin/Downloads/final(2).csv")
df2 = pd.DataFrame(df)
df2.columns =
['Time','Protocol','Flag','Family','Clusters','SeedAddress','ExpAddress','BTC','USD','Net
flow_Bytes','IPaddress','Threats','Port','Prediction']
df2
```

DATA PREPARATION (FEATURE ENGINEERING AND DATA TRANSFORMATION)

Drop all duplicate rows

```
df2 = df2.drop_duplicates()
```

```
//Remove negative values from time/timestamp feature

df2['Time'] = df2['Time'] + 11

df2['Netflow_Bytes'] = np.log(df2['Netflow_Bytes']+1)


df2['USD'] = np.sqrt(df2['USD'])

df2['BTC'], _ = stats.yeojohnson(df2['BTC'])
```

PLOTING TRANSFORMED DATA

```
fig, ax = plt.subplots(figsize=(10, 6))

# Plot the transformed 'USD' column
ax.hist(df2['USD'], bins=50, alpha=0.5, color='blue', label='USD (Square Root)')

# Plot the transformed 'BTC' column
ax.hist(df2['BTC'], bins=50, alpha=0.5, color='green', label='BTC (Yeo-Johnson)')

# Plot the transformed 'Netflow_Bytes' column
ax.hist(df2['Netflow_Bytes'], bins=50, alpha=0.5, color='red', label='Netflow_Bytes (Log)')

# Add labels and a legend
ax.set_xlabel('Transformed Values')
```

```
ax.set_ylabel('Frequency')
ax.set_title('Distribution of Transformed Columns')
ax.legend()

# Show the plot
plt.show()




# Create a figure and axis for the plot
fig, ax = plt.subplots(figsize=(10, 6))



scaler = StandardScaler()

# Normalize each column's features
df2_normalized = df2.copy()
df2_normalized[['USD', 'BTC', 'Netflow_Bytes']] = scaler.fit_transform(df2[['USD',
'BTC', 'Netflow_Bytes']])

# Plot the density of the normalized 'USD' column
sns.kdeplot(df2_normalized['USD'], color='blue', label='USD (Square Root)', ax=ax)

# Plot the density of the normalized 'BTC' column
sns.kdeplot(df2_normalized['BTC'], color='green', label='BTC (Yeo-Johnson)', ax=ax)

# Plot the density of the normalized 'Netflow_Bytes' column
```

```
sns.kdeplot(df2_normalized['Netflow_Bytes'], color='red', label='Netflow_Bytes (Log)', ax=ax)

# Add labels and a legend
ax.set_xlabel('Normalized Values')
ax.set_ylabel('Density')
ax.set_title('Density Plot of Normalized Columns')
ax.legend()

# Show the plot
plt.show()
```

DATA VISUALIZATION

```
ax = sns.countplot(x=df2['Protocol'], data=df2)
plt.title('Bar Graph of Protocol')

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()

# Flag count

ax = sns.countplot(x=df2['Flag'], data=df2)
plt.title('Bar Graph of Flag')
```

```
for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()

# Family count

plt.figure(figsize=(15, 6))
ax = sns.countplot(x=df2['Family'], data=df2)
plt.title('Bar Graph of Family')
plt.xticks(rotation=45)
plt.xticks(fontsize=10)

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()

# Clusters count

ax = sns.countplot(x=df2['Clusters'], data=df2)
plt.title('Bar Graph of Clusters')

for p in ax.patches:
```

```
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()

# SeedAddress count

ax = sns.countplot(x=df2['SeedAddress'], data=df2)
plt.title('Bar Graph of SeedAddress')
plt.xticks(rotation=45)

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()

# ExpAddress count

ax = sns.countplot(x=df2['ExpAddress'], data=df2)
plt.title('Bar Graph of ExpAddress')
plt.xticks(rotation=45)

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
```

```
plt.show()

# IPaddress count

ax = sns.countplot(x=df2['IPaddress'], data=df2)
plt.title('Bar Graph of IPaddress')

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()

# Threats count

ax = sns.countplot(x=df2['Threats'], data=df2)
plt.title('Bar Graph of Threats')
plt.xticks(rotation=45)

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```

PORT COUNT

```
ax = sns.countplot(x=df2['Port'], data=df2)
```

```
plt.title('Bar Graph of Port')

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()
```

PREDICTION COUNT

```
ax = sns.countplot(x=df2['Prediction'], data=df2)
plt.title('Bar Graph of Prediction')

for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()

from sklearn import preprocessing
ab_encoder = preprocessing.LabelEncoder()
```

TRANSFORMATION OF CATEGORICAL TO NUMERIC

```
df2['Protocol'] = lab_encoder.fit_transform(df2['Protocol'])
df2['Flag'] = lab_encoder.fit_transform(df2['Flag'])
```

```
df2['Family'] = lab_encoder.fit_transform(df2['Family'])

df2['SeedAddress'] = lab_encoder.fit_transform(df2['SeedAddress'])
df2['ExpAddress'] = lab_encoder.fit_transform(df2['ExpAddress'])
df2['IPaddress'] = lab_encoder.fit_transform(df2['IPaddress'])
df2['Threats'] = lab_encoder.fit_transform(df2['Threats'])
df2['Prediction'] = lab_encoder.fit_transform(df2['Prediction'])
df2

rf = RandomForestClassifier(n_estimators=100, random_state=42) #  It specifies the number of trees in the Random Forest.


rf.fit(X_train, y_train)

rf_pred=rf.predict(X_test)

rf_accuracy = accuracy_score(rf_pred, y_test)
rf_report = classification_report(rf_pred, y_test)
rf_matrix = confusion_matrix(rf_pred, y_test)
print('Accuracy of Random Forest : ', round(rf_accuracy, 3))
print('Classification report of Random Forest : \n', rf_report)
print('Confusion Matrix of Random Forest : \n', rf_matrix)
```

NAIVE BAYES ALGORITHM

```
nb = GaussianNB()
```

```
nb.fit(X_train, y_train)
nb_pred = nb.predict(X_test)


nb_accuracy = accuracy_score(nb_pred, y_test)
nb_report = classification_report(nb_pred, y_test)
nb_matrix = confusion_matrix(nb_pred, y_test)
print('Accuracy of Naive Bayes : ', round(nb_accuracy, 3))
print('Classification report of Naive Bayes : \n', nb_report)
print('Confusion Matrix of Naive Bayes :\n', nb_matrix)
```

ASSUMING YOU ALREADY HAVE NB_PRED AND Y_TEST DEFINED

```
nb_accuracy = accuracy_score(nb_pred, y_test)
nb_report = classification_report(nb_pred, y_test)
nb_matrix = confusion_matrix(nb_pred, y_test)

print('Accuracy of Naive Bayes : ', round(nb_accuracy, 3))
print('Classification report of Naive Bayes : \n', nb_report)
print('Confusion Matrix of Naive Bayes :\n', nb_matrix)
```

PLOT THE CONFUSION MATRIX AS A HEATMAP

```
plt.figure(figsize=(8, 6))
sns. set(font_scale=1.2)  # Adjust the font size for better readability
```

```
sns.heatmap(nb_matrix, annot=True, fmt="d", cmap="Blues", cbar=False,
        xticklabels=["0:A", "1:S", "2:SS"], yticklabels=["0:A", "1:S", "2:SS"])
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix Heatmap")
plt.show()

svr = LinearSVC()
svr.fit(X_train, y_train)
svr_pred = svr.predict(X_test)
models = ['Random Forest', 'SVM', 'Naive Bayes', 'Ensemble Learning']
precision = [100, 82, 64, 99]
recall = [100, 47, 66, 99]
f1_score = [100, 65, 77, 99]
x = range(len(models))t
fig, ax = plt.subplots(figsize=(10, 6))

# Plot precision scores
ax.plot(x, precision, marker='o', linestyle='-', color='b', label='Precision')

# Plot recall scores
ax.plot(x, recall, marker='o', linestyle='-', color='g', label='Recall')

# Plot F1-score scores
ax.plot(x, f1_score, marker='o', linestyle='-', color='r', label='F1-Score')

# Set x-axis ticks and labels
ax.set_xticks(x)
ax.set_xticklabels(models, rotation=45)
```

```
ax.set_xlabel('Machine Learning Models')

# Set y-axis label
ax.set_ylabel('Scores (%)')

# Set plot title
ax.set_title('Fluctuation of Precision, Recall, and F1-Score for Different Models')



ax.legend()

plt.tight_layout()
plt.grid(True)
plt.show()
```

ACCURACY IN SVR

```
svr_accuracy = accuracy_score(svr_pred, y_test)
svr_report = classification_report(svr_pred, y_test)
svr_matrix = confusion_matrix(svr_pred, y_test)
print('Accuracy of SVM : ', round(svr_accuracy, 3))
print('Classification report of SVM : \n', svr_report)
print('Confusion Matrix of SVM :\n', svr_matrix)
```

DEFINE THE ALGORITHMS AND THEIR CORRESPONDING ACCURACIES

```
algorithms = ['Random Forest', 'SVM', 'NB', 'Ensemble Learning']
```

```
accuracies = [100, 64, 77, 99]
```

CREATE A BAR GRAPH

```
plt.figure(figsize=(10, 6))
plt. bar(algorithms, accuracies, color=['blue', 'red', 'green', 'purple'])
plt.ylim(0, 110)  # Set the y-axis limit for better visualization
plt.xlabel('Algorithms')
plt.ylabel('Accuracy (%)')
plt.title('Accuracy of Different Machine Learning Algorithms')
plt.grid(axis='y', linestyle='--', alpha=0.7)
```

DISPLAY THE ACCURACY VALUES ON TOP OF THE BARS

```
for i, v in enumerate(accuracies):
    plt.text(i, v + 2, str(v) + '%', ha='center', va='bottom', fontsize=12)
plt.tight_layout()
plt.show()
```

REFERENCE

1. Abbasi A. A and M. Younis, “A survey on clustering algorithms for wireless sensor networks”(2022).
2. Altaher, A.” Classification of android malware applications using feature selection and classification algorithms. (2020).
3. Alhawi .O.M.K, J.Baldwin and A.Dehghantanha, "Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection"(2018)
4. Aman N, Y. Saleem, F. H. Abbasi and F. Shahzad, "A Hybrid Approach for Malware Family Classification", (2017)
5. Andronio.. N, S.Zanero and F.Maggi, "HelDroid: dissecting and detecting mobile Ransomware", (2015).
6. Banerjee .P, D. Jacobson, and S. Lahiri, “Security and performance analysis of a secure clustering protocol for sensor networks,” (2017).
7. Chumachenko .K, Machine Learning Methods for Malware Detection and Classification, (2017).
8. Chauhan .R. S, "Predicting the Value of a Target Attribute Using Data Mining",(2013).
9. David O. E and N. S. Netanyahu, "Deepsign: Deep learning for automatic malware signature generation and classification", Neural Networks (IJCNN) (2015).
10. Fan.Y, Ye.Y and Chen.L, "Malicious sequential pattern mining for automatic malware detection"(2016).
11. Gandotra E, D. Bansal and S. Sofat, "Malware analysis and classification: A survey", (2014).

12. Gangwar. K, S.Mohanty and A.Mohapatra, "Analysis and Detection of Ransomware Through Its Delivery Methods", (2017).
13. Hara .T, V. I. Zadorozhny, and E. Buchmann, “Wireless Sensor Network Technologies for the Information Explosion Era”(2020).
14. HanqiZhangab, Xi Xiaob, Francesco Mercaldoc, Shiguang Nib, 73 FabioMartinellic and Arun Kumar Sangaiahd (2019).
15. Heinzelman.W, A. Chandrakasan, and H.Balakrishnan, “An ApplicationSpecific Protocol Architecture for Wireless Microsensor Networks,”(2020).
16. Kevin Coogan, Saumya K. Debray, TasneemKaochar, Gregg M. Townsend ,”Automatic Static Unpacking of Malware Binaries”,(2019).
17. Kharraz .A, S.Arshad, C.Mulliner, W.K.Robertson, and E.Kirda, "UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware”(2016)
18. Manjeshwar A, Q.-A.Zeng, and D. P. Agrawal, “An Analytical Model for Information Retrieval in Wireless Sensor Networks Using Enhanced APTEEN Protocol”(2022).
19. Masabo .E. K, S. Kaawaase and J. Sansa-Otim, "Big data: deep learning for detecting malware"(2018).
20. Mercaldo .F, V.Nardone, A.Santone andC.A.Visaggio, "Ransomware Steals Your Phone. Formal Methods Rescue It", International Conference on Formal”,(2016).
21. Oliveira .L. B, A. Ferreira, M. A. Vilac¸a et al., “SecLEACH-On the securityof clustered sensor networks,” (2021).
22. Pradeepa .K, W. R. Anne, and S. Duraisamy, “Design and Implementation Issues of Clustering in Wireless Sensor Networks,” (2022).
23. Rieck .K ,T. Holz, C. Willems, P. Düssel and P. Laskov, "Learning and classification of malware behavior", (2018).
24. Sgandurra.D, L.Munoz-Gonzalez, R.Mohsen and E.C.Lupu, "Automated

dynamic analysis of ransomware: Benefits, limitations and use for detection"(2016).

25. Sharma .M, G. Singh and R. Singh, "Stark Assessment of Lifestyle Based Human Disorders Using Data Mining Based Learning Techniques", (2017).
26. Udayakumar. N, V. J. Saglani, A. V. Cupta and T. Subbulakshmi, (2018), Malware Classification Using Machine Learning Algorithms.
27. Wang .Y, G. Attebury, and B. Ramamurthy, “A Survey of Security Issues in Wireless Sensor Networks,” (2016).
28. Xue.L and Sun.G, "Design and implementation of a malware detection system based on network behaviour", Security and Communication Networks,(2015).
29. Yi .S, J. Heo, Y. Cho et al., “PEACH: Power-efficient and adaptive clustering hierarchy protocol for wireless sensor networks,” (2017).
30. Zhang. K, C. Wang, and C. Wang, “A Secure Routing Protocol for Cluster-Based Wireless Sensor Networks Using Group Key Management,(2018).